Prof. Dr. Jasmin Blanchette
Jannis Limperg
Lydia Kondylidou

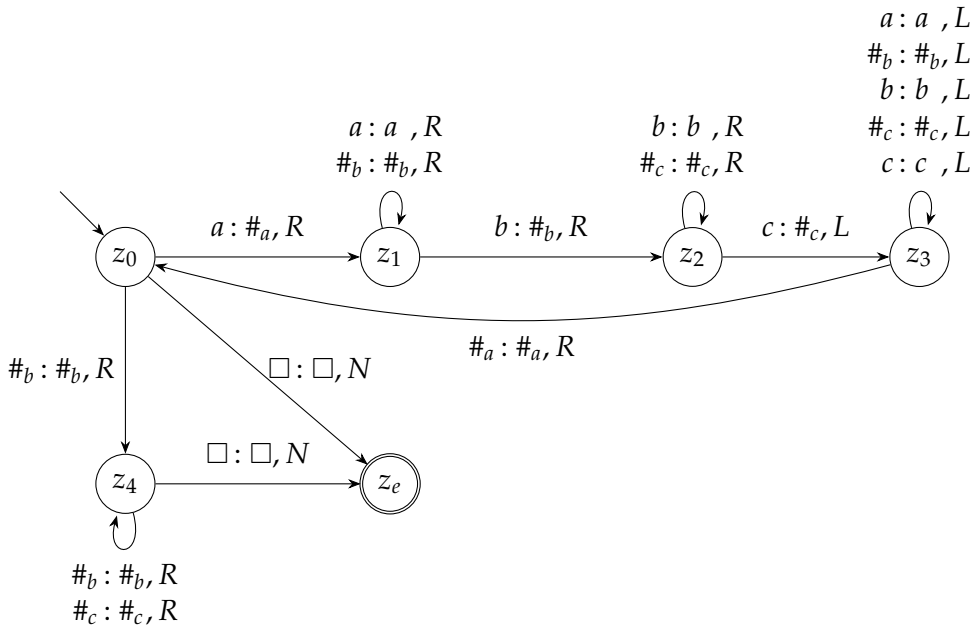
Ludwig-Maximilians-Universität München
Institut für Informatik
Abgabe bis 21. Juni 2023, 10:00 Uhr

Lösungsvorschlag zur Übung 9 zur Vorlesung

Formale Sprachen und Komplexität

FSK9-1 ***Turingmaschinen verstehen*** (2 Punkte)

Die folgende DTM[1] T ist als Zustandsgraph gegeben, wobei $\Sigma = \{a, b, c\}$, $\Gamma = \Sigma \cup \{\#_a, \#_b, \#_c, \square\}$ und $\square$ das Blank-Symbol ist.



LÖSUNGSVORSCHLAG:

Simulator: `http://turingmachinesimulator.com/shared/ykmvnxarwx`

a) Geben Sie Läufe der Turingmaschine (Übergänge von der Startkonfiguration bis zur Endkonfiguration) für die Wörter ε, $abcc$ und abc an.

[1] Im Zustandsgraph ist nicht für jeden Zustand z und jedes Zeichen a ein Übergang definiert. Wir betrachten die Übergangsfunktion δ also als partielle Funktion. Wenn $\delta(z, a)$ undefiniert ist, hält die Maschine an und akzeptiert nicht. Das ist angenehmer, als einen Müllzustand einzuführen, um δ total zu machen.

LÖSUNGSVORSCHLAG:

- $z_0\square \vdash_T z_e\square$
- $z_0abcc \vdash_T \#_a z_1 bcc \vdash_T \#_a\#_b z_2 cc \vdash_T \#_a z_3\#_b\#_c c \vdash_T z_3\#_a\#_b\#_c c$
  $\vdash_T \#_a z_0\#_b\#_c c \vdash_T \#_a\#_b z_4\#_c c \vdash_T \#_a\#_b\#_c z_4 c$
- $z_0abc \vdash_T \#_a z_1 bc \vdash_T \#_a\#_b z_2 c \vdash_T \#_a z_3\#_b\#_c \vdash_T z_3\#_a\#_b\#_c$
  $\vdash_T \#_a z_0\#_b\#_c \vdash_T \#_a\#_b z_4\#_c \vdash_T \#_a\#_b\#_c z_4\square \vdash_T \#_a\#_b\#_c z_e\square$

b) Welche Sprache akzeptiert die Turingmaschine T? Begründen Sie Ihre Antwort.

LÖSUNGSVORSCHLAG:

Die Turingmaschine akzeptiert $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. Dazu geht sie wie folgt vor:

- In den Zuständen z_0, z_1 und z_2 wird ein a, dann ein b und dann ein c gesucht, wobei jeweils ein gefundener Buchstabe x durch $\#_x$ ersetzt wird. Dabei überspringt die TM in z_1 nur a's und $\#_b$'s und in z_2 nur b's und $\#_c$'s, das heißt es müssen tatsächlich zuerst die a's, dann die b's und dann die c's im Wort auftreten.
- Im Zustand z_3 fährt die TM zurück zum ersten a, das noch nicht verarbeitet wurde. Somit werden in einer Schleife alle a's mit b's und c's gepaart.
- Wenn die TM in z_0 ein $\#_b$ liest, sind alle a's abgearbeitet. Wir müssen dann nur noch sicherstellen, dass keine b's oder c's mehr im Wort sind, und laufen deswegen noch einmal über $\#_b$'s und $\#_c$'s bis an das Wortende.
- Spezialfall: wenn wir in z_0 ein $\square$ lesen, ist das Wort leer und die TM akzeptiert sofort.

In einer alten Version des Blatts war $\Sigma = \{a, b, c, \#_a, \#_b, \#_c\}$ definiert. In diesem Fall akzeptiert die Turingmaschine zusätzlich noch andere Wörter, z.B. alle Wörter aus der Sprache $L(\#_b(\#_b \mid \#_c)^*)$.

c) Eine Turingmaschine T mit Alphabet Σ und Bandalphabet Γ berechnet eine (partielle) Funktion $f\colon \Sigma^* \to \Gamma^*$, wenn für alle $u \in \Sigma^*$ und $v \in \Gamma^*$ gilt: $f(u) = v$ g.d.w. $z_0 u \vdash_T^* \cdots\square\cdots\square z_e v\square\cdots$ mit z_e Endzustand. (Beachten Sie: Diese Definition weicht leicht von der aus der Vorlesung ab, weil die Wertemenge von f nicht Σ^* ist, sondern Γ^*.)

Welche Funktion berechnet T?

LÖSUNGSVORSCHLAG:

Wie in Teilaufgabe b) gezeigt, erkennt T genau die Wörter der Form $a^n b^n c^n$. Im Endzustand sind dabei alle $a's$ durch $\#_a$'s ersetzt und analog für b und c. Allerdings hält T für $n \neq 0$ stets in einer Konfiguration, in der der Lesekopf im Endzustand nicht am Anfang des Wortes steht, wie es die obige Definition von Turing-Berechenbarkeit verlangt, sondern am Ende des Wortes. Somit berechnet T die Funktion, die ε auf sich selbst abbildet und sonst undefiniert ist.

Wenn wir (wie eigentlich intendiert) die Definition von Berechenbarkeit etwas aufweichen und erlauben, dass der Lesekopf in der Endkonfiguration an einer beliebigen Stelle steht, berechnet T folgende Funktion f:

$$f(w) = \begin{cases} \#_a^n \#_b^n \#_c^n & \text{für } w = a^n b^n c^n \\ \text{undefiniert} & \text{andernfalls} \end{cases}$$

Wir akzeptieren beide Lösungen.

d) Bestimmen Sie asymptotisch, also in O-Notation, die Anzahl der Schritte (abhängig von n), die die Turingmaschine braucht, um das Wort $w = a^n b^n c^n$ zu erkennen.

LÖSUNGSVORSCHLAG:

Wir zählen die Schritte eines erfolgreichen Laufs auf w.

- Von z_0 zu z_1 kommen wir in einem Schritt, der asymptotisch vernachlässigbar ist. Von z_1 zu z_2 und z_2 zu z_3 kommen wir in jeweils n Schritten (da wir immer das i-te a mit dem i-ten b und i-ten c paaren), also insgesamt in $2n$ oder $O(n)$ Schritten.
- Von z_3 zu z_0 kommen wir in mindestens $2n$ und höchstens $3n$, also $O(n)$ Schritten.
- Die Schleife von z_0 bis z_3 wird n-mal durchlaufen. Da jeder Durchlauf $O(n)$ Schritte benötigt, benötigt die gesamte Schleife $O(n^2)$ Schritte.
- Am Ende laufen wir via z_4 noch einmal in $2n$, also $O(n)$, Schritten über das ganze Wort.

Die gesamte Laufzeit ist also $O(n^2)$.

FSK9-2 ***Snapshot-Kellerautomaten*** (2 Punkte)

Ein *Snapshot-Kellerautomat* (Snapshot-PDA) ist ein Kellerautomat mit einem zusätzlichen „Einwegkeller“ sowie zwei zusätzlichen Aktionen SPEICHERN und LADEN. Formal ist ein Snapshot-PDA immer noch ein 6-Tupel $(Z, \Sigma, \Gamma, \delta, z_0, \#)$, aber

$$\delta\colon (Z \times \Sigma \cup \{\varepsilon\} \times \Gamma \cup \{\varepsilon\}) \to \mathcal{P}_e(Z \times (\Gamma^* \cup \{\mathsf{SPEICHERN}, \mathsf{LADEN}\}))$$

wobei SPEICHERN und LADEN keine Wörter aus Γ^* sind.

Jedes Mal, wenn die Aktion SPEICHERN ausgeführt wird, wird der aktuelle Kellerinhalt in den Einwegkeller kopiert. Jedes Mal, wenn die Aktion LADEN ausgeführt wird, wird der aktuelle Keller mit dem Inhalt des Einwegkellers überschrieben. Der Einwegkeller wird dabei geleert, also auf # gesetzt.

Eine Konfiguration eines Snapshot-PDA ist ein Quadrupel (z, w, K, T). Dabei ist z der aktuelle Zustand, $w \in \Sigma^*$ das verbleibende Wort, $K \in \Gamma^*$ der Inhalt des Kellers und $T \in \Gamma^*$ der Inhalt des Einwegkellers.

Die Übergangsrelation $\vdash$ eines Snapshot-PDA ist definiert durch:

- $(z, aw, AK, T) \vdash (z', w, WK, T)$ falls $(z', W) \in \delta(z, a, A)$;
- $(z, w, AK, T) \vdash (z', w, WK, T)$ falls $(z', W) \in \delta(z, \varepsilon, A)$;
- $(z, w, K, T) \vdash (z', w, K, K)$ falls $(z', \mathsf{SPEICHERN}) \in \delta(z, \varepsilon, \varepsilon)$;
- $(z, w, K, T) \vdash (z', w, T, \#)$ falls $(z', \mathsf{LADEN}) \in \delta(z, \varepsilon, \varepsilon)$

wobei $z, z' \in Z$; $A \in \Gamma$; $W, K, T \in \Gamma^*$; $a \in \Sigma$ und $w \in \Sigma^*$.

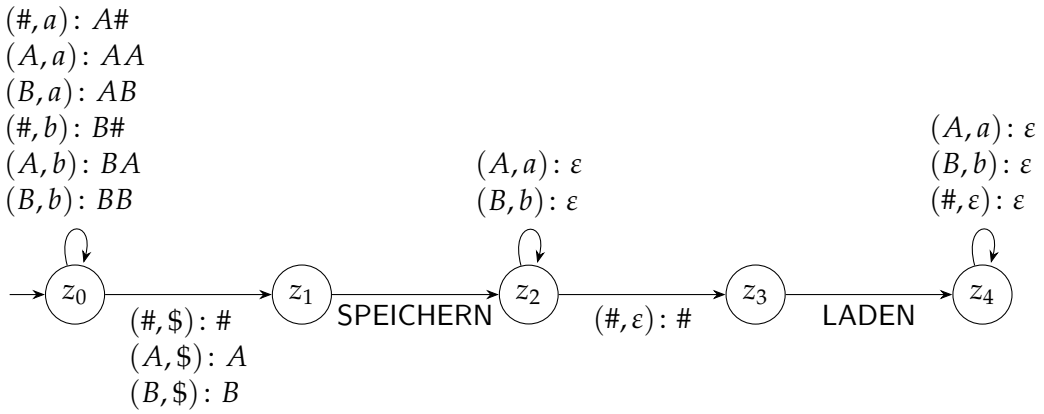
Die Übergangsrelation zeigt, dass die SPEICHERN- und LADEN-Aktionen unabhängig von Kellerinhalt und Resteingabe durchgeführt werden. Daher notieren wir sie im Zustandsgraph als mit SPEICHERN bzw. LADEN beschriftete Pfeile.

Schließlich ist die akzeptierte Sprache eines Snapshot-PDAs M

$$L(M) := \{w \in \Sigma^* \mid (z_0, w, \#, \#) \vdash^* (z, \varepsilon, \varepsilon, T) \text{ für } z \in Z, T \in \Gamma^*\}.$$

Der Automat akzeptiert also mit leerem Keller, wobei der Inhalt des Einwegkellers keine Rolle spielt.

Betrachten Sie nun den folgenden Snapshot-PDA S über dem Alphabet $\Sigma = \{a, b, \$\}$.



a) Welche Sprache erkennt S? Begründen Sie Ihre Antwort.

LÖSUNGSVORSCHLAG:

S erkennt die Sprache

$$L(S) = \{w\$\overline{w}w \mid w \in \{a, b\}^*\}.$$

Begründung: In z_0 wird das Wort w gelesen, wobei für jedes a ein A und für jedes b ein B auf den Keller gelegt wird. Der Keller enthält also nach z_0 genau w (in Großbuchstaben). Mit \$ gehen wir zu z_1 über, wobei der Keller nicht verändert wird, und speichern dann den Kellerinhalt. In z_2 bauen wir den Keller ab und lesen für jedes Symbol im Keller das entsprechende Symbol im Wort. Wenn der Keller abgebaut ist, gehen wir zu z_3 über, laden den gespeicherten Keller und bauen ihn in z_4 nochmal wie in z_3 ab. Wenn der Keller vollständig abgebaut ist und wir # erreichen, entfernen wir dieses. Wenn dann auch das Wort zuende ist, akzeptieren wir mit leerem Keller.

b) Betrachten Sie den Automaten S', der entsteht, wenn wir S wie folgt ändern.

- Entferne den Zustand z_3 und seine Übergänge.
- Füge einen LADEN-Übergang von z_2 nach z_4 hinzu.

Erkennt S' dieselbe Sprache wie S? Begründen Sie Ihre Antwort.

LÖSUNGSVORSCHLAG:

Nein, denn Snapshot-PDAs wie hier definiert sind nichtdeterministisch, d.h. S' kann das Leeren des Kellers in z_2 jederzeit abbrechen und mit dem LADEN-

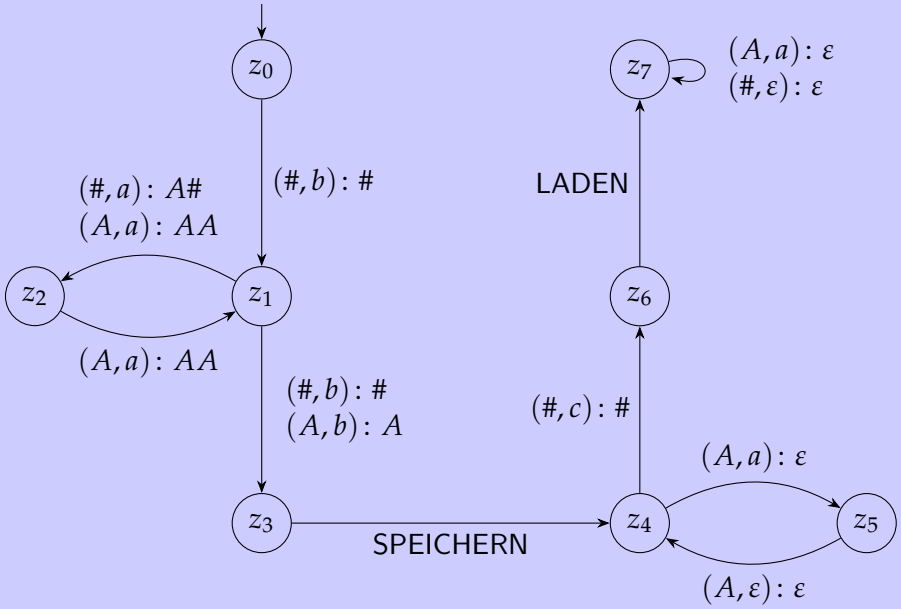
Übergang zu z_4 wechseln. Dadurch ist beispielsweise $a\$a$ in $L(S')$, aber nicht in $L(S)$.

c) Geben Sie den Zustandsgraphen eines Snapshot-PDA an, der die Sprache $L = \{ba^{2k}ba^kca^{2k} \mid k \in \mathbb{N}\}$ über dem Alphabet $\{a, b, c\}$ akzeptiert.

Erläutern Sie, wie Ihr Automat funktioniert. Welche Aufgaben übernehmen die Zustände, der Kellerinhalt und der Einwegkeller?

(Nebenbei: Wir wissen durch FSK7-2a, dass L nicht kontextfrei ist. Somit sind Snapshot-PDAs mächtiger als normale PDAs.)

LÖSUNGSVORSCHLAG:



Wir starten in z_0 und gehen mit b zu z_1 über, ohne den Keller zu verändern. Dann lesen wir in z_1 und z_2 den Wortteil a^{2k} für beliebige $k \in \mathbb{N}$ und legen dabei A^{2k} auf den Keller. Mit einem weiteren b gehen wir zu z_3 über und speichern den Keller, beides ohne den Keller zu verändern. In z_4 und z_5 bauen wir den Keller ab und lesen dabei für jedes zweite A im Keller ein a des Wortes ein; insgesamt lesen wir hier also a^k. Dann lesen wir ein c, ohne den Keller zu verändern, und laden den Einwegkeller, also A^{2k}. Schließlich bauen wir in z_7 den Keller ab und lesen dabei für jedes A im Keller ein a des Wortes ein; insgesamt also a^{2k}. Wir akzeptieren, wenn wir den Keller vollständig abbauen können und damit auch am Ende des Wortes angelangt sind. Insgesamt müssen wir also ein Wort der Form $ba^{2k}ba^kca^{2k}$ gelesen haben.

FSK9-3 ***Gummibandturingmaschinen*** (0 Punkte)

Eine Turingmaschine mit flexiblem Band ist eine deterministische Turingmaschine mit zwei zusätzlichen Bandbewegungsoperationen (also zusätzlich zu L, N, R):

- Copy: Kopiert eine Bandzelle, d.h. das Band wird an der Stelle um 1 verlängert, auf beiden Zellen ist danach das gleiche Symbol und der Kopf ist dann auf der linken dieser beiden Zellen
- Delete: Löscht eine Bandzelle, d.h. das Band wird an der Stelle um 1 verkürzt, das Symbol also komplett gelöscht; die Turingmaschine ist danach auf der Bandzelle, die vor dem Löschen rechts von dieser Zelle war

Zeigen Sie: Turingmaschinen mit flexiblen Band akzeptieren genau die gleichen Sprachen wie normale Turingmaschinen.

Formal definieren wir Turingmaschinen mit flexiblen Band analog zu den Definitionen 6.2.1 bis 6.2.5 aus dem Skript mit folgenden Anpassungen:

- Der Typ der Zustandsüberführungsfunktion δ ist nun:

$$\delta\colon (Z \times \Gamma) \to (Z \times \Gamma \times \{L, R, N, \text{Copy}, \text{Delete}\})$$

- Zur Definition der Übergangsrelation $\vdash_M$ werden folgende Fälle hinzugefügt:

$$b_1 \dots b_m\, z\, a_1 \dots a_n \vdash_M b_1 \dots b_m\, z'cc\, a_2 \dots a_n, \quad \text{wenn } \delta(z, a_1) = (z', c, \text{Copy}),\ m \geq 0, n \geq 1, z \notin E$$

$$b_1 \dots b_m\, z\, a_1 \dots a_n \vdash_M b_1 \dots b_m\, z'\, a_2 \dots a_n, \quad \text{wenn } \delta(z, a_1) = (z', c, \text{Delete}),\ m \geq 0, n \geq 2, z \notin E$$

$$b_1 \dots b_m\, z\, a_1 \vdash_M b_1 \dots b_m\, z'\, \square, \quad \text{wenn } \delta(z, a_1) = (z', c, \text{Delete}),\ m \geq 0, z \notin E$$

Sie sollen nun die folgenden Aussagen zeigen:

- Für jede Turingmaschine (ohne Anpassungen) M existiert eine Turingmaschine mit flexiblen Band M', sodass $L(M) = L(M')$ und

LÖSUNGSVORSCHLAG:

Gummibandturingmaschinen können jede Aktion einer normalen Turingmaschine ausführen, können also mindestens das ausführen, was normale Turingmaschinen können.

- für jede Turingmaschine mit flexiblen Band M' existiert eine Turingmaschine (ohne Anpassungen) M, sodass $L(M') = L(M)$.

LÖSUNGSVORSCHLAG:

Es muss gezeigt werden, dass Turingmaschinen (ohne Anpassungen) die Copy- und Delete-Operation simulieren können.

Betrachte hierfür eine gegebene Turingmaschine mit flexiblem Band M'.

Wir fügen zu M' für jeden Zustand z aus M' und jedes Symbol a aus dem Bandalphabet Γ (überschneidungsfrei) die Zustände $C_{(z,a)}, D_{(z,a)}, F_z$ und B_z hinzu.
Mit *überschneidungsfrei* ist hier und im Folgenden gemeint, dass bereits existente Zustände (oder Symbole aus dem Bandalphabet) zunächst umbenannt werden, sodass M' die neu hinzugefügten Zustände und Symbole nicht verwendet.

Zudem fügen wir (überschneidungsfrei) für jedes Symbol $a \in \Gamma$ ein neues Symbol a' zum Bandalphabet von M' hinzu. Wir definieren $\Gamma' = \{a' \mid a \in \Gamma\}$ als die Menge der neuen Symbole.

Dann fügen wir für jeden Zustand z aus M' und alle Symbole a, b aus dem Bandalphabet folgende Übergänge zu M' hinzu:

$$\delta(C_{(z,a)}, b) = \begin{cases} (B_z, a, L) & \text{falls } b = \square \\ (C_{(z,b)}, a, R) & \text{sonst} \end{cases}$$

$$\delta(D_{(z,a)}, b) = \begin{cases} (z, a, N) & \text{falls } b = c' \in \Gamma' \\ (D_{(z,b)}, a, L) & \text{sonst} \end{cases}$$

$$\delta(F_z, a) = \begin{cases} (D_{(z,\square)}, \square, L) & \text{falls } a = \square \\ (F_z, a, R) & \text{sonst} \end{cases}$$

$$\delta(B_z, a) = \begin{cases} (z, b, N) & \text{falls } a = b' \in \Gamma' \\ (B_z, a, L) & \text{sonst} \end{cases}$$

Zuletzt ersetzen wir jeden Übergang der Form $\delta(z_1, a) = (z_2, b, \text{Copy})$ durch $\delta(z_1, a) = (C_{(z_2,b)}, b', R)$ und jeden Übergang der Form $\delta(z_1, a) = (z_2, b, \text{Delete})$ durch $\delta(z_1, a) = (F_{z_2}, b', R)$.

Wir beobachten, dass nur endlich viele Zustände, Übergänge und Symbole hinzugefügt wurden.

Wir nennen die derart aus M' erzeugte Turingmaschine im Folgenden M.

Um zu prüfen, dass M' korrekt durch M simuliert wird (unsere Konstruktion also korrekt ist) müssen wir zeigen, dass $\vdash_M^* = \vdash_{M'}^*$ gilt. Hierfür genügt es sich davon zu überzeugen, dass jede Konfiguration A, die unter M' in einem Schritt in B

überführt wird, A durch M in endlichen vielen Schritten ebenfalls in B überführt wird.

Um sich die Funktionsweise der Konstruktion zu veranschaulichen können Sie die folgenden (anhand dieser Konstruktion aus Turingmaschinen mit flexiblem Band) erzeugten Maschinen online ausprobieren:

TM, die das erste Vorkommen von b verdoppelt `https://turingmachinesimulator.com/shared/bdnqdqrbgu`

TM, die das erste Vorkommen von b löscht `https://turingmachinesimulator.com/shared/zxfspdqhxl`

Intuitiv kann man sagen:

- Für Copy wird zunächst das aktuelle Symbol markiert.
  Danach wird das zusätzliche Symbol aufs Band geschrieben und alle Symbole rechts davon um eins nach rechts verschoben.
  Dann läuft M wieder nach links bis sie ihre Markierung findet und verhält sich danach weiter wie M'.
- Für Delete wird ebenfalls zunächst das aktuelle Symbol markiert.
  Es wird dann nach rechts bis zum Ende des Bands gelaufen.
  Zuletzt werden von dort aus nach links alle Symbole um eins nach links verschoben, bis M auf ihre Markierung trifft und sich danach ebenfalls weiter verhält wie M'.