

TECHNISCHE UNIVERSITÄT DRESDEN

Fachrichtung Mathematik

Institut für Algebra

**Zur Minimalisierung und Determinisierung von
sequentiellen Transducern**

Diplomarbeit

zur Erlangung des ersten akademischen Grades

Diplommathematikerin

vorgelegt von: **Ina Mäurer**

geboren am: **21. 07. 1978**

Geburtsort: **Jena**

Betreuer: **Prof. Dr. M. Droste**

Tag der Einreichung: **21. November 2002**

Zusammenfassung

Die in [3] bzw. in [15] definierten sequentiellen Transducer charakterisieren unterschiedliche Klassen von rationalen Funktionen. Die Erweiterungen der beiden Modelle zu subsequentiellen Transducern beschreiben dann allerdings identische Klassen. Entgegen der Behauptung von Mohri in [15, S. 273, Theorem 2] muss die Vereinigung zweier sequentieller Funktionen nicht 2-subsequentiell und die Vereinigung einer p -subsequentiellen mit einer q -subsequentiellen keine $(p + q)$ -subsequentielle Funktion sein. Für links-sequentielle Funktionen ist die Vereinigung, falls sie eine partielle Funktion darstellt, erneut links-sequentuell. Mit einem Gegenbeispiel widerlege ich die von Mohri in [15] angegebene Charakterisierung der sequentiellen Funktionen und beweise einen Satz, der die sequentiellen unter den subsequentiellen Funktionen beschreibt. In [16] werden λ -sequentielle Transducer eingeführt. Sie bilden eine echte Erweiterung der sequentiellen Transducer. Es gibt Funktionen, deren minimaler sequentieller Transducer einen Zustand mehr besitzt als der minimale λ -sequentielle Transducer. Es wird die Konstruktion des minimalen Transducers der beiden Modelle untersucht. Daraufhin werden in diesem Zusammenhang konkrete Algorithmen verschiedener Autoren [16, 2, 8, 5] angegeben, diskutiert und auf λ - p -subsequentielle Transducer erweitert. Außerdem wird gezeigt, wie die definierten Klassen von Funktionen mengentheoretisch in Beziehung stehen.

Weiterhin wird in das Gebiet der gewichteten Automaten eingeführt. Zunächst wird ein Beispiel angegeben, welches belegt, dass die Behauptung von Mohri [15, Theorem 9, S. 283] nicht korrekt ist. Es existieren rationale Potenzreihen mit beschränkter Variation, welche nicht subsequentiell sind. Daraufhin wird eine neue Klasse von rationalen Potenzreihen definiert. Diese durch pfadneutrale gewichtete Automaten berechenbare Funktionen haben die Eigenschaft, dass sie genau dann beschränkte Variation besitzen, wenn sie subsequentiell sind. Außerdem terminiert für diese gewichteten Automaten (falls sie schlank sind) der von Mohri angegebene Determinisierungsalgorithmus [15, S. 185] genau dann, wenn die dargestellte Funktion tatsächlich subsequentiell ist, was wiederum äquivalent dazu ist, dass in dem betrachteten gewichteten Automaten die Zwillingsseigenschaft gilt. Für pfadneutrale gewichtete Automaten ist es entscheidbar, ob sie beschränkte Variation haben oder ob sie subsequentiell sind. Setzt man zusätzlich voraus, dass sie schlank sind, ist es entscheidbar, ob sie die Zwillingsseigenschaft besitzen und ob der Algorithmus von Mohri terminiert.

Inhaltsverzeichnis

1	Einleitung	7
2	Einführung in das Thema	9
2.1	Definitionen und Hintergrund	9
2.2	Subsequentielle Transducer	13
2.3	Vereinigung von Transducern und Charakterisierung der sequentiellen Funktionen	15
3	Minimalisierung sequentieller Transducer	20
3.1	Endliche Automaten	20
3.1.1	Minimalisierung	21
3.1.2	Satz von Brzozowski	21
3.2	Charakterisierung der minimalen sequentiellen Transducer	22
3.3	Ein Algorithmus zur Minimalisierung sequentieller Transducer . .	35
3.3.1	Der Präfix eines Automaten	35
3.3.2	Der Algorithmus	40
3.3.3	Komplexität und Konstruktion des minimalen λ STs bzw. STs	40
3.4	Die Vervollständigung von Mohris Algorithmus	41
3.5	Subsequentielle und p -subsequentielle Transducer	43
3.5.1	Erweiterung der Algorithmen von Mohri und Béal/ Carton	44
3.5.2	Choffruts Minimalisierungsverfahren	46
3.5.3	Der Suffix-Baum eines Baumes	47
4	Gewichtete Automaten	49
4.1	Hintergrund	49
4.2	Subsequentielle gewichtete Automaten	50
4.3	Ein Algorithmus zur Determinisierung von gewichteten Automaten	54
4.4	Pfadneutrale gewichtete Automaten	57
5	Ausblick	66
6	Abkürzungsverzeichnis	67

1 Einleitung

Diese Arbeit beinhaltet eine Untersuchung sequentieller Transducer. Das sind endliche Automaten, welche aber bei Zustandsübergängen auch Wörter eines Alphabets ausgeben können. Durch jede Transition werden Buchstaben eingelesen, aber Wörter ausgegeben. Deshalb bezeichnet man solche Transducer auch als sequentiell, da die Eingabe (Input) Buchstabe für Buchstabe gelesen wird. Natürlich kann man sich auch viele andere Transducer vorstellen. Wenn zum Beispiel Zahlen anstatt Wörter ausgegeben werden, führt das zur Definition von gewichteten Automaten. Endliche Automaten und Transducer haben einen großen Anwendungsbereich in Sprachverarbeitung, lexikalischer Analyse und natürlicher Textverarbeitung. Dabei kann es bei der konkreten Modellierung eines Transducers oder eines endlichen Automaten Millionen von Zuständen geben. Eine ganz entscheidende Aufgabe ist es deshalb, die Anzahl der Zustände, die den Transducer oder den endlichen Automaten beschreiben, auf ein Minimum zu reduzieren, wobei es auf die Komplexität der Algorithmen ankommt. Für endliche Automaten ist dieses Problem der Minimalisierung vollständig gelöst [17]. Daraufhin begann man, sich mit diesem Problem im Kontext von Transducern zu beschäftigen. Verschiedene Autoren publizierten zum Thema [13, 5, 15, 16, 2, 8].

Mit Hilfe der Arbeit von M. Mohri [15] und den Ausführungen in [3] soll im zweiten Kapitel in das Thema eingeführt werden. Dabei gebe ich Definitionen und Aussagen, die im Verlauf der Diplomarbeit wichtig sind. Die beiden Autoren stellen verschiedene Modelle von Transducern vor, verwenden allerdings gleiche Bezeichnungen. Ich zeige, wie die durch diese unterschiedlichen Transducer dargestellten Funktionen in Beziehung stehen, und gebe Beispiele, die den Leser mit der gesamten Thematik vertraut machen sollen. Mohri [15] betrachtet die Vereinigung von Transducern. Durch ein Gegenbeispiel beweise ich, dass seine angegebene Behauptung [15, S. 273, Theorem 2] falsch ist und auch ein Satz mit der Aussage, dass die Vereinigung zweier sequentieller Funktionen (falls sie eine Funktion darstellt), erneut sequentiell ist, nicht möglich ist. Allerdings zeige ich einen solchen Satz für die in [3] definierten Transducer. Weiterhin zeige ich, dass die von Mohri in [15] angegebene Charakterisierung der sequentiellen Funktionen (das sind die Funktionen, welche durch sequentielle Transducer berechnet werden, vgl. Definition 2.9) nicht korrekt ist. Stattdessen gebe ich einen eigenen Satz zur Beschreibung der sequentiellen Funktionen an. Der Satz lehnt an eine Ausführung in [3, S. 102] an und verwendet subsequentielle Funktionen, welche eine Erweiterung der in [15] definierten sequentiellen Funktionen sind.

In Kapitel 3 gehe ich auf die konkrete Minimalisierung von Transducern ein, welche den Input deterministisch lesen. Den ersten Teil bildet ein kurzer Abschnitt über die Minimalisierung endlicher Automaten. Der Leser kann dann im weiteren Verlauf der Arbeit Vergleiche ziehen und die Analogie zur klassischen Theorie der endlichen Automaten erkennen. Mohri gibt in [16] ein neues Modell

von Transducern an, welche ich λ -sequentielle Transducer nenne. Durch Beispiele und Aussagen beschreibe ich, wie diese mit der Menge der sequentiellen Transducer in Beziehung stehen. Der minimale Transducer zu einer Funktion ist derjenige, der die minimale Anzahl von Zuständen besitzt, um diese Funktion darzustellen. Wichtig sind bei dieser Betrachtung die Eigenschaften, welche wir für das beschreibende Modell des minimalen Transducers annehmen. So zeige ich, dass es entscheidend ist, ob der minimale Transducer λ -sequentiell ist oder sequentiell. Danach stelle ich spezielle Algorithmen verschiedener Autoren [13, 16, 2, 8, 5] zur Minimalisierung vor. Mit geeigneten Gegenbeispielen widerlege ich das Lemma 1 und das Theorem 1 [13] von Mohri, welche die Minimalisierung sequentieller Transducer beschreiben. Der Autor bezieht sich bei seinen Ausführungen auf Spezialfälle. Ich untersuche, wie man bei der Minimalisierung im allgemeinen Fall vorgehen muss, und gebe die entsprechenden Sätze an. Im Anschluss erweitere ich die Algorithmen für die Minimalisierung p -subsequentieller und λ - p -subsequentieller Transducer.

Im anschließenden Kapitel stelle ich gewichtete Automaten und damit verbundene Definitionen vor. Mohri gibt in [15, S. 283] einen Charakterisierungssatz für subsequentielle Funktionen an. Dieser besagt, dass für eine rationale Potenzreihe die Eigenschaft der beschränkten Variation mit der Aussage, dass die Potenzreihe durch einen deterministischen gewichteten Automaten dargestellt werden kann, gleichbedeutend ist. Mit einem Gegenbeispiel beweise ich, dass dieser Satz nicht korrekt ist, die Eigenschaft der beschränkten Variation ist zu schwach. Anschließend beschreibe ich eine Klasse von Funktionen, die durch bestimmte gewichtete Automaten (ich nenne diese pfadneutrale gewichtete Automaten) dargestellt werden können. Für diese Funktionen ist die Behauptung von Mohri dann richtig. Auf der Menge der Funktionen, welche durch einen pfadneutralen gewichteten Automaten berechnet werden, ist es entscheidbar, ob die Funktion subsequentiell, also deterministisch darstellbar, ist. Darüberhinaus zeige ich, dass für einen pfadneutralen und schlanken gewichteten Automaten der von Mohri angegebene Algorithmus für die Determinisierung gewichteter Automaten genau dann anhält, wenn die berechnete Funktion subsequentiell ist (vgl. Satz 4.26). Dazu äquivalent ist die Bedingung, dass der betrachtete gewichtete Automat die Zwillingseigenschaft besitzt.

Die Definitionen werden mit eigenen Beispielen unterlegt. Ich zeige, wie die unterschiedlichen Mengen von Funktionen, die durch solche gewichtete Automaten berechnet werden, mengentheoretisch in Beziehung stehen.

Den Abschluss bildet ein kurzes Kapitel mit Themen und Problemen, die ausblickend zu lösen sind. Am Ende der Arbeit ist eine Liste der verwendeten Abkürzungen zu finden.

Sätze und Beispiele, welche vollständig im gleichen Kontext wie in dieser Arbeit auch in der Literatur zu finden sind, gebe ich ohne Beweis oder weitere Ausführungen wieder.

2 Einführung in das Thema

Anhand einer Publikation von Mohri [15] und einer klassischen Arbeit [3] gebe ich in diesem Abschnitt Definitionen und Sätze über Transducer an. Mohri und andere Autoren [5, 7] betrachten sequentielle Transducer, definieren sie allerdings anders als Berstel [3], der den gleichen Begriff verwendet. Sie erhalten dadurch eine echt größere Menge von rationalen Funktionen. Im Folgenden werde ich die in [3] definierten sequentiellen Transducer *links-sequentielle* Transducer nennen. In [15] bzw. [3] werden subsequentielle Transducer betrachtet, die eine Erweiterung der sequentiellen bzw. links-sequentiellen Transducer bilden. Ich stelle fest, dass diese jedoch die gleiche Menge von Funktionen beschreiben können. Die durch bestimmte Transducer realisierten Relationen oder Funktionen bilden Teilmengen der rationalen Relationen bzw. der rationalen Funktionen. Ich untersuche, wie die definierten Mengen von Transduktionen in Beziehung stehen und gebe Beispiele an. Dabei gehe ich näher auf eine Art der Vereinigung von Transducern ein, die Mohri in [15] betrachtet. Ich definiere einen Transducer, welcher die Behauptung von Mohri (Theorem 2, S. 273) für sequentielle Transducer widerlegt, und beweise dann einen ähnlichen Satz für links-sequentielle Transducer.

Den Abschluss dieses Kapitels bildet eine Charakterisierung der sequentiellen Funktionen unter den rationalen Funktionen. Zunächst widerlege ich das Theorem 3 von Mohri [15] und gebe dann eine eigene Charakterisierung unter Verwendung der subsequentiellen Funktionen an.

2.1 Definitionen und Hintergrund

Definition 2.1. Sei M ein Monoid. Die Familie $\text{Rat}(M)$ der *rationalen Teilmengen* von M ist die kleinste Familie von Teilmengen aus M , die die folgenden drei Eigenschaften erfüllt:

1. $\emptyset \in \text{Rat}(M)$; für alle $m \in M$ gilt: $\{m\} \in \text{Rat}(M)$;
2. $A, B \in \text{Rat}(M) \Rightarrow A \cup B, AB \in \text{Rat}(M)$;
3. $A \in \text{Rat}(M) \Rightarrow A^+ = \bigcup_{n \geq 1} A^n \in \text{Rat}(M)$.

Definition 2.2. Seien Σ und Δ Alphabete. Eine rationale Teilmenge des Monoids $\Sigma^* \times \Delta^*$ heißt *rationale Relation* über Σ und Δ .

Definition 2.3. Eine *Transduktion* τ von Σ^* nach Δ^* ist eine Funktion von Σ^* in die Potenzmenge $\mathfrak{P}(\Delta^*)$ von Δ^* . Der Graph von τ ist dann die Relation

$$G = \{(f, g) \in \Sigma^* \times \Delta^* \mid g \in \tau(f)\}.$$

Ist G eine rationale Relation über Σ und Δ , so wird τ *rational* genannt.

Definition 2.4. Ein *Transducer* $T = (\Sigma, \Delta, Q, i, F, E)$ besteht aus einem Eingabe-Alphabet Σ , einem Ausgabe-Alphabet Δ , einer endlichen Menge von Zuständen Q , einem Anfangszustand $i \in Q$, einer Menge von Endzuständen $F \subseteq Q$ und einer endlichen Menge von Transitionen E mit

$$E \subseteq Q \times \Sigma^* \times \Delta^* \times Q.$$

Ich gebe jetzt Definitionen und Formulierungen an, die in der gesamten Diplomarbeit eine grundlegende Rolle spielen. Sei $T = (\Sigma, \Delta, Q, i, F, E)$ ein fester Transducer.

Transducer haben eine graphische Repräsentation, ähnlich wie endliche Automaten. Im Beispiel 2.5 ist dies dargestellt. Jeder Zustand q des Transducers wird durch einen Kreis mit dem Symbol q , jeder Endzustand durch einen Doppelkreis charakterisiert. Anfangszustände kennzeichnen einführende Pfeile. Einer Transition $e = (p, u, v, q) \in E$ wird ein mit u/v beschrifteter Pfeil von p nach q zugeordnet. Ein *Pfad* π von $q_0 \in Q$ nach $q_n \in Q$ der *Länge* n über dem Wort $x_0x_1 \cdots x_{n-1}$ ist eine Folge $((q_0, x_0, a_0, q_1), (q_1, x_1, a_1, q_2), \dots, (q_{n-1}, x_{n-1}, a_{n-1}, q_n))$ von Transitionen aus E . Die *Ausgabe* von π ist $\sigma(\pi) = a_0a_1 \cdots a_{n-1} \in \Delta^*$. Ein *erfolgreicher* Pfad führt von einem Anfangszustand zu einem Endzustand. Die Menge der Wörter aus Σ^* , über denen es einen erfolgreichen Pfad in T gibt, bilden den *Definitionsbereich* von T , $\text{Dom}(T)$. Einem Wort $w \in \text{Dom}(T)$ wird durch T die Menge der Ausgaben von erfolgreichen Pfaden über w zugeordnet. Dadurch beschreibt jeder Transducer eine Abbildung $|T| : \Sigma^* \rightarrow \mathfrak{P}(\Delta^*)$ bzw. eine Relation auf $\Sigma^* \times \Delta^*$. Zwei Transducer sind *äquivalent*, wenn sie gleiche Relationen beschreiben.

Beispiel 2.5. Der Transducer aus Abbildung 1 ist das Tupel $T_1 = (\{x, y, z\}, \{a, b, c\}, \{1, 2, 3, 4, 5\}, 1, \{3, 4, 5\}, E_1)$, dabei ist $E_1 = \{(1, x, ab, 2), (1, zzy, a, 3), (2, y, ba, 4), (3, z, c, 4), (3, z, ab, 5), (3, z, \varepsilon, 5), (4, y, a, 4), (4, xy, \varepsilon, 5), (5, \varepsilon, aa, 1)\}$.

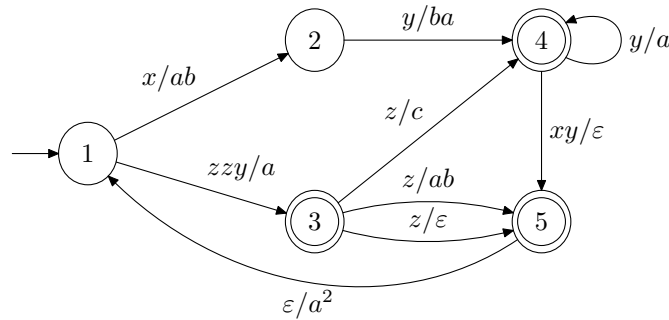


Abbildung 1: Allgemeiner Transducer T_1

Der Definitionsbereich der durch T_1 dargestellten Funktion ist die folgende Menge: $\text{Dom}(T_1) = (xy^+xy \cup z^2yzy^*xy \cup z^2yz)^*(xy^+ \cup xy^+xy \cup z^2y \cup z^2yzy^* \cup z^2yzy^*xy)$. Der

Satz 2.6 zeigt, dass die rationalen Transduktionen genau mit den Abbildungen zusammenfallen, die über Transducer definiert werden können.

Satz 2.6. (Berstel [3]) Sei $\tau : \Sigma^* \rightarrow \Delta^*$ eine Transduktion. Dann sind die beiden Aussagen äquivalent:

- (1) τ ist rational.
- (2) τ wird durch einen Transducer realisiert.

Eine rationale Transduktion $\tau : \Sigma^* \rightarrow \Delta^*$, die eine partielle Funktion von Σ^* nach Δ^* darstellt, bezeichnet man als *rationale Funktion*.

Es folgt die Definition des in der Einleitung erwähnten speziellen Transducers, der die Eingabe sequentiell liest, nur Endzustände besitzt und dessen Zustandsübergänge deterministisch sind. In [3] wird dafür der Begriff „sequentieller Transducer“ verwendet.

Definition 2.7. Ein *links-sequentieller Transducer* (kurz LST) ist ein Transducer $L = (\Sigma, \Delta, Q, i, F, E)$ mit

1. $F = Q$
2. $E \subseteq Q \times \Sigma \times \Delta^* \times Q$
3. Für alle $q, q_1, q_2 \in Q, x \in \Sigma, w_1, w_2 \in \Delta^*$ mit $(q, x, w_1, q_1), (q, x, w_2, q_2) \in E$ gilt: $w_1 = w_2, q_1 = q_2$.

Für solche Transducer enthalten die Transitionen also keine Wörter aus Σ^* , sondern Buchstaben. Man kann dann die Transitionsmenge E auch durch zwei partielle Funktionen

$$\delta : Q \times \Sigma \rightarrow Q, \quad \sigma : Q \times \Sigma \rightarrow \Delta^*$$

beschreiben, wenn man für jede Transition $e = (p, x, w, q)$, $\delta(p, x) := q$ und $\sigma(p, x) := w$ setzt. Mit den Eigenschaften in der Definition 2.7 sind beide Funktionen dann korrekt definiert, haben den gleichen Definitionsbereich und werden *Zustandsfunktion* bzw. *Ausgabefunktion* genannt. Sie werden auf $Q \times \Sigma^*$ fortgesetzt, indem man $\delta(p, w) := q$ und $\sigma(p, w) := v$ setzt, falls es in dem betrachteten Transducer einen Pfad von p nach q über w mit Ausgabe v gibt. Ein LST kann nun ebenso durch ein 7-Tupel $(\Sigma, \Delta, Q, i, F, \delta, \sigma)$ dargestellt werden. Man beachte, dass alle Zustände Endzustände bilden, so dass u. a. das leere Wort ε immer auf ε selber abgebildet wird. Außerdem ist jeder LST präfix-abgeschlossen, das heißt, falls ein Wort erkannt wird, so wird auch jeder Präfix dieses Wortes erkannt. Wenn genau fest steht, dass es sich bei einem betrachteten Transducer um einen LST handelt, müssen die Endzustände in der graphischen Darstellung nicht extra gekennzeichnet werden. Ebenso muss F in der Tupel-Schreibweise

nicht angegeben werden. Die Ausgabe über einem Pfad mit zugehörigem Eingabewort w (bezeichnet mit $\sigma(w)$, Pfade sind eindeutig festgelegt) fällt hier also mit $|L|(w)$ zusammen, da alle Pfade erfolgreich sind. Solche, durch LSTs beschriebene, *links-sequentielle Funktionen* stellen also rationale Funktionen dar.

Beispiel 2.8. Der Transducer $T_2 = (\{x, y, z\}, \{a, b\}, \{1, 2, 3\}, 1, \delta_2, \sigma_2)$ aus Abbildung 2 beschreibt eine links-sequentielle Funktion.

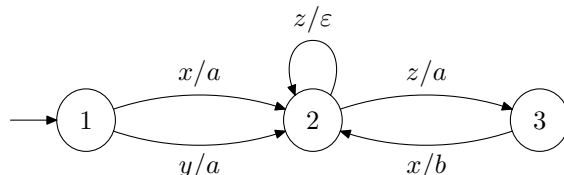


Abbildung 2: Links-sequentiieller Transducer T_2

Betrachtet man in der Definition links-sequentiieller Transducer auch Zustände, die nicht final sind, gelangt man zu *sequentiellen Transducern*, welche *sequentielle Funktionen* beschreiben. In [13, 15, 5] werden diese Transducer verwendet. Es folgt die genaue Definition.

Definition 2.9. Ein *sequentieller Transducer* (kurz ST) $T = (\Sigma, \Delta, Q, i, F, E)$ ist ein Transducer mit

1. $E \subseteq Q \times \Sigma \times \Delta^* \times Q$
2. Für alle $q, q_1, q_2 \in Q, x \in \Sigma, w_1, w_2 \in \Delta^*$ mit $(q, x, w_1, q_1), (q, x, w_2, q_2) \in E$ gilt: $w_1 = w_2, q_1 = q_2$.

Analog zu den Ausführungen, die der Definition 2.7 folgen, können δ und σ zur Beschreibung von E eingeführt werden.

Historisch betrachtete man zunächst links-sequentielle Transducer [3, 10] und führte später nicht-finale Zustände ein [13, 15, 5, 7]. Für beide Modelle wurden die gleichen Begriffe verwendet. Mit Beispiel 2.10 zeige ich kurz, dass mit der Definition der sequentiellen Transducer eine echt größere Menge von rationalen Funktionen beschrieben wird.

Beispiel 2.10. Setzt man in Beispiel 2.5 $F := \{4, 5\}$ und schränkt den Definitionsbereich auf $(xy^+ \cup xy^+xy)$ ein, so erhält man einen ST T_3 , der eine sequentielle Funktion f_3 beschreibt und ε nicht auf ε abbildet:

$$f_3 = |T_3| : \begin{cases} xy^n \mapsto ab^2a^n, & n \geq 1 \\ xy^nxy \mapsto ab^2a^n, & n \geq 1. \end{cases}$$

Deshalb kann f_3 nicht links-sequentiell sein. Außerdem ist f_3 nicht präfix-abgeschlossen. Abbildung 3 gibt die graphische Darstellung von T_3 .

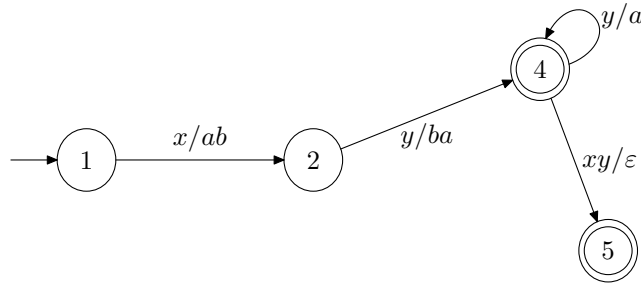


Abbildung 3: Sequentieller Transducer T_3

2.2 Subsequentielle Transducer

Ich gebe nun die Definitionen für subsequentielle und p -subsequentielle Transducer an. In der Literatur tritt der Begriff der subsequentiellen Transducer zunächst als Erweiterung der links-sequentiellen Transducer [3, 8] und später ausgehend von sequentiellen Transducern auf [13, 15]. Wir haben aber gesehen, dass LSTs und STs unterschiedliche Mengen von Funktionen beschreiben. Deshalb bezeichne ich in der folgenden Definition die Erweiterung der LSTs zunächst als *links-subsequentielle* Transducer.

Definition 2.11. Ein *links-subsequentieller* (bzw. *subsequentieller*) Transducer, kurz LSST (bzw. SST), $S = (\Sigma, \Delta, Q, i, F, \delta, \sigma, \rho)$ setzt sich aus einem LST (bzw. ST) $(\Sigma, \Delta, Q, i, F, \delta, \sigma)$ sowie einer partiellen Funktion $\rho : F \rightarrow \Delta^*$ (der *Endausgabe-Funktion*) zusammen. Die *links-subsequentielle* (bzw. *subsequentielle*) Funktion $|S| : \Sigma^* \rightarrow \Delta^*$, die durch S berechnet wird, ist definiert durch:

$$|S|(w) = \begin{cases} \sigma(i, w)\rho(\delta(i, w)), & \text{falls } \rho(\delta(i, w)) \text{ und } \sigma(i, w) \text{ definiert sind} \\ \perp, & \text{sonst.} \end{cases}$$

Für Ausgaben $\sigma(\pi)$ über Pfade π wird die Endausgabe-Funktion ρ nicht berücksichtigt, sie fallen, falls π ein Pfad über $w \in \text{Dom}(S)$ ist, nicht zwangsläufig mit $|S|(w)$ zusammen.

Bemerkung 2.12. Im Weiteren ist eine Funktion, die sich aus Komposition anderer Funktionen ergibt, genau dann definiert, wenn alle Komponenten definiert sind.

Da ρ eine partielle Funktion ist, werden so für links-sequentielle Transducer indirekt doch nicht-finale Zustände eingeführt. Dass die Menge der subsequentiellen mit der Menge der links-subsequentiellen Funktionen zusammenfällt, kann ich nun leicht zeigen (Satz 2.13), so dass wir dann allgemein von subsequentiellen Funktionen sprechen können.

Satz 2.13. Sei $f : \Sigma^* \rightarrow \Delta^*$ eine rationale Funktion. Dann sind äquivalent:

- (1) f ist links-subsequentiell.
- (2) f ist subsequentiell.

Beweis: (1) \Rightarrow (2): Nach Voraussetzung existiert ein LSST $T = (\Sigma, \Delta, Q, i, \delta, \sigma, \rho)$ mit $f = |T|$.

Dann ist $T' := (\Sigma, \Delta, Q, i, Q, \delta, \sigma, \rho)$ ein äquivalenter subsequentieller Transducer, der f berechnet.

(2) \Leftarrow (1): Sei nun $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma, \rho)$ ein f berechnender SST. Ich setze $T' := (\Sigma, \Delta, Q, i, \delta, \sigma, \rho)$. Die Endausgabe-Funktion ρ ist partiell auf F definiert, also auch partiell auf Q . Es gilt $\text{Dom}(T) = \text{Dom}(T')$ und für $w \in \text{Dom}(T)$ folgt $|T|(w) = (\sigma(i, w))\rho(\delta(i, w)) = |T'|(w)$. \square

Eine Erweiterung der sequentiellen Transducer bilden p -subsequentielle Transducer, die von Mohri [12] eingeführt wurden. Diese ermöglichen es durch eine Endausgabe-Funktion ρ an finalen Zuständen bis zu p Endausgabewörter zu bilden.

Definition 2.14. Ein p -subsequentieller Transducer (p SST) $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma, \rho)$ setzt sich aus einem ST $(\Sigma, \Delta, Q, i, F, \delta, \sigma)$ und einer partiellen Funktion $\rho : F \rightarrow (\Delta^*)^p$, $\rho = (\rho_1, \dots, \rho_p)$, $p \in \mathbb{N}$ zusammen. Einem Eingabewort $w \in \text{Dom}(T)$, welches zu einem Endzustand führt, wird die Menge der möglichen p zugehörigen Ausgaben entlang des Pfades unter Beachtung der zusätzlichen Endausgabewörter zugeordnet:

$$|T|(w) = \{(\sigma(i, w))\rho_j(\delta(i, w)) \mid j = 1, \dots, p\}.$$

Der Fall der subsequentiellen Transducer ist für $p = 1$ enthalten. Solche p -subsequentiellen Transducer beschreiben p -subsequentielle Funktionen bzw. Relationen.

Beispiel 2.15. Subsequentielle Transducer realisieren im Vergleich zu STs eine echt größere Menge von rationalen Funktionen. Durch den subsequentiellen Transducer in Abbildung 4 wird ε auf a ($\neq \varepsilon$) abgebildet, die beschriebene Funktion kann somit durch keine sequentielle Funktion dargestellt werden.

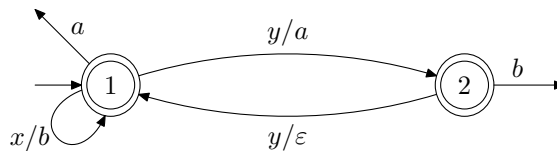


Abbildung 4: Subsequentieller Transducer

Wir könnten auch für links-sequentielle Transducer eine partielle Funktion definieren, die Zuständen bis zu p Endausgabewörter zuordnet. Die so definierten Relationen würden dann aber, analog zum Beweis im Satz 2.13, mit denen, die durch p -subsequentielle Transducer definiert werden, zusammenfallen.

Sequentielle Funktionen sind auch subsequentiell. Es gelten die beiden folgenden Sätze: (vgl. u. a. in [7, 15]).

Satz 2.16. (Berstel [3]) *Jede subsequentielle Funktion ist rational.*

Satz 2.17. (Berstel [3], Mohri [15]) *Seien $\alpha : \Sigma^* \rightarrow \Delta^*$ und $\beta : \Delta^* \rightarrow \Omega^*$ zwei subsequentielle (bzw. links-sequentielle, sequentielle) Funktionen. Dann ist $\beta \circ \alpha$ eine subsequentielle (bzw. links-sequentielle, sequentielle) Funktion.*

Die definierten Funktionen sind also abgeschlossen unter Komposition.

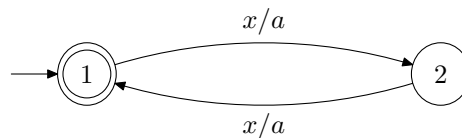
Mohri betrachtet [15, S. 273f] die Vereinigung von Transducern. Darauf soll im nächsten Abschnitt eingegangen werden.

2.3 Vereinigung von Transducern und Charakterisierung der sequentiellen Funktionen

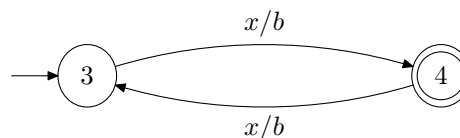
Führt man eine weitere Operation „Vereinigung“ auf der Menge der rationalen Relationen ein, welche zwei Relationen f_1 und f_2 auf diejenige Relation $f_1 + f_2$ abbildet, die jedem Eingabewort $w \in (\text{Dom}(f_1) \cup \text{Dom}(f_2))$ die Vereinigung der Anwendung von f_1 und f_2 auf w zuordnet, so ergeben sich einige interessante Eigenschaften. Zunächst stellt $f_1 + f_2$ eine rationale Relation dar und wird somit von einem Transducer realisiert. Entgegen der Behauptung von Mohri [15, Theorem 2, S. 273, ohne Beweis] muss die Vereinigung zweier sequentieller Funktionen keine 2-subsequentielle und die Vereinigung einer p -subsequentiellen und einer q -subsequentiellen Funktion keine $(p+q)$ -subsequentielle Funktion darstellen. Dazu habe ich das folgende Gegenbeispiel 2.18 gefunden. In Satz 2.19 beweise ich, dass damit die erwähnte Annahme von Mohri tatsächlich falsch ist.

Beispiel 2.18.

$$|\tau_1|(x^n) = \begin{cases} a^n, & n \text{ gerade} \\ \perp, & \text{sonst} \end{cases}$$



$$|\tau_2|(x^n) = \begin{cases} b^n, & n \text{ ungerade} \\ \perp, & \text{sonst} \end{cases}$$



Satz 2.19. *Die Vereinigung der beiden sequentiellen Funktionen $|\tau_1|$ und $|\tau_2|$ aus Beispiel 2.18 realisiert für alle $p \in \mathbb{N}$ keine p -subsequentielle Funktion, ist insbesondere also nicht 2-subsequentiell.*

Beweis: Sei $\tau := \tau_1 + \tau_2$. Es genügt zu zeigen, dass τ nicht subsequentiell ist, denn τ stellt eine Funktion dar. Wäre τ p -subsequentiell für $p \geq 2$, so auch subsequentiell. Nehmen wir also an, τ sei subsequentiell mit $\tau = |T|$ für einen subsequentiellen Transducer $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma, \rho)$, wir setzen $K := \max\{|\rho(q)| : q \in Q\}$. Dann gilt für gerade n :

$$a^n = |T|(x^n) = (\sigma(i, x^n))\rho(\delta(i, x^n))$$

$$b^{n+1} = |T|(x^{n+1}) = (\sigma(i, x^n))\sigma(\delta(i, x^n), x)\rho(\delta(i, x^{n+1})).$$

Für $n > K$ ist $w := \sigma(i, x^n) \neq \varepsilon$ und $w \in a^* \cap b^*$. Dies ist ein Widerspruch. \square

Aus diesem Beispiel ergeben sich Schlussfolgerungen, die im Gegensatz zu Mohris Artikel [15] stehen. Wenn τ_1 und τ_2 sequentiell sind, dann sind sie für $p \geq 1$ auch p -subsequentiell. Es gibt demnach stets Funktionen f, g , die p - bzw. q -subsequentiell sind, deren Vereinigung $g + f$ jedoch nicht $(p+q)$ -subsequentiell ist. Da die Funktion τ nicht subsequentiell ist, kann sie auch nicht durch einen SST dargestellt werden, so dass es somit einen möglichen Satz darüber, dass die Vereinigung $f + g$ sequentieller Funktionen f, g , falls sie eine partielle Funktion ergibt, sequentiell ist, nicht geben kann.

Den Fall, bei dem die Eigenschaft der Funktionen f, g bei der Vereinigung $f + g$ erhalten bleibt, bilden die links-sequentuellen Funktionen. Den Beweis gebe ich mit Satz 2.20. Im darauf folgenden Beispiel 2.21 habe ich die Konstruktion, die im Beweis verwendet wird, nachvollzogen.

Satz 2.20. *Seien $f, g : \Sigma^* \rightarrow \Delta^*$ zwei links-sequentuelle Funktionen derart, dass $h := f + g$ eine partielle Funktion darstellt. Dann ist auch h links-sequentuell.*

Beweis: Seien $T_j = (\Sigma, \Delta, Q_j, i_j, \delta_j, \sigma_j)$ ($j = 1, 2$) zwei LSTs, die f bzw. g realisieren. Die Funktionen δ_j und σ_j ($j = 1, 2$) seien durch Mengen von Transitionen E_j dargestellt. Ohne Einschränkung gelte $Q_1 \cap Q_2 = \emptyset$. Wir konstruieren einen neuen links-sequentuellen Transducer $T = (\Sigma, \Delta, Q, i, \delta/Q, \sigma)$, der h realisiert. Sei dazu $i := \{i_1, i_2\}$; $\delta' := \delta_1 \cup \delta_2$ und $\sigma' := \sigma_1 \cup \sigma_2$. Wir definieren:

$$\begin{aligned} \delta &: \mathfrak{P}(Q_1 \cup Q_2) \times \Sigma \rightarrow \mathfrak{P}(Q_1 \cup Q_2) \text{ mit} \\ (P, a) &\mapsto \cup_{p \in P} \delta'(p, a) \text{ falls } \cup_{p \in P} \delta'(p, a) \neq \emptyset. \end{aligned}$$

Sei nun Q die kleinste Teilmenge von $\mathfrak{P}(Q_1 \cup Q_2)$, die $\{i_1, i_2\}$ enthält, und unter Anwendung von δ abgeschlossen ist, d. h. für beliebige $\{q_1, q_2\} \in Q$ und $a \in \Sigma$ ist $\delta(\{q_1, q_2\}, a) \in Q$, falls $\delta(\{q_1, q_2\}, a)$ definiert ist.

Sei weiterhin

$$\sigma : Q \times \Sigma \rightarrow \Delta^* \text{ mit}$$

$$(P, a) \mapsto \sigma'(p, a) \text{ f\"ur ein beliebiges } p \in P.$$

Den Hintergrund bildet also die übliche Potenzmengen-Konstruktion bei der Determinisierung endlicher Automaten. Intuitiv kann man diese anwenden, da ja beide Transducer, die wir hier betrachten, als Eingabe-Automat endliche Automaten darstellen. Die gesamten Zustandsmengen sind dabei final. Da h eine Funktion darstellt, können wir die Ausgaben der alten Transitionen direkt übernehmen, die neuen Zustandsübergänge sind korrekt definiert. Wie in der Theorie der endlichen Automaten kann man nun zeigen, dass der neu definierte Transducer τ genau h realisiert. \square

Beispiel 2.21. In Abbildung 5 entsteht τ durch Vereinigung der beiden links-sequentiellen Transducer τ_1 und τ_2 . Dabei werden diese zunächst als ein Transducer (mit mehreren Initialzuständen) aufgefasst. Im Anschluss wenden wir die klassische Determinisierung endlicher Automaten an und übernehmen die Ausgaben.

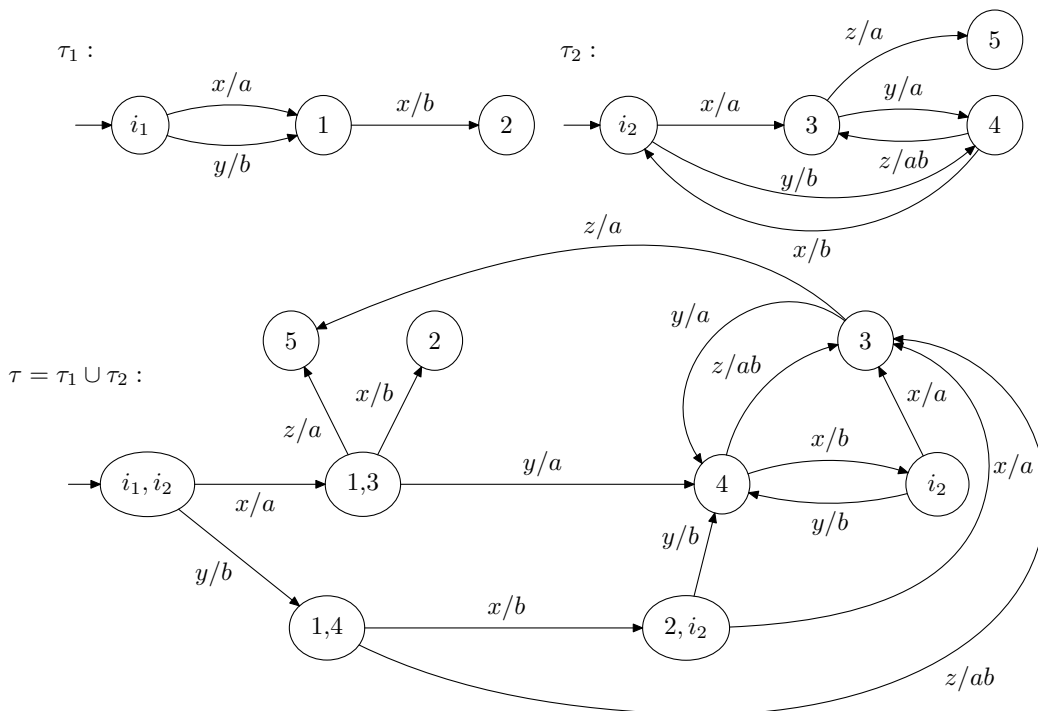


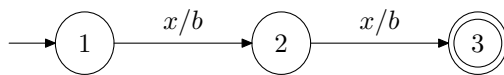
Abbildung 5: Vereinigung zweier LSTs

In [15] wird auf S. 275 auch auf eine mögliche Charakterisierung der sequentiellen Funktionen unter den rationalen Funktionen eingegangen. Es wird eine

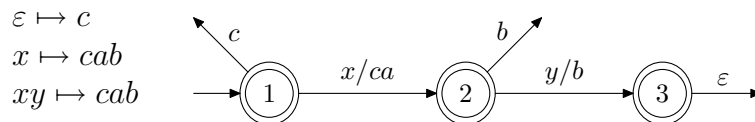
äquivalente Eigenschaft dazu gesucht, dass eine rationale Funktion sequentiell ist. Mohri bezieht sich dabei auf eine Arbeit von S. Ginsburg und G.F. Rose [10]. Diese behandelt allerdings sogenannte GSMs (generalized sequential machines), welche genau die links-sequentiellen Transducer darstellen und somit eine echt kleinere Menge von Funktionen beschreiben als die in [15] verwendeten STs. Ein Satz, welcher eine Transducer-unabhängige Bedingung dafür angibt, dass eine rationale Funktion links-sequentiell ist, kann deswegen auf die Klasse aller sequentieller Funktionen nicht angewendet werden. Die von Mohri angegebene Bedingung ist die folgende:

$$\exists K > 0 : [\forall u \in \Sigma^*, \forall a \in \Sigma, \exists w \in \Delta^* : |w| \leq K \text{ und } f(ua) = f(u)w]. \quad (1)$$

Ich gebe ein Beispiel. Für den sequentiellen Transducer τ_3 :



soll gelten $|\tau_3|(xx) = |\tau_3|(x)w$ für ein w aus $\{b\}^*$, aber $|\tau_3|(x)$ ist undefiniert, da Zustand 2 nicht final ist. Der angegebene Transducer τ_3 ist sequentiell, es gilt jedoch die zugehörige äquivalente Aussage (1) nicht. Umgekehrt können wir aber auch eine Funktion angeben, welche zwar die Charakterisierung (1) erfüllt und rational ist, jedoch nicht sequentiell. Ein Beispiel dafür ist der folgende Transducer τ_4 :



Dieser ist subsequentiell (also auch rational), aber nicht sequentiell, da gilt: $|\tau_4|(\varepsilon) = c (\neq \varepsilon)$. Andererseits erfüllt τ_4 die Bedingung (1), zum Beispiel für $K := 5$.

Ein Charakterisierungssatz für sequentielle Funktionen sollte also in der äquivalenten Bedingung nicht den Funktionswert eines Präfixes enthalten (da dieser nicht existieren muss), sondern eine Aussage über die Ausgabe eines entsprechenden Pfades für dieses Teilwort verwenden. In [3] wird eine Charakterisierung der links-sequentiellen Funktionen unter allen partiellen Funktionen angegeben, welche auch subsequentielle Funktionen verwendet. Diese Idee gibt Anlass zu Satz 2.26. Dazu führe ich zunächst den Begriff *präfix-erhaltend* für subsequentielle Funktionen ein.

Definition 2.22. Eine subsequentielle Funktion f heißt *präfix-erhaltend*, falls es einen subsequentiellen Transducer τ gibt, der f berechnet und folgende zwei Eigenschaften erfüllt sind:

1. $|\tau|(\varepsilon) = \varepsilon$ oder $|\tau|(\varepsilon) = \perp$ und
2. für alle $w, ww' \in \text{Dom}(\tau)$ gilt: $\sigma(ww') \in |\tau|(w)\Delta^*$.

Dabei ist $\sigma(w)$, wie oben bereits erwähnt, die Ausgabe über dem Pfad für w unter τ ohne Berücksichtigung der Endausgabe ρ . Für ein Wort $w \in \text{Dom}(\tau)$ fällt also $\sigma(w)$ nicht immer mit $|\tau|(w)$ zusammen. Auf den Satz 2.25 wird ausführlich im Abschnitt 3.5 mit Lemma 3.30 und Satz 3.31 eingegangen. Ich verwende ihn hier im Beweis von Satz 2.26. Zuvor führe ich den Begriff *schlank* (trim in [3, 15, u. a.]) für Transducer ein, er ist grundlegend für die nächsten Abschnitte.

Definition 2.23. Ein Transducer τ heißt *schlank*, wenn durch jeden Zustand von τ ein erfolgreicher Pfad verläuft.

Definition 2.24. Ein Transducer (bzw. ST, SST) τ ist ein *minimaler* Transducer (bzw. ST, SST) für $|\tau|$, falls jeder zu τ äquivalente Transducer (bzw. ST, SST) mindestens so viele Zustände wie τ besitzt.

Satz 2.25. Zu jedem subsequentiellen Transducer τ existiert ein äquivalenter minimaler SST τ' , für den insbesondere gilt, dass der größte gemeinsame Präfix von Ausgaben über erfolgreichen Pfaden, die von ein und demselben Zustand q (q ist kein Initialzustand von τ') starten, gleich ε ist.

Satz 2.26. Sei $f : \Sigma^* \rightarrow \Delta^*$ eine rationale Funktion. Dann sind die beiden folgenden Aussagen äquivalent:

- (1) f ist sequentiell.
- (2) f ist subsequentiell und präfix-erhaltend.

Beweis: (1) \Rightarrow (2): Diese Folgerung ist klar, da jede sequentielle Funktion auch subsequentiell ist und es einen f berechnenden ST gibt, welcher präfix-erhaltend ist, denn es gibt keine zusätzliche Endausgabe ρ .

(2) \Leftarrow (1): Der subsequentielle Transducer $\tau = (\Sigma, \Delta, Q, i, F, \delta, \sigma, \rho)$ realisiere f . Wir untersuchen die Zustandsmenge Q . Ohne Einschränkung kann man annehmen, dass ρ auf F total definiert ist. Außerdem nehmen wir mit Satz 2.25 an, dass τ minimal ist und die dort angegebene Präfixbedingung erfüllt.

Sei nun $q \in F$, $q \neq i$, $\delta(i, w) = q$, $w' \in \Sigma^*$ mit $ww' \in \text{Dom}(f)$. Dann gilt:

$$\begin{aligned}
 f(ww') &= \sigma(i, w)\sigma(q, w')\rho(\delta(q, w')) \\
 &= f(w)g = \sigma(i, w)\rho(q)g, \quad g \in \Delta^* \text{ (} f \text{ präfix-erhaltend)} \\
 \implies \rho(q)g &= \sigma(q, w')\rho(\delta(q, w'))
 \end{aligned}$$

Die Ausgabe jedes erfolgreichen Pfades von q beginnt also mit $\rho(q)$. Da wegen der Präfixbedingung der größte gemeinsame Präfix der Ausgaben über diesen Pfaden ε ist, folgt $\rho(q) = \varepsilon$. Wenn $q = i$ gilt, dann ist $\rho(q)$ ebenfalls ε , da f präfix-erhaltend ist. Für alle Zustände aus F ist also $\rho(q) = \varepsilon$. Das heißt aber, dass $\tau' := (\Sigma, \Delta, Q, i, F, \delta, \sigma)$ einen ST darstellt, der f berechnet. \square

3 Minimalisierung sequentieller Transducer

Bei der Minimalisierung von Transducern oder Automaten geht es um das Problem, zu einem konkret gegebenen Transducer bzw. Automaten einen äquivalenten zu finden, der eine Minimalitätsbedingung erfüllt. Gewöhnlich betrachtet man die Anzahl der Zustände. Oft sind dann bei einer angegebenen Konstruktion aber auch andere Parameter des Transducers oder Automaten, wie zum Beispiel die Anzahl der Transitionen, minimal. Dabei spielt die Komplexität der Algorithmen, die man verwendet, eine entscheidende Rolle. Im klassischen Fall der endlichen Automaten ist das Problem der Minimalisierung gelöst: Zu jedem deterministischen endlichen Automaten (DEA) kann man einen äquivalenten Minimalautomaten angeben [1, 11, 17, u. a.]. Der Algorithmus mit der bisher kleinsten Komplexität ist von Hopcroft [11].

Erstaunlich ist die folgende Betrachtung: Transformiert man einen gegebenen DEA A in den (nichtdeterministischen) Umkehr-Automaten, diesen in einen DEA durch die übliche Teilmengen-Konstruktion (nur wirklich erreichbare Zustände werden konstruiert), konstruiert daraus erneut zunächst den Umkehr-Automaten und daraufhin den äquivalenten DEA, dann ist der resultierende Automat ein minimaler Automat für A . Im ersten Teil dieses Kapitels wollen wir grob die Resultate der Minimalisierung endlicher Automaten zusammenfassen.

Choffrut gibt in seiner Dissertation die entscheidenden Schritte für die Konstruktion des minimalen sequentiellen Transducers zu einem gegebenen ST an. Anlehnend an die Theorie endlicher Automaten, fasst er gewisse äquivalente Zustände zusammen. Mohri geht in [13, 16] erstmals in diesem Kontext auf Algorithmen und ihre Komplexität ein. Im zweiten Teil wird die Grundidee der Minimalisierung sequentieller Transducer vorgestellt und diskutiert. Ich folge dabei den beiden Arbeiten [13, 16], gebe Beispiele und gehe auf Annahmen und Behauptungen ein, welche nicht korrekt sind. Verschiedene Autoren [13, 16, 5, 8, 2] geben Algorithmen für einige Schritte dieser Minimalisierung an, sie werden dargestellt und im Kontext der bisherigen Ergebnisse untersucht. Ich erweitere diese Algorithmen für die Minimalisierung der p -subsequentiellen Transducer.

3.1 Endliche Automaten

In diesem Abschnitt verwende ich zwei zunächst unterschiedliche Automatenmodelle: DEA und NEA. Sie akzeptieren genau die gleiche Familie von regulären Sprachen. Zu jedem solchen Automaten gibt es einen bis auf Isomorphie eindeutigen äquivalenten DEA mit minimaler Zustandsmenge. Genaue Definitionen, Beweise und weitere Sätze sind u. a. in [1, 11, 17] nachzulesen.

Darauf folgend wird auf den schon in der Einleitung dieses Kapitels erwähnten Satz über den Umkehr-Automaten und die daraus resultierende Konstruktion des Minimalautomaten eingegangen. Diese Idee wurde erstmals durch J. A. Brzozow-

ski [6] gegeben, deshalb habe ich den Satz nach ihm benannt.

3.1.1 Minimalisierung

Ein DEA A wird durch ein 5-Tupel $A = (Q, \Sigma, i, F, \delta)$ beschrieben. Falls die Zustandsübergangsfunktion $\delta : Q \times \Sigma \rightarrow Q$ total definiert ist, so heißt A *vollständig*. Bildet δ auf die Potenzmenge von Q ab: $\delta : Q \times \Sigma \rightarrow \mathfrak{P}(Q)$, so gelangt man zu einem NEA. Die akzeptierte Sprache eines Automaten A wird mit $L(A)$ bezeichnet.

Satz 3.1. (Yu [17]) *Zu jedem NEA A existiert ein vollständiger DEA A' mit $L(A) = L(A')$. (Potenzmengenkonstruktion)*

Die Sprachen, welche durch DEAs dargestellt werden können, bilden die regulären Sprachen. Um nun zu dem minimalen Automaten eines DEA A mit $L(A) = L$ zu gelangen, wird für $L \subseteq \Sigma^*$ eine Äquivalenzrelation \equiv_L auf $\Sigma^* \times \Sigma^*$ eingeführt:

$$\begin{aligned} x \in \Sigma^* : D_x L &:= \{y \in \Sigma^* \mid xy \in L\} \\ x \equiv_L y &\Leftrightarrow D_x L = D_y L. \end{aligned}$$

Entweder man betrachtet vollständige DEAs und kann Satz 3.2 beweisen, oder man schränkt \equiv_L auf die Menge der Präfixe von erkennbaren Wörtern in A ein:

$$\begin{aligned} x, y \in D(A) \times D(A) : x \equiv^L y &\Leftrightarrow D_x L = D_y L \text{ mit} \\ D(A) &= \{u \in \Sigma^* \mid \exists w \in \Sigma^* : uw \in \text{Dom}(A)\} \end{aligned} \quad .$$

Bei der Konstruktion des Minimalautomaten fasst man solche Zustände zusammen, welche durch äquivalente Wörter erreichbar sind.

Satz 3.2. (Yu [17]) *Die minimale Anzahl von Zuständen eines vollständigen DEA, der L akzeptiert, ist gleich dem Index von \equiv_L .*

Satz 3.3. (Yu [17] bez. weitere Überlegungen) *Die minimale Anzahl von Zuständen eines DEA, der L akzeptiert, ist gleich dem Index von \equiv^L .*

Satz 3.4. (Hopcroft [11]) *Die Komplexität, um zu einem gegebenen DEA mit n Zuständen den äquivalenten minimalen DEA zu konstruieren, ist $\mathcal{O}(n \log n)$.*

3.1.2 Satz von Brzozowski

Hier soll kurz ein anderer Weg zur Konstruktion des minimalen DEA zu einem gegebenen NEA vorgestellt werden [6, 17].

Ein NEA mit *nichtdeterministischer Initialzustandsmenge* (NNEA) ist durch ein 5-Tupel $(Q, \Sigma, I, F, \delta)$ gegeben. Es kann eine Menge von Anfangszuständen geben. Dadurch wird ebenfalls die Klasse der regulären Sprachen definiert, denn zu jedem NNEA gibt es einen äquivalenten DEA.

Definition 3.5. Sei $A = (Q, \Sigma, i, F, \delta)$ ein deterministischer endlicher Automat. Der *Umkehr-Automat* $A^R = (Q, \Sigma, F, \{i\}, \delta^R)$ von A ist der NNEA mit der Funktion $\delta^R : Q \rightarrow \mathfrak{P}(Q)$, $\delta^R(p, a) := \{q \in Q \mid \delta(q, a) = p\}$, der durch Änderung der Pfeilrichtung entsteht. Mit $\tau(A) = (Q', \Sigma, i', F', \delta')$ bezeichnen wir den DEA mit $L(\tau(A)) = L(A)$, welcher durch die Potenzmengenkonstruktion gegeben ist. Dabei soll $Q' \subseteq \mathfrak{P}(Q)$ die Menge der Zustände sein, die von i' erreichbar sind.

Satz 3.6. (Brzozowski [6], Yu [17]) Sei A ein NEA. Der DEA A' entstehe aus A wie folgt:

1. Konstruktion des Umkehr-Automaten : A^R
2. Determinisierung : $\tau(A^R)$
3. Konstruktion des Umkehr-Automaten : $(\tau(A^R))^R$
4. Determinisierung : $\tau(\tau(A^R))^R =: A'$

Dann ist A' der zu A äquivalente minimale DEA.

Der Beweis zu diesem Satz ist ausführlich in [17, S. 95] angegeben.

3.2 Charakterisierung der minimalen sequentiellen Transducer

In diesem Abschnitt möchte ich deutlich machen, dass die in [13] und [16] unter ein und dem selben Begriff definierten Transducer sehr verschiedene Modelle sind, welche im Zusammenhang mit der Frage nach dem minimalen Transducer zu unterschiedlichen Ergebnissen führen. Zum Beispiel gibt es sequentielle Funktionen, deren minimaler Transducer in dem ersten Modell einen Zustand mehr besitzt als der zugehörige minimale Transducer des zweiten Modells.

In der Einführung wurden zwei unterschiedliche Transducer, die den Input Buchstabe für Buchstabe einlesen, definiert. Nun betrachte ich die sequentiellen Transducer. Genau diese Klasse von Funktionen bildet die Grundlage in [13, 2, 8]. Mohri [16] erweitert sie, indem er ein zusätzliches Wort von links an jede Ausgabe konkateniert. Im Folgenden werde ich solche Transducer *λ -sequentiell* nennen und zeigen, dass sie eine echte Erweiterung der bisherigen sequentiellen Transducer darstellen. Es wird die Charakterisierung und die Konstruktion der minimalen sequentiellen bzw. λ -sequentiellen Transducer untersucht. Dabei folge ich Mohris Ausführungen und widerlege sowohl Lemma 1 als auch Theorem 1 in [13]. Mohri nimmt bei dem Problem der Minimalisierung an, dass der größte gemeinsame Präfix von Ausgaben über Pfade, welche im Initialzustand starten und zu finalen Zuständen führen, gleich ε ist. Er gibt die Begründung, dass dies keine Beschränkung der Allgemeinheit darstelle. Mit Beispielen zeige ich, dass es

mit dieser Begründung jedoch eine Beschränkung ist, und untersuche, wie man bei der Minimalisierung im allgemeinen Fall korrekt vorgehen muss. Ich gebe an, wie man den minimalen λ ST zu einem gegebenen λ ST berechnet. Außerdem zeige ich Schritte zur Konstruktion des minimalen STs zu einem sequentiellen Transducer oder einem λ ST, welcher eine sequentielle Funktion berechnet. Dabei forme ich einen gegebenen λ -sequentiellen Transducer in den äquivalenten minimalen λ ST und darauf folgend in einen ST mit minimaler Anzahl von Zuständen um.

Ferner stelle ich fest, dass es für sequentielle Funktionen bisher keine definierte Äquivalenzrelation (die nur von der Funktion selber abhängt) gibt, deren Index, analog der klassischen Theorie der endlichen Automaten, mit der Anzahl der Zustände im minimalen sequentiellen Transducer zusammenfällt. Ich gebe zwei Beispiele an, die beweisen, dass die durch Mohri definierten Relationen die eben erwähnte Eigenschaft nicht erfüllen.

Definition 3.7. Ein λ -sequentieller Transducer (λ ST) $T = (\Sigma, \Delta, Q, i, F, \lambda, \delta, \sigma)$ setzt sich aus einem sequentiellen Transducer $(\Sigma, \Delta, Q, i, F, \delta, \sigma)$ und einem Wort $\lambda \in \Delta^*$ zusammen, welches von links an jede Ausgabe konkateniert wird. Die partielle Funktion $|T| : \Sigma^* \rightarrow \Delta^*$, die durch T beschrieben wird (λ -sequentielle Funktion), ist definiert durch:

$$|T|(w) = \lambda(\sigma(i, w)).$$

Mohri verwendet für diese Definition erneut den Begriff „sequentieller Transducer“. Allerdings wird dadurch die Klasse der so beschriebenen Funktionen echt vergrößert. Ich zeige dies in Lemma 3.8.

Lemma 3.8. *Die Menge der sequentiellen Funktionen ist eine echte Teilmenge der λ -sequentuellen Funktionen. Sei f λ -sequentiell und $T = (\Sigma, \Delta, Q, i, F, \lambda, \delta, \sigma)$ ein beliebiger λ ST, der f berechnet. Dann sind die beiden folgenden Aussagen äquivalent:*

- (1) *Die Funktion f ist sequentiell.*
- (2) *Der Zustand i ist nicht in F oder $\lambda = \varepsilon$.*

Beweis: (2) \Rightarrow (1): Falls $\lambda = \varepsilon$, so ist f trivialerweise sequentiell. Für den Fall, dass der Initialzustand kein Endzustand ist, können wir durch eine einfache Konstruktion einen äquivalenten ST definieren. Dazu führen wir einen neuen Initialzustand j ein, alle Transitionen, welche in T von i starteten, zeigen jetzt auf die gleichen Zustände, jedoch von j aus. Dabei wird λ auf diese Transitionen von links an die Ausgabe konkateniert (Es entsteht eine neue Transitionsmenge, die durch δ' und σ' beschrieben wird.). Der gesuchte äquivalente ST ist dann $T' = (\Sigma, \Delta, Q \cup \{j\}, j, F, \delta', \sigma')$, er berechnet f . Beispiel 3.9 zeigt diese Konstruktion.

(1) \Rightarrow (2): Wenn umgekehrt f sequentiell ist, so muss entweder $\lambda = \varepsilon$ sein, oder i ist kein finaler Zustand von T , denn andernfalls wäre $\varepsilon \in \text{Dom}(f)$ und

$f(\varepsilon) \neq \varepsilon$. Alle sequentiellen Transducer können aber das leere Wort nur wieder auf das leere Wort abbilden. \square

Beispiel 3.9. Sei $T = (\{a, b, c, d\}, \{A, B, C\}, \{i, 2, 3, 4, 5, 6\}, i, \{4, 5\}, BC, \delta, \sigma)$. Dann ist $T' = (\{a, b, c, d\}, \{A, B, C\}, \{i, 2, 3, 4, 5, 6, j\}, j, \{4, 5\}, \delta', \sigma')$ ein äquivalenter ST. Abbildung 6 zeigt die graphischen Darstellungen.

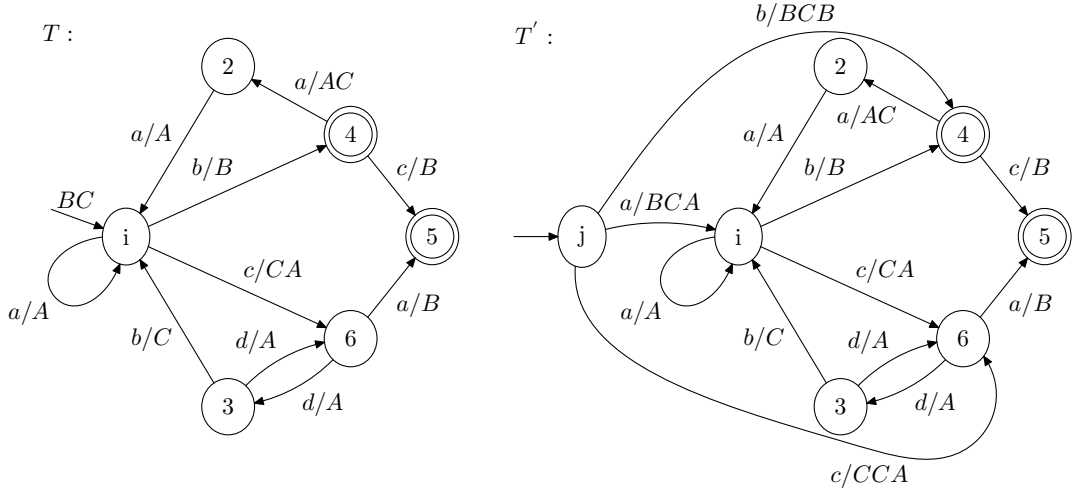


Abbildung 6: Konstruktion des STs T' aus einem äquivalenten λ ST T

Für die Beschreibung der minimalen Transducer unter den STs wird für eine sequentielle Funktion f eine Äquivalenzrelation R^f auf Σ^* definiert [13].

$$\forall u, v \in \Sigma^*, uR^fv \iff \begin{cases} \exists u', v' \in \Delta^* : \\ \forall w \in \Sigma^* : uw \in \text{Dom}(f) \iff vw \in \text{Dom}(f), \\ uw \in \text{Dom}(f) \Rightarrow u'^{-1}f(uw) = v'^{-1}f(vw). \end{cases}$$

Um nun anlehnd an die Theorie der endlichen Automaten die Eigenschaft zu erhalten, dass die Anzahl der Zustände des minimalen Transducers für f mit dem Index von R^f zusammenfällt, müsste die Relation zunächst anders definiert werden. Ich gebe ein Beispiel.

Beispiel 3.10. Gegeben sei der ST $T = (\{a, b\}, \{A, B\}, \{1, 2, 3\}, 1, \delta, \sigma)$. Die Äquivalenzklassen bezüglich R^f sind $\Sigma^*_{/R^f} = \{\bar{a}, \bar{\varepsilon}, \bar{b}\} = \{\{a\}, \{\varepsilon\}, \Sigma^* \setminus \{a, \varepsilon\}\}$. Der minimale Transducer T_{min} zu T besteht aber aus zwei Zuständen. Die Abbildung 7 zeigt die graphischen Darstellungen.

Man kann nun wie Mohri annehmen, der Transducer sei schlank. Dann würde zwar der Zustand 3 in Abbildung 7 wegfallen und der Transducer wäre minimal. Aber $\Sigma^*_{/R^f}$ besteht noch immer aus 3 Elementen. Das ist ja auch ganz natürlich,

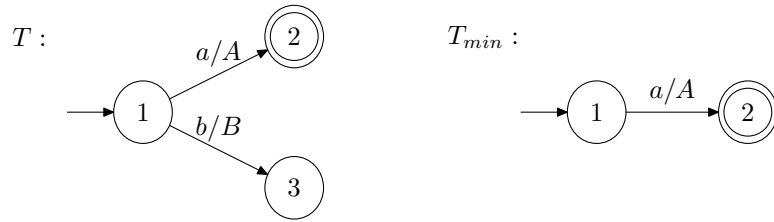


Abbildung 7: Die Relation R^f muss größer definiert werden.

denn die Definition der Äquivalenzrelation ist unabhängig von dem betrachteten Transducer. Im Folgenden wird obige Relation für allgemeine λ STs auf der Menge der Präfixe von erkennbaren Wörtern definiert.

Definition 3.11. Sei f eine λ -sequentielle Funktion über Σ . Dann ist R_f die obige Relation R^f eingeschränkt auf die Menge

$$D(f) = \{u \in \Sigma^* : \exists w \in \Sigma^*, uw \in \text{Dom}(f)\}.$$

Die definierte Relation ist eine Äquivalenzrelation. Wenn man R^f betrachtet, gibt es eine zusätzliche Äquivalenzklasse für alle Wörter, welche nicht „gebraucht“ werden, also nicht in $D(f)$ liegen. Der Index von R^f fällt nicht zwingend mit der Anzahl der Zustände im minimalen ST zusammen. Der in Beispiel 3.10 angegebenen Transducer widerlegt das Lemma 1 sowie das Theorem 1 [13, S. 158]. Mohri betrachtet im Beweis seiner Behauptungen die zusätzliche Klasse der Wörter $w \notin D(f)$ nicht, führt aber dafür einen unnötigen Zustand ein.

In [16] wird die Relation auf $D(f)$ definiert, jedoch für λ -sequentielle Funktionen. Mit Lemma 3.8 habe ich zwar gezeigt, dass diese die STs als Teilmenge enthalten, bei der Konstruktion im Beweis wurde jedoch ein neuer Zustand eingeführt. Jetzt soll untersucht werden, inwiefern dieser neu eingeführte Zustand wirklich notwendig ist. Tatsächlich gibt es sequentielle Funktionen, für die der minimale sequentielle Transducer mehr Zustände als der minimale λ -sequentielle Transducer besitzt. Nach der Konstruktion im Lemma 3.8 kann die Differenz dann höchstens eins sein. Im Beispiel 3.12 gebe ich eine solche Funktion an und es gilt somit Lemma 3.13. Ich folgere, dass es einen Satz, der besagt, dass für STs die Anzahl der Zustände im minimalen ST mit dem Index einer dieser Äquivalenzrelationen, R^f oder R_f , zusammenfällt, nicht geben kann.

Beispiel 3.12. Abbildung 8 stellt einen minimalen ST T_1 dar, der f berechnet. Der minimale λ ST für f , T_2 , hat jedoch nur 2 Zustände.

Lemma 3.13. *Es gibt eine sequentielle Funktion f , deren minimaler λ ST einen Zustand weniger besitzt als der minimale ST zu f .*

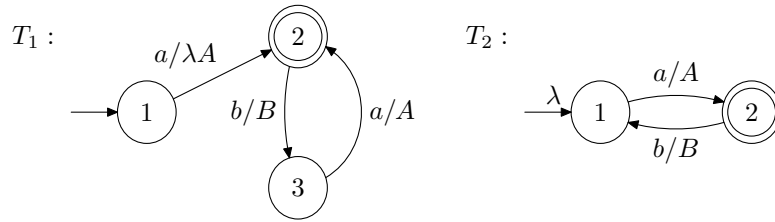


Abbildung 8: f als minimaler ST bzw. minimaler λ -ST

Korollar 3.14. *Es gibt eine sequentielle Funktion f , so dass die Anzahl der Zustände des minimalen sequentiellen Transducers zu f nicht mit dem Index von R^f oder R_f zusammen fällt*

Beweis: Die Beispiele 3.10 und 3.12 zeigen die Behauptung. \square

Da die Äquivalenzrelation für eine sequentielle Funktion f , jetzt auf der Menge $D(f)$ definiert, unabhängig von dem Transducer für f ist, können einige Sätze und Lemmata aus [16] übertragen werden, sie enthalten zunächst den Fall der sequentiellen Funktionen. Man könnte ebenso die Beweise Mohris [13] auf die neue Menge anwenden. Darauf soll in dieser Arbeit nicht weiter eingegangen werden. Zunächst gilt mit dem Beweis von Mohri [16, S. 191]:

Lemma 3.15. (Mohri [16]) *Für jede λ -sequentielle Funktion f ist der Index von R_f endlich. Jeder λ ST, der f darstellt, hat mindestens so viele Zustände, wie dieser Index angibt.*

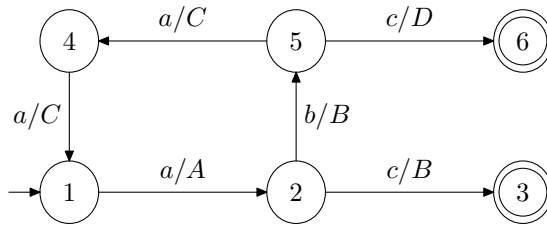
Die folgenden Definitionen und Bezeichnungen sind entscheidend für die eigentliche Konstruktion des minimalen Transducers und Satz 3.19. Dabei bezeichnet $x \wedge y$ den größten gemeinsamen Präfix zweier Wörter. Wir fixieren einen ST $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma)$ bzw. einen λ ST $T = (\Sigma, \Delta, Q, i, F, \lambda, \delta, \sigma)$. Jedem sequentiellen Transducer T wird in [13] ein Transducer (Präfix von T) zugeordnet, der die Ausgabe so früh wie möglich liefert. Dieser soll die gleiche Funktion berechnen. Zur Definition dieses Präfixes eines STs wird eine Funktion, ich nenne sie hier Pr_T , eingeführt, welche jedem Zustand von T den größten gemeinsamen Präfix der möglichen Ausgaben über alle Pfade dieses Zustandes zu Endzuständen zuordnet. Der Index T soll kennzeichnen, dass die Präfixe im Transducer T gebildet werden. Der Präfix $\text{Pr}(T)$ entsteht dann, indem man Transitionen $(q \xrightarrow{w/w'} p)$ aus T in $(q \xrightarrow{w/[\text{Pr}_T(q)]^{-1}w' \text{Pr}_T(p)} p)$ mit $q \neq i$ und Transitionen $(i \xrightarrow{w/w'} p)$ in $(i \xrightarrow{w/w' \text{Pr}_T(p)} p)$ umwandelt [13, S. 153]. Initialzustand, Zustandsmenge und finale Zustände bleiben erhalten. Der Präfix $\text{Pr}(T)$ unterscheidet sich von T nur durch die Verteilung der Ausgaben entlang der Pfade, kann aber gegebenenfalls eine andere Funktion berechnen. Die Eigenschaften in [13, 16] gelten nur mit

einigen Einschränkungen und tieferen Überlegungen, die uns jetzt beschäftigen sollen. Die Definition des Präfixes eines STs führt nicht zwangsläufig zu einem äquivalenten Automaten, die Behauptung von Mohri [13, S. 153] ist deshalb nicht korrekt. Ich zeige dies mit Beispiel 3.16. Wenn es in dem Transducer, der betrachtet wird, keine erfolgreichen Pfade gibt, die mehrmals den Initialzustand enthalten, oder $\text{Pr}_T(i)$ das leere Wort ist, führt die Definition des Präfixes eines STs in [13] zu einem äquivalenten Transducer.

Beispiel 3.16. Sei T der folgende in Abbildung 9 dargestellte ST mit der Präfixfunktion $\text{Pr}_T : Q \rightarrow \Delta^*$. Dann berechnen T und $\text{Pr}(T)$ verschiedene Funktionen, denn es gilt:

$$T(aba^3b) = ABC^2AB; \quad (\text{Pr}(T))(aba^3b) = ABC^2ABAB.$$

T :



$$\text{Pr}_T(1) = AB$$

$$\text{Pr}_T(2) = B$$

$$\text{Pr}_T(3) = \varepsilon$$

$$\text{Pr}_T(4) = CAB$$

$$\text{Pr}_T(5) = \varepsilon$$

$$\text{Pr}_T(6) = \varepsilon$$

$\text{Pr}(T)$:

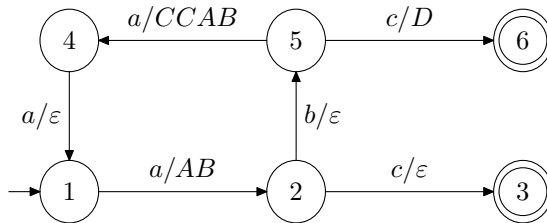


Abbildung 9: Es kann gelten: $\text{Pr}(T) \not\cong T$.

Entweder man beweist nun, dass ohne Beschränkung der Allgemeinheit angenommen werden kann, $\text{Pr}_T(i) = \varepsilon$ (i Initialzustand), oder man erweitert die Definition eines sequentiellen Transducers in der Hinsicht, dass man $\text{Pr}_T(i)$ auch „vorziehen“ kann. Das würde genau zu dem Modell des λ STs führen. An Beispiel 3.12 haben wir aber gesehen, dass dies zwei verschiedene Ansätze sind. Ebenso wenig kann man $\text{Pr}_T(i) = \varepsilon$ setzen und somit die Präfixe nur bis zu dem Initialzustand „vorziehen“. Ein Beispiel dazu gebe ich am Ende dieses Abschnitts. Zunächst möchte ich auf λ -sequentielle Transducer eingehen.

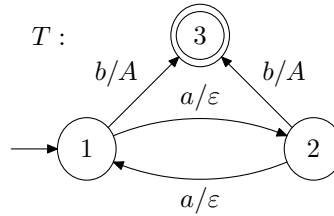
In dem überarbeiteten Artikel [16, S. 180], der nun auf der Definition der λ STs beruht, gibt Mohri den Präfix-Transducer zu einem λ ST T leicht anders an:

$P_T : Q \rightarrow \Delta^*$ sei:

$$\begin{aligned} \forall q \in F : P_T(q) &= \varepsilon \\ \forall q \in Q \setminus F : P_T(q) &= \bigwedge_{a \in \Sigma} \sigma(q, a) P_T(\delta(q, a)). \end{aligned}$$

Allerdings ist das keine korrekte Definition, sondern ein Gleichungssystem, da durch die Rekursion Mehrdeutigkeiten entstehen können. Die maximale Lösung ist die eigentlich gesuchte.

Beispiel 3.17. Betrachten wir dazu den folgenden Transducer:



Dann ist $P_T(3) = \varepsilon$. Der Wert $P_T(1)$ verwendet $P_T(2)$, aber $P_T(2)$ verwendet wiederum $P_T(1)$. Tatsächlich gibt es für P_T mehrere Lösungen:

1. $P_T(1) = P_T(2) = \varepsilon$
2. $P_T(1) = P_T(2) = A$

Wir wollen hier die (intuitive) Definition über die Transitionen verwenden. Die Funktion (Präfixfunktion) $P_T : Q \rightarrow \Delta^*$ ordnet jedem Zustand von T den größten gemeinsamen Präfix der möglichen Ausgaben über alle Pfade dieses Zustandes zu Endzuständen von T zu. Der Präfix $P(T)$ entsteht indem man Transitionen $(q \xrightarrow{w/w'} p)$ aus T in $(q \xrightarrow{w/[P_T(q)]^{-1}w'P_T(p)} p)$ umwandelt [16, S. 180]. In $P(T)$ gilt dann für alle Zustände $P_{P(T)}(q) = \varepsilon$. Mohri macht die Einschränkung, dass gilt: $P_T(i) = \varepsilon$. Seine angegebenen Theoreme 1 [16, S. 191] und 2 [16, S. 195] gelten für diesen Fall. Mit dieser Bedingung kann ich die beiden Sätze direkt auf STs übertragen, denn der minimale λ ST zu einem ST mit Initialzustand i und $P_T(i) = \varepsilon$ ist auch der minimale ST.

Im Weiteren nehme ich an, dass das Ergebnis einer Minimalisierung endlicher Automaten schlank ist. So muss bei der Minimalisierung von Transducern nicht angenommen werden, dass der Transducer schlank ist, da diese eine Minimalisierung endlicher Automaten enthält.

Satz 3.18. (Mohri [16] erweitert auf STs) Zu jeder λ -sequentiellen (bzw. sequentiellen) Funktion f , die durch einen λ ST (bzw. ST) dargestellt wird mit

$$P_T(i) = \varepsilon,$$

gibt es einen minimalen λ ST (bzw. ST) für f . Die Anzahl der Zustände ist dabei gleich dem Index von R_f .

Satz 3.19. (Mohri [16] erweitert auf STs) Sei T ein λ ST (bzw. ST) mit $P_T(i) = \varepsilon$. Der minimale λ ST (bzw. ST) zu T entsteht durch Anwendung der Minimalisierung im Sinne der klassischen Automatentheorie (die Ein- und Ausgaben der Transitionen werden zu Buchstaben eines Alphabetes konkateniert) auf den Präfix $P(T)$ [13, 16].

Mohri argumentiert wie folgt: die Definition des Präfixes ist unabhängig von λ . Falls $P_T(i) \neq \varepsilon$, ersetze λ durch $\lambda \cdot P_T(i)$. Dann soll wie im Satz 3.19 vorgegangen werden. Mit Beispiel 3.20 zeige ich, dass dies nicht zu $P(i) = \varepsilon$ führt. Es kann Kreise geben, die mehrmals den Initialzustand durchlaufen und deshalb auch $P_T(i)$ mehrmals ausgeben.

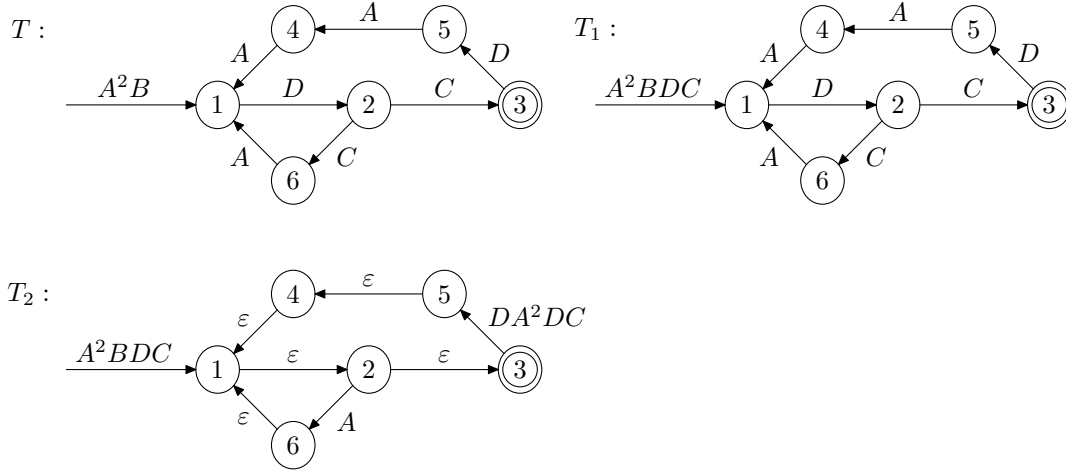


Abbildung 10: $P(i) = \varepsilon$ ist zunächst eine Beschränkung der Allgemeinheit.

Beispiel 3.20. Sei T der λ ST aus Abbildung 10 mit $|T| = f$ unter Vernachlässigung des Eingabe-Automaten. Wenn $\lambda = A^2B$ durch $\lambda \cdot P_T(i) = A^2BDC$ ersetzt wird, erhält man T_1 . Die beiden Transducer T und T_1 stellen verschiedene Funktionen dar. Der Präfix-Transducer $P(T_1)$ zu T_1 berechnet dann allerdings f . Analog dazu könnte man in einem ersten Schritt den Präfix zu T bilden, im zweiten Schritt dann λ durch $\lambda \cdot P_T(i)$ in T ersetzen. Eine Reduzierung des Problems auf $P_T(i) = \varepsilon$ ist also nur nach einer Präfix-Bildung $P(T)$ möglich.

werden als Buchstaben eines Alphabetes aufgefasst. Dann ist $T_3 = (\Sigma, \Delta, Q_2, i_2, F_2, \lambda', \delta_2, \sigma_2)$ der minimale λ ST äquivalent zu T , wobei $\lambda' := \lambda \cdot P_T(i)$ ist. Die Anzahl der Zustände in T_3 ist gleich dem Index bez. $R_{|T_3|}$.

Beweis: $T \xrightarrow{P(T), \text{ ohne } \lambda} T_1 \xrightarrow{\text{kl. Min.}} T_2 \xrightarrow{+\lambda'} T_3$

Für den ST T_1 gilt $P_{T_1}(i) = \varepsilon$ und $P(T_1) = T_1$. Der ST T_2 ist der minimale Transducer zu $|T_1|$ nach Satz 3.19. Ich betrachte ein Wort $w = a_1 a_2 \cdots a_n$ aus dem Definitionsbereich von T . Der Transducer T ist bezüglich der Eingabe deterministisch. Sei

$$(i, a_1, \sigma_1, q_1)(q_1, a_2, \sigma_2, q_2) \cdots (q_{n-1}, a_n, \sigma_n, q_n)$$

der erfolgreiche Pfad für w in T mit $q_n \in F$. Dann ist $|T|(w) = \lambda \sigma_1 \cdots \sigma_n$ und

$$(i, a_1, P_T(i)^{-1} \sigma_1 P_T(q_1), q_1)(q_1, a_2, P_T(q_1)^{-1} \sigma_2 P_T(q_2), q_2) \cdots \\ \cdots (q_{n-1}, a_n, P_T(q_{n-1})^{-1} \sigma_n P_T(q_n), q_n)$$

der erfolgreiche Pfad für w in T_1 . Es gilt also

$$|T_1|(w) = P_T(i)^{-1} \sigma_1 P_T(q_1) \cdot P_T(q_1)^{-1} \sigma_2 P_T(q_2) \cdots P_T(q_{n-1})^{-1} \sigma_n \overbrace{P_T(q_n)}^{= \varepsilon} \\ = P_T(i)^{-1} \sigma_1 \cdots \sigma_n = P_T(i)^{-1} \lambda^{-1} |T|(w).$$

Die Transducer T und T_3 stellen die gleiche Funktion dar, denn die Definitionsbereiche sind identisch. Sei also $w \in \text{Dom}(T)$ mit $|T|(w) = \lambda P_T(i) w'$. Dann gilt:

$$|T_1|(w) = w' = |T_2|(w) \\ |T_3|(w) = \lambda' |T_2|(w) = \lambda P_T(i) |T_2|(w) = \lambda P_T(i) w' = |T|(w).$$

Nach Satz 3.21 haben die Transducer T_2, T_3 die gleiche Anzahl von Äquivalenzklassen bez. der Funktion, die sie jeweils berechnen. Der Transducer T_2 ist minimal, nach Satz 3.18 also ist $\text{Index}(|T_2|) = |Q_2| = \text{Index}(|T_3|)$. Der Transducer T_3 ist ein λ ST, der genau so viele Zustände besitzt, wie der Index von $|T_3|$ angibt. Lemma 3.15 zeigt die Behauptung. \square

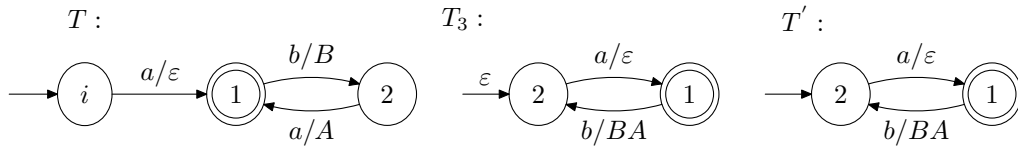
Das eigentliche Problem der Minimalisierung eines λ -sequentiellen Transducers ist also tatsächlich auf die Minimalisierung mit $\lambda = P_T(i) = \varepsilon$ reduzierbar. Eine Präfix-Bildung mit gegebenenfalls $P_T(i) \neq \varepsilon$ ist allerdings nicht zu umgehen. Deshalb ist dann bei Angabe des allgemeinen Algorithmus Satz 3.22 wesentlich. Dieser ist der gesuchte analoge Satz zu den Sätzen 3.18 und 3.19 für allgemeine λ STs.

Wie geht man nun im Fall der minimalen sequentiellen Transducer vor? Wenn wir einen λ ST T gegeben haben, welcher auch sequentiell ist, können wir nach

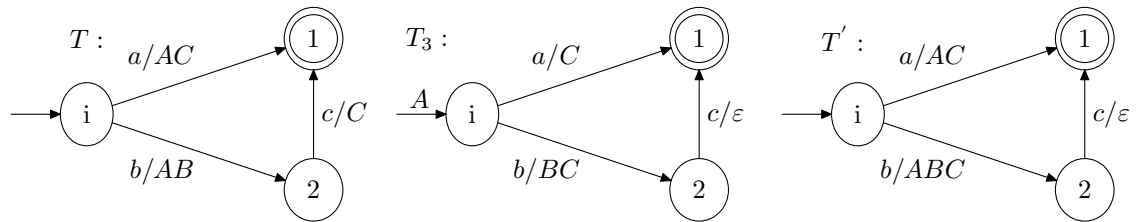
Lemma 3.8 einen äquivalenten ST konstruieren und daraus wiederum den minimalen ST, äquivalent zu T . Ohne Einschränkung lässt sich die Frage also auf das Problem reduzieren, zu einem sequentiellen Transducer den minimalen sequentiellen Transducer zu finden. Die λ -sequentiellen Transducer enthalten zwar die STs als Teilmenge, aber der Transducer T_3 aus Satz 3.22 muss nicht sequentiell sein. Für $P_T(i) \neq \varepsilon$ oder $\lambda \neq \varepsilon$ ist T_3 ein λ ST, für den zunächst gilt: $\lambda' \neq \varepsilon$. Die folgenden Untersuchungen sollen als Vermutung oder Konstruktionsanleitung verstanden werden. In der Literatur wurden bisher keine korrekten Ergebnisse für die Konstruktion des minimalen STs zu einer sequentiellen Funktion angegeben. Das Lemma 3.13 und das Korollar 3.14 lassen eine grundlegend neue Herangehensweise vermuten.

Sei $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma)$ ein fester ST. Nach Anwendung der Schritte aus Satz 3.22 auf T könnten wir aus dem erhaltenen λ -sequentiellen Transducer $T_3 = (\Sigma, \Delta, Q_2, i_2, F_2, \lambda', \delta_2, \sigma_2)$, der den minimalen λ ST zu T darstellt, einen äquivalenten ST mit minimaler Anzahl von Zuständen konstruieren. Nach dem obigen Lemma 3.8 bleibt die Anzahl der Zustände dann erhalten oder erhöht sich gegebenenfalls um eins. Die Transitionsmenge von T_3 sei E_3 . Wir untersuchen T_3 genauer.

Gilt $\lambda' = \varepsilon$, so ist $T' := (\Sigma, \Delta, Q_2, i_2, F_2, \delta_2, \sigma_2)$ der minimale ST zu T .



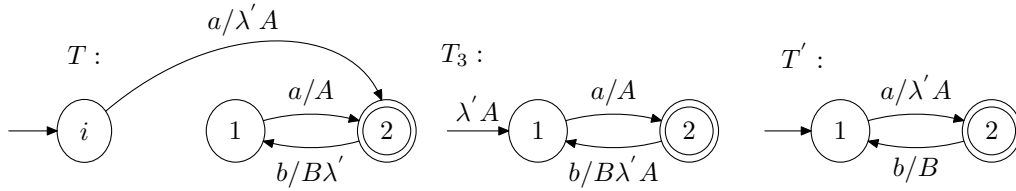
Gibt es in T_3 keine Transitionen, welche zu i_2 führen, so kann λ' von links an die Ausgabe von Transitionen startend in i_2 konkateniert werden und als Eingangswort selbst vernachlässigt werden. Hier im Beispiel ist λ' gleich A.



Im Falle, dass zwar Transitionen in T_3 zu i_2 führen, jedoch eine „Suffix-Eigenschaft“ gilt, können wir aus T_3 ebenfalls einen äquivalenten ST mit gleicher Anzahl von Zuständen konstruieren. Die Eigenschaft ist die folgende:

Wir betrachten zum einen Pfade von i_2 nach i_2 , welche keinen finalen Zustand (aus F_2) passieren, zum anderen Pfade von F_2 nach i_2 . Enthält der größte gemein-

same Suffix von Ausgaben über der Menge dieser Pfade λ' selbst als Suffix, so könnten wir uns einen „Suffix-Transducer“ über Σ, Δ vorstellen (oder gedanklich konstruieren) mit gleichen Zuständen und gleicher Anzahl von Transitionen, gleichem λ' und gleichem Input-Automat, der die Ausgabe bis i_2 so spät wie möglich ausgibt. Nennen wir diesen Transducer $S(T_3)$. Die Ausgaben von Transitionen in $S(T_3)$, die auf i_2 treffen, enthalten λ' als Suffix und könnten von rechts mit $(\lambda')^{-1}$ konkateniert werden (oder äquivalent: gestrichen werden). Dann könnten wir λ' von links an die Ausgabe der Transitionen startend in i_2 konkatenieren.



Bei all diesen drei vorgestellten Fällen bilden wir aus T_3 den minimalen ST zu T , dieser hat die gleiche Anzahl von Zuständen, wie der Index von R_f angibt.

Die Menge der sequentiellen Transducer, welche bisher nicht in den obigen drei Fällen betrachtet wurden, bezeichne ich mit L . Das sind also diejenigen sequentiellen Transducer T , für die, nach Anwendung der Schritte aus Satz 3.22, der resultierende λ ST T_3 folgende Eigenschaften besitzt:

1. $\lambda' \neq \varepsilon$
2. Es gibt einen Pfad $\pi \in i_2 \xrightarrow{w} i_2$, so dass λ' kein Suffix von $\sigma_2(\pi)$ ist oder es gibt einen Pfad π von $f \in F_2$ nach i_2 , so dass λ' kein Suffix von $\sigma_2(\pi)$ ist.

Ich bezeichne die Menge der sequentiellen Funktionen, die durch Transducer aus L dargestellt werden, mit \mathcal{F}_L . Außerdem sei \mathcal{F}_U die Menge derjenigen sequentiellen Funktionen, für die der minimale ST einen Zustand mehr besitzt als der minimale λ ST. Jetzt kann man eventuell zeigen, dass die beiden Mengen \mathcal{F}_L und \mathcal{F}_U zusammen fallen. Gegebenenfalls muss die Menge \mathcal{F}_L vergrößert werden.

Ein möglicher Satz zur Berechnung des minimalen sequentiellen Transducers zu einem gegebenen ST könnte dann wie folgt aussehen:

Vermutung 3.23. Sei $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma)$ ein ST, der die Funktion f darstellt. Der mit Satz 3.22 aus T konstruierte λ ST ist $T_3 = (\Sigma, \Delta, Q_2, i_2, F_2, \lambda', \delta_2, \sigma_2)$ mit der Transitionsmenge E_3 . Ein minimaler sequentieller Transducer zu T , T' , entsteht mit einer Fallunterscheidung wie folgt:

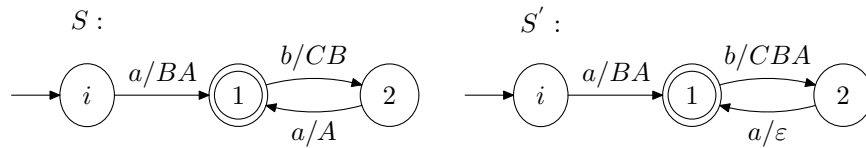
1. Es gilt $\lambda' = \varepsilon$. Dann ist $T' := (\Sigma, \Delta, Q_2, i_2, F_2, \delta_2, \sigma_2)$.
2. Es gibt in T_3 keine Transition $(q, a, x, i_2) \in E_3$. Die Transitionsmenge E' für T' entsteht aus E_3 , indem Kanten (i_2, a, x, q) durch $(i_2, a, \lambda'x, q)$ ersetzt werden. Die neue Ausgabefunktion ist σ' . Es ist $T' := (\Sigma, \Delta, Q_2, i_2, F_2, \delta_2, \sigma')$.

3. Für $S(T_3) = (\Sigma, \Delta, Q_2, i_2, F_2, \delta_S, \sigma_S)$ mit der Transitionsmenge E_S gilt, dass alle Transitionen, die auf i_2 treffen, λ' als Suffix enthalten. Die Menge E' entsteht aus E_S , indem Kanten (i_2, a, x, q) durch $(i_2, a, \lambda'x, q)$ bzw. $(q, a, x\lambda', i_2)$ durch (q, a, x, i_2) ersetzt werden, und wird durch σ' und δ_2 beschrieben. Dann ist $T' = (\Sigma, \Delta, Q_2, i_2, F_2, \delta_2, \sigma')$.

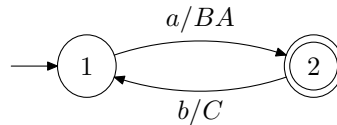
4. Auf den Transducer T_3 treffen nicht obige drei Fälle zu. Dann ist T' derjenige ST , der durch die im Beweis von Lemma 3.8 angegebene Konstruktion entsteht. Dieser hat genau einen Zustand mehr als der minimale λST zu T .

M. P. Béal und O. Carton geben in [2] einen Algorithmus für die Berechnung der Präfix-Funktion an. Dieser soll als Vorstufe der Minimalisierung sequentieller Transducer dienen. Die beiden Autoren betrachten dabei die Ausgabe von Pfaden zu finalen oder initialen Zuständen. Ich gebe ein Beispiel dafür an, dass eine darauf folgende klassische Minimalisierung zu keinem minimalen sequentiellen Transducer führt. Die betrachtete Funktion liegt nicht in L .

Seien dazu S und S' definiert durch:



Der Transducer S' entsteht aus S mit Bildung des Präfixes, welcher auch den Initialzustand einschließt (der Präfix von i ist also ε , deshalb wurde die Ausgabe BA nicht vorgezogen), und einer Minimalisierung endlicher Automaten. Allerdings ist S' nicht minimal, denn



ist ein äquivalenter Transducer zu S' .

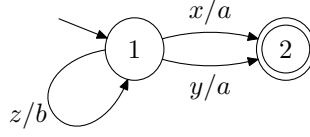
Bemerkung 3.24. Es ist ein offenes Problem, eine (transducer-unabhängige) Äquivalenzrelation anzugeben, deren Index für eine sequentielle Funktion f mit der Anzahl der Zustände des minimalen sequentiellen Transducers zu f zusammenfällt. Wenn man die bereits definierten Relationen verwendet, müsste man eine Bedingung für eine sequentielle Funktion f finden, aus der folgt, dass diese Funktion durch keinen Transducer aus der oben definierten Menge L dargestellt werden kann. Die Bedingung muss unabhängig von einem f darstellenden Transducer sein.

3.3 Ein Algorithmus zur Minimalisierung sequentieller Transducer

In seinen beiden Publikationen [13, 16] gibt Mohri Algorithmen zur Minimalisierung sequentieller bzw. λ -sequentieller Transducer an. Dabei auftretende Fehler, z. B. die Definition des Präfixes, die sicher stellen soll, dass nach Ausführung des gesamten Algorithmus die Äquivalenz der dargestellten Funktion erhalten bleibt, habe ich im vorherigen Abschnitt diskutiert. Die grundlegende Vorgehensweise ist in zwei Schritte unterteilt, die im Satz 3.22 dargestellt wurden. Im ersten Schritt geht man zu dem Präfix des Transducers über. Darauf wendet man im zweiten Schritt die klassische Minimalisierung von deterministischen endlichen Automaten an. Der Transducer wird dabei als DEA betrachtet mit den Ein- und Ausgaben der Transitionen als Buchstaben. Der Präfix-Automat eines Transducers ist im Wesentlichen ein Transducer mit gleichem zugrunde liegenden Graphen, der die Ausgabe so früh wie möglich gibt. Die durch Mohri angegebenen Algorithmen für die Minimalisierung sequentieller und λ -sequentieller Transducer haben eine Komplexität von $\mathcal{O}(S + |Q| + |E| \cdot (\log |Q| + |P_{max}|))$. Dabei ist S die Summe aller Längen von Wörtern, die über Transitionen ausgegeben werden, E die Transitionsmenge und P_{max} die maximale Länge aller $P_T(q)$ ($q \in Q$). Der Algorithmus für STs [13] funktioniert, entgegen der Behauptung von Mohri, nur für STs mit $P_T(i) = \varepsilon$ (siehe Beispiel 3.16 bzw. vorheriger Abschnitt). Im Folgenden werden wir das Prinzip von Mohri anwenden. Das Entscheidende für die Konstruktion des minimalen λ STs oder STs ist die Berechnung der Präfix-Funktion. Für alle Zustände des betrachteten Transducers muss der größte gemeinsame Präfix von Ausgaben über Pfade dieses Zustandes zu finalen Zuständen berechnet werden. Ich stelle die Idee von Mohri zu dieser Berechnung größter gemeinsamer Präfixe vor. Eine vollständige Betrachtung der Algorithmen und ihrer Komplexität für die Minimalisierung λ -sequentieller (und mit Vermutung 3.23 eventuell sequentieller) Transducer (der Satz 3.22 ist dann wesentlich) folgt in 3.3.2 und 3.3.3. Ein Problem entsteht, wenn der gegebene Transducer Kreise enthält, über denen ε ausgegeben wird.

3.3.1 Der Präfix eines Automaten

Die Idee der Konstruktion des Präfixes ist unabhängig von dem Konzept der Transducer, da die Eingabe dabei vernachlässigt wird. Der Präfix kann somit auch allgemein für Automaten definiert werden [13]. Diese enthalten allerdings nicht alle Fälle, welche dann bei der Minimalisierung der STs eine Rolle spielen. Ein sequentieller Transducer kann beispielsweise folgendermaßen aussehen:



Deshalb müssen die Zustandsübergänge des Automaten mit Multimengen definiert werden. (In [16] wird das berücksichtigt, allerdings nicht in [13] bei der Betrachtung sequentieller Transducer.) Ich beschränke mich hier auf die Betrachtung eines einzelnen Initialzustandes, da der Präfix für Automaten im Kontext der Minimalisierung sequentieller Transducer gesehen wird. Mohri [16, S. 197] definiert Automaten mit einer Initialzustandsmenge.

Definition 3.25. Ein *Automat* ist ein 6-Tupel $G = (\Delta, Q, i, F, \lambda, \delta)$, wobei Δ ein Alphabet, Q eine Menge von Zuständen, $i \in Q$ der Initialzustand, $F \subseteq Q$ eine Menge von Endzuständen und $\lambda \in \Delta^*$ das Initialwort ist, welches von links an jedes erkannte Wort konkateniert wird. Die Funktion δ ist die Zustandsübergangsfunktion, die $Q \times \Delta^*$ nach \mathbb{N}^Q abbildet, die Menge der Multimengen von Q . Die zu G gehörige Multimenge von Transitionen ist dann $E_G = \{(q, w, \delta(q, w)) \mid (q, w) \in Q \times \Delta^*\}$. Das Wort $\lambda \cdot w$ wird genau dann von G erkannt, falls es für w einen Pfad von i zu einem finalen Zustand gibt.

Sei $P_G : Q \rightarrow \Delta^*$ die Präfix-Funktion aus dem letzten Abschnitt, welche jedem Zustand den größten gemeinsamen Präfix von erkennbaren Wörtern startend in q zuordnet. Wir verwenden, soweit dies möglich ist, die Notationen und Begriffe von Mohri [16, S. 179ff.]. So ist $\text{trans}[q]$ die Menge der Transitionen startend in q , $\text{trans}^T[q]$ die Menge der Transitionen, die zu q führen, $l(t)$ das einzulesende Wort einer Transition t und $n(t)$ der Zustand, auf den die Transition t zeigt. Mit der Präfix-Funktion ergibt sich ein Präfix für Automaten:

Definition 3.26. Der *Präfix* $P(G) = (\Delta, Q, i, F, \lambda \cdot P_G(i), \delta_P)$ zu dem Automaten $G = (\Delta, Q, i, F, \lambda, \delta)$ mit der Transitionsmenge E_G ist ein Automat mit verändertem Initialwort und neuer Transitionsmenge:

$$E_{P(G)} = \left\{ \left(q \xrightarrow{P_G(q)^{-1}uP_G(p)} p \mid (q \xrightarrow{u} p) \in E_G \right) \right\}.$$

Der Präfix $P(G)$ ist dann äquivalent zu G . Man könnte sich erneut ein Gleichungssystem

$$\begin{aligned} \forall q \in F : P_G(q) &= \varepsilon \\ \forall q \in Q \setminus F : P_G(q) &= \left(\bigwedge_{t \in \text{trans}[q]} l(t) P_G(n(t)) \right). \end{aligned}$$

vorstellen, welches möglicherweise mehrere Lösungen hat. Dies suggeriert einen rekursiven Algorithmus für $P(G)$. Um $P_G(q)$ zu berechnen, müssten alle Werte

von P für die Zustände aus der Adjazenzliste von q bereits vorhanden sein. Allerdings muss es eine solche lineare Ordnung nicht geben, denn zwei Zustände können auf einem Kreis liegen.

Für den Fall, dass G einem direkten azyklischen Graphen entspricht, erfüllt die Breitensuche mit Start in F entgegen der Pfeilrichtung die Bedingung. Man betrachtet Zustände in dieser Ordnung und kann mit obiger Definition für alle Zustände aus G $P_G(q)$ und damit auch $P(G)$ berechnen. Die Komplexität ist $\mathcal{O}(|Q| + |E|)$ [16].

Enthält G Kreise, werden SCCs (starke Zusammenhangskomponenten) betrachtet, denn es gibt dann eine solche gesuchte lineare Ordnung der SCCs, gegeben jetzt durch die Breitensuche im Komponentengraphen G^{SCC} von G , so dass man dann wie im azyklischen Fall vorgehen kann, wobei aber die Transitionen innerhalb der SCCs extra verändert werden müssen. Der Komponentengraph G^{SCC} kann in $\mathcal{O}(|Q| + |E|)$ berechnet werden. Falls die Adjazenzliste eines Zustandes $q \in scc_1$ (scc_1 ist ein Zustand von G^{SCC}) den Zustand $p \in scc_2$ enthält, dann erscheint p vor q in der linearen Ordnung. Um $P(G)$ zu berechnen, werden Transitionen entsprechend der Ordnung der SCCs verändert. Wenn eine Komponente scc betrachtet wird, werden die größten Präfixe in scc berechnet, dann alle Transitionen, startend in bzw. ausgehend von $q \in scc$, folgendermaßen verändert:

$$\begin{aligned} \forall t \in \text{trans}[q] : l(t) &:= P_G(q)^{-1} l(t), \text{ falls } t \text{ nicht auf } scc \text{ zeigt} \\ \forall t \in \text{trans}^T[q] : l(t) &:= l(t) P_G(q), \text{ falls } t \text{ auf } scc \text{ zeigt.} \end{aligned}$$

Alle Transitionen innerhalb scc werden wie in der Präfix-Definition verändert. Seien scc aus G^{SCC} und alle SCCs, die vor scc in der Ordnung erscheinen, bereits berechnet. Der Start ist durch den Anfang der Ordnung gegeben, denn dort gibt es keine Transitionen, die herausführen. Wir nehmen ohne Einschränkung an, G sei schlank. Mohri definiert für $u \in scc$ ([16]):

$$\begin{aligned} I[u] &= \{t \in \text{trans}[u] : n(t) \in scc\} \\ O[u] &= \{t \in \text{trans}[u] : n(t) \notin scc\} \end{aligned}$$

als Transitionen innerhalb von scc bzw. herausführend aus scc . Dann betrachtet er für $u \in scc$ dieses Gleichungssystem:

$$\begin{aligned} u \in F : \quad X_u &= \varepsilon, \\ u \notin F, O[u] = \emptyset : \quad X_u &= \left(\bigwedge_{t \in I[u]} l(t) X_{n(t)} \right), \\ u \notin F, O[u] \neq \emptyset : \quad X_u &= \left(\bigwedge_{t \in I[u]} l(t) X_{n(t)} \right) \wedge \left(\bigwedge_{t \in O[u]} l(t) \right). \end{aligned} \tag{2}$$

Es soll für alle $u \in scc$ auf die eindeutige Lösung $X_u = P_G(u)$ führen. M. P. Béal, O. Carton und C. Choffrut [2, 8] erkannten, dass dieses Gleichungssystem (kurz GS) für den Fall, dass der Automat Kreise enthält, deren Transitionen alle ε einlesen, nicht zwangsläufig eine eindeutige Lösung besitzt und Mohris Algorithmus somit für bestimmte Fälle nicht korrekt ist. In Beispiel 3.17 wurde bereits auf das Problem bei der rekursiven Angabe von P_G hingewiesen. Ich gebe ein weiteres Beispiel. Wenn wir den Komponentengraphen T in Beispiel 3.27 betrachten, gibt es genau 3 Lösungen für das zugehörige Gleichungssystem,

$$X_{u_1} = X_{u_2} \wedge de, \quad X_{u_2} = X_{u_3} = X_{u_4} = X_{u_1},$$

der 3. Komponente des Komponentengraphen, nämlich:

1. $X_{u_1} = X_{u_2} = X_{u_3} = X_{u_4} = \varepsilon$
2. $X_{u_1} = X_{u_2} = X_{u_3} = X_{u_4} = d$
3. $X_{u_1} = X_{u_2} = X_{u_3} = X_{u_4} = de$.

Dabei ist die 3. Lösung die eigentlich gesuchte.

Beispiel 3.27. In Abbildung 11 entsteht der Automat T' aus T nach Bearbeitung der letzten beiden Komponenten durch den Algorithmus und der zweiten Lösung des GS.

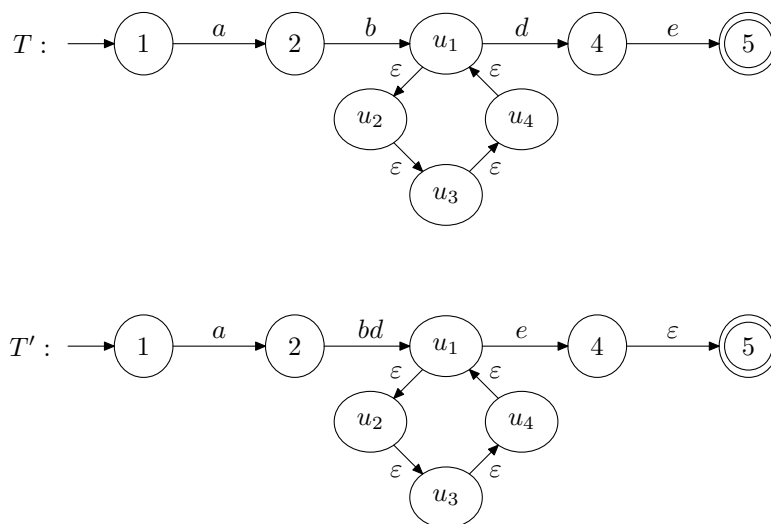


Abbildung 11: Automat mit mehreren Lösungen

Wir schließen zunächst solche ε -Kreise aus und gehen wie Mohri vor, im Abschnitt 3.4 werden die restlichen Fälle vorgestellt.

Das GS (2) hat nun für alle $u \in scc$ eine eindeutige Lösung $X_u = P_G(u)$. Um diese zu finden geht man von (2) schrittweise, durch Veränderungen von Variablen, zu neuen Gleichungssystemen über, aus deren Lösung man die eigentlich gesuchte konstruieren kann. Für $u \in scc$ definiert Mohri:

$$\pi_u \leftarrow \left(\bigwedge_{t \in \text{trans}[u]} l(t) \right), \quad u \notin F \quad (3)$$

$$\pi_u \leftarrow \varepsilon, \quad \text{sonst.}$$

Für genau einen Zustand $u_0 \in scc$ mit $\pi_{u_0} \neq \varepsilon$ verändert er dann die zugehörige Variable:

$$X_{u_0} \leftarrow \pi_{u_0} Y_{u_0}$$

$$\text{Für alle } u \in scc \setminus \{u_0\} : X_u \leftarrow Y_u$$

So wird zu einem neuen System übergegangen:

$$\begin{aligned} u \in F : \quad & Y_u = \varepsilon, \\ u \notin F, O[u] = \emptyset : \quad & Y_u = \left(\bigwedge_{t \in I[u]} l'(t) Y_{n(t)} \right), \\ u \notin F, O[u] \neq \emptyset : \quad & Y_u = \left(\bigwedge_{t \in I[u]} l'(t) Y_{n(t)} \right) \wedge \left(\bigwedge_{t \in O[u]} l'(t) \right), \end{aligned} \quad (4)$$

mit

$$\begin{aligned} \forall t \notin (\text{trans}[u_0] \cup \text{trans}^T[u_0]) : \quad & l'(t) = l(t), \\ \forall t \in \text{trans}[u_0] \setminus (\text{trans}[u_0] \cap \text{trans}^T[u_0]) : \quad & l'(t) = \pi_{u_0}^{-1} l(t), \\ \forall t \in \text{trans}^T[u_0] \setminus (\text{trans}[u_0] \cap \text{trans}^T[u_0]) : \quad & l'(t) = l(t) \pi_{u_0}, \\ \forall t \in (\text{trans}[u_0] \cap \text{trans}^T[u_0]) : \quad & l'(t) = \pi_{u_0}^{-1} l(t) \pi_{u_0}. \end{aligned} \quad (5)$$

Nun kann man zeigen [16, S. 182f.], dass das System (2) endlich viele der obigen Veränderungen von Variablen zulässt und man dann zu einem System $(Z_u)_{u \in scc}$, $Z_u = \varepsilon$ gelangt. Wenn man jetzt die einzelnen π_u , welche bei den Übergängen eine Rolle gespielt haben, konkateniert, kann man für alle $q \in scc$ die Lösung $P_G(q)$ konstruieren. Bei jedem Schritt verändert man auch die Transitionen ($l'(t)$), so dass gilt:

Lemma 3.28. (Mohri [16]) Sei $S = (Z_u)_{u \in scc}$ mit $Z_u = \varepsilon$ das GS, welches aus (2) durch schrittweise Veränderung von Variablen entsteht. Sei $\{l(t), t \in scc\}$ die neue Menge von Transitionen von S (definiert durch (5)). Dann sind diese Transitionen genau die Transitionen aus scc im Präfix $P(G)$.

In jedem einzelnen Schritt, in dem man zu einem neuen System übergeht, wird bei einem bestimmten Zustand ein Buchstabe des Alphabetes vorgezogen. Nach endlich vielen solchen Systemumwandlungen kommt man zu dem trivialen System aus Lemma 3.28.

3.3.2 Der Algorithmus

In Anlehnung an Mohri kann sein angegebener Algorithmus [16, S. 183f.] QUASIDETERMINIZATION für unsere Berechnung des Präfixes eines Automaten G übernommen werden. Für uns gilt die Einschränkung $P_G(i) = \varepsilon$ nicht. Der von Mohri angegebene Algorithmus behandelt allerdings den allgemeinen Fall bereits, es ist im eigentlichen Algorithmus an keiner Stelle notwendig, dass $P_G(i) = \varepsilon$ ist. Der Algorithmus wird hier nicht noch einmal aufgeführt. Während des Abarbeitens muss $P_G(i)$ gespeichert werden und das neue Initialwort von $P(G)$ gleich $\lambda \cdot P_G(i)$ gesetzt werden.

In [16] gibt es auch den Korrektheitsbeweis für seine Anwendung, welche für uns in analoger Weise gilt. Wir konstruieren uns also in Gedanken eine Algorithmus PRÄFIX, der Mohris QUASIDETERMINIZATION verwendet, anschließend das neue Initialwort berechnet und den Präfix $P(G)$ ausgibt.

Im Abschnitt 3.2 wurden die Schritte zur Minimalisierung der STs λ STs beschrieben. Wie obiger Algorithmus in Bezug auf die Komplexität darin eingebettet ist, soll jetzt beschrieben werden. Dabei übernehme ich die Überlegungen von Mohri und verwende Satz 3.4.

3.3.3 Komplexität und Konstruktion des minimalen λ STs bzw. STs

Für die Komplexität von PRÄFIX ergibt sich analog zu Mohri $\mathcal{O}(|Q| + |E| + (|E| - (|Q| - |F|)) \cdot |P_{max}|)$, wobei P_{max} einer der längsten der größten gemeinsamen Präfixe $P(q)$ ist [16, S. 185f.]. Darin sind die Bildung des Komponentengraphen, Initialisierung und die Definition der Ordnung enthalten. Betrachten wir zunächst einen λ ST T . Es soll dabei keine Kreise geben, über deren Transitionen ε ausgegeben wird. Ob alle Zustände auf einem erfolgreichen Pfad liegen, also T schlank ist, spielt für den Präfix keine Rolle. Man beachte, dass das Resultat einer Minimalisierung endlicher Automaten stets ein Automat ist, welcher schlank ist. Der Ausgabe-Transducer von T ist ein Automat, auf den wir PRÄFIX anwenden. Im Folgenden konstruieren wir den äquivalenten Transducer zu T , dessen Ausgabe-Automat nun das Ergebnis des Algorithmus ist. Unter Vernachlässigung des Initialwortes fassen wir den resultierenden Transducer als DEA auf (Eingabe und Ausgabe einer Transition jeweils als ein Buchstabe betrachtet) und bilden den Minimalautomaten. Nach Satz 3.4 und [16] ergibt sich für die Minimalisierung eines λ -sequentiellen Transducers eine Komplexität von $\mathcal{O}(S + |Q| + |E| \cdot (\log |Q| + |P_{max}|))$. Die Größe S bezeichnet die Summe al-

ler Längen von Wörtern, die über Transitionen ausgegeben werden. Unter der Annahme, dass die Anzahl der Transitionen $|E|$ größer ist als die Anzahl der Zustände $|Q|$, können wir dies mit $\mathcal{O}(S + |E| \cdot (P_{max} + |Q| + 1))$ abschätzen.

Gehen wir von einem λ ST T aus und wollen den äquivalenten minimalen ST konstruieren, so muss die Vermutung 3.23 beachtet werden. Zunächst berechnen wir den minimalen λ ST zu T . Danach überprüfen wir die Fälle in 3.23. Für die Berechnung des größten gemeinsamen Suffixes von Ausgaben über Pfade startend in $F \cup \{i\}$, wenden wir PRÄFIX auf den Umkehr-Automaten des Ausgabe-Automaten von T an, wobei die Werte für $F \cup \{i\}$ mit ε vorgegeben sind. Die Ausgaben können dann in Richtung Initialzustand verschoben werden (die Ausgabe wird vor i also so spät wie möglich ausgegeben). Sonst werden die Anweisungen, die in 3.23 angegeben sind, befolgt. Die Konstruktion im Lemma 3.8 hat eine Komplexität von $\mathcal{O}(|E|)$, damit würde die Komplexität der Konstruktion des minimalen ST zu einem gegebenen λ ST ebenso $\mathcal{O}(S + |Q| + |E| \cdot (\log |Q| + |P_{max}|))$ sein.

3.4 Die Vervollständigung von Mohris Algorithmus

M. P. Béal und O. Carton geben in [2] einen anderen Algorithmus für die Konstruktion des Präfixes eines Automaten an. Dieser funktioniert für alle Automaten in der Hinsicht, dass der Fehler von Mohri bei Kreisen, über denen ε ausgegeben wird, nicht mehr auftritt. Allerdings betrachten die beiden Autoren die größten gemeinsamen Präfixe der Ausgaben von Pfaden eines Zustandes zu finalen Zuständen oder dem Initialzustand. Im letzten Teil von 3.2 habe ich ein Beispiel angegeben, welches zeigt, dass diese Berechnung mit einer darauf folgenden Minimalisierung nicht zu dem Minimal-Transducer führt. Die Untersuchungen der beiden Autoren zur Minimalisierung sequentieller Transducer ist somit nicht korrekt.

In diesem Abschnitt beschreibe ich die Idee von M. P. Béal und O. Carton [2] zur Konstruktion des Präfixes eines Automaten. Wir ändern den Algorithmus auf die Betrachtung von Pfaden, die zu finalen Zuständen gehen, ab. So erhalten wir mit 3.3.2 und 3.3.3 einen weiteren Algorithmus für die Minimalisierung λ -sequentieller Transducer. Zunächst variieren wir die Definition eines Automaten [2, S. 3]. Die Transitionen werden nun über Multimengen definiert: $E \subset \mathbb{N}^{Q \times \Delta^* \times Q}$, so dass wir mit $I := \{i\}$ in unserer alten Definition (3.25) von Automaten sind. Wir fixieren einen Automaten $G = (\Delta, Q, i, F, \lambda, \delta)$. Der größte gemeinsame Präfix über Pfade, die von einem Zustand q zu einem finalen Zustand in G führen, wurde mit $P_G(q)$ bezeichnet. Es sei $p_G(q)$ der erste Buchstabe (ggf. ε) von $P_G(q)$. Jetzt bezeichne P_G das Wort P_{max} im Automaten G . Wenn $|P_G| > 0$, wird aus $G = (\Delta, Q, i, F, E)$ ein äquivalenter Automat $G' = (\Delta, Q, i, F, E')$ konstruiert mit:

$$E' = \left\{ (q \xrightarrow{p_G(q)^{-1}u p_G(p)} p) \mid (q \xrightarrow{u} p) \in E \right\}.$$

Dieser Vorgang wird wiederholt und führt zu $P(G)$, denn in G' wird über einem Pfad von q_1 zu q_2 genau $p_G(q_1)^{-1}w p_G(q_2)$ eingelesen, falls in G das Wort w eingelesen wurde. Wörter über erfolgreichen Pfaden bleiben also identisch. Um G' zu konstruieren, wird wie folgt vorgegangen. Sei G_ε der Teilautomat, welcher aus G entsteht, indem man die ε -Kanten betrachtet. Die Zustandsmenge bleibt erhalten. In dem zu G_ε gehörigen Komponentengraph G_ε^{SCC} fallen die Präfixe $P_G(q)$ und damit auch die $p_G(q)$ für Zustände der gleichen Komponente zusammen und müssen nur einmal berechnet werden. In einem ersten Schritt wird für alle Zustände q des Gesamtautomaten der Buchstabe $p_G(q)$ berechnet. Im zweiten Schritt werden dann die nötigen Veränderungen der Transitionen vorgenommen, um G' zu erhalten. Wie wird nun $p_G(q)$ für einen Zustand q berechnet? Es bezeichne $S(q)$ die Menge der Anfangs-Buchstaben von Wörtern $l(t) \neq \varepsilon$, deren Transitionen in q starten. Die Autoren führen ein Feld $\text{letter}[q]$ ein, welches mit

$$\text{letter}[q] := \begin{cases} \varepsilon, & q \in F \\ \bigwedge_{s \in S(q)} s, & S(q) \neq \emptyset \\ \perp, & \text{sonst} \end{cases}$$

initialisiert wird. Wir haben hier $\text{letter}[q]$ nur für finale Zustände auf ε gesetzt. Dieses Feld wird nun für alle Zustände q schrittweise zu $\text{letter}[q] := p_G(q)$, also Werten aus $\Sigma \cup \{\varepsilon\}$ verändert. Entweder man setzt $p_G(q) = \varepsilon$ für Zustände mit $\text{trans}[q] = \emptyset$ oder man nimmt an, dass G schlank ist, und muss dann nur diejenigen Zustände weiter untersuchen, von denen ε -Kanten abgehen und gegebenenfalls $\bigwedge_{s \in S(q)} s \neq \varepsilon$ ist. Dann werden ε -Kanten solange weiter verfolgt, bis man entweder zu einem Zustand gelangt, von dem keine weiteren ε -Kanten abgehen, oder man auf einen Zustand trifft, welchen man bei der „Verfolgung“ bereits überschritten hat. (Dies würde nämlich bedeuten, dass der betrachtete Pfad in einer Komponente von G_ε^{SCC} liegt.) Von dem durch (r_1, ε, r_2) erreichten Zustand r_2 verändert man nun rückwärts die Werte $\text{letter}[r_2] := \text{letter}[r_1] := \dots := \text{letter}[q]$. Schließlich gilt für alle Zustände $q \in Q$: $\text{letter}[q] = p_G(q)$. Der angegebene Algorithmus lässt sich problemlos auf die Betrachtung der Pfade zu finalen Zuständen erweitern.

Für die Komplexität des gesamten Algorithmus ergibt sich, wie in [2, S. 11f.] gezeigt, $\mathcal{O}((|Q| + |E|) \cdot (P_{max} + 1))$, was unter der Annahme, dass es höchstens eine Kante weniger als Zustände gibt (dies ist u. a. der Fall, wenn G schlank ist), zu $\mathcal{O}(|E| \cdot (P_{max} + 1))$ führt. Die Erweiterung der Transitionsmenge E auf Multimengen ist darin enthalten. Der Algorithmus, den M. P. Béal und O. Carton zur Konstruktion des Präfix-Automaten angeben, hat demnach die gleiche Komplexität wie Mohris, ist allerdings mit unserer Erweiterung auf alle Automaten anwendbar. Sieht man den Algorithmus im Kontext von Transducern, so wird dadurch auch ein Algorithmus für die Minimalisierung von sequentiellen und λ -sequentiellen Transducern gegeben. Das Ergebnis einer Minimalisierung eines endlichen Automaten soll erneut schlank sein. Um die vollständigen Algorithmen

auch für M. P. Béal und O. Carton zu erhalten, kommen die gleichen Berechnungen wie bei Mohri dazu. So muss für λ STs zum Beispiel mindestens einmal $P_G(i)$ einfließen. Die Idee von Mohri, die $p_G(q)$ über die Transitionen auszurechnen, welche direkt vom Zustand q abgehen, kann man in dieser Form nicht auf alle STs bzw. λ -STs erweitern. Wenn ε -Pfade zu $p_G(q)$ führen, müsste man sie verfolgen und somit die direkten Transitionen verlassen. Dies würde dann genau zum Algorithmus von M. P. Béal und O. Carton führen.

3.5 Subsequentielle und p -subsequentielle Transducer

Im Abschnitt 2.2 wurden subsequentielle bzw. p -subsequentielle Transducer als Erweiterung der STs vorgestellt. Durch Mohri [16] werden diese Transducer erneut mit einem zusätzlichen Initialwort λ definiert. Wir sprechen hier also von λ -subsequentiellen und λ - p -subsequentiellen Funktionen sowie von λ SSTs und λp SSTs. Wie stehen diese nun in Beziehung?

Dazu führe ich neue Bezeichnungen ein. Die Menge der Funktionen bzw. Relationen, die durch einen ST (bzw. λ ST, SST, λ SST, p SST, λp SST) dargestellt werden können, wird mit \mathcal{F}_{ST} (bzw. $\mathcal{F}_{\lambda ST}$, \mathcal{F}_{SST} , $\mathcal{F}_{\lambda SST}$, \mathcal{F}_{pSST} , $\mathcal{F}_{\lambda pSST}$) bezeichnet. Sie stehen in der folgenden Beziehung:

Satz 3.29. *Für $p \geq 2$ gilt:*

$$\mathcal{F}_{ST} \subsetneq \mathcal{F}_{\lambda ST} \subsetneq \mathcal{F}_{SST} = \mathcal{F}_{\lambda SST} \subsetneq \mathcal{F}_{pSST} = \mathcal{F}_{\lambda pSST}.$$

Beweis: Die erste (echte) Inklusion folgt aus Lemma 3.8. Aus jedem λ ST T kann man analog zur Konstruktion 3.9 einen äquivalenten SST T' konstruieren, ε kann durch T' auf λ abgebildet werden, so dass das Problem in 3.9 nicht auftritt. Die echte Inklusion folgt mit Abbildung 12. Mit dem Charakterisierungssatz für subsequentielle Funktionen, 2.26, wissen wir aber auch bereits, dass es sequentielle Funktionen gibt, die nicht in \mathcal{F}_{SST} sind.

Jeder SST ist auf natürliche Weise ein λ SST, die umgekehrte Inklusion folgt erneut mit einer Konstruktion analog 3.9.

$\mathcal{F}_{\lambda SST} \subsetneq \mathcal{F}_{pSST}$ ist trivial, da \mathcal{F}_{pSST} bereits Relationen enthält. Die anderen Beziehungen wurden gezeigt oder folgen auf ähnliche Weise. \square

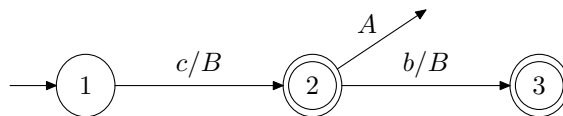


Abbildung 12: Ein subsequentieller Transducer, der nicht λ -sequentiell ist.

Im Weiteren werde ich die Algorithmen aus der Literatur zur Minimalisierung von λp SSTs vorstellen, vergleichen und gegebenenfalls erweitern. Es soll auch

auf die Probleme, wie sie im Abschnitt 3.2 auftraten, eingegangen werden. So könnte zum Beispiel der minimale SST zu einer subsequentiellen Funktion mehr Zustände als der minimale λ SST besitzen, da bei der Konstruktion 3.9 ein neuer Zustand eingeführt wurde. Im Falle der endlichen Automaten oder der sequentiellen Transducer haben die minimalen Automaten bzw. STs auch eine minimale Transitionsmenge [15]. Was gilt für die erweiterten Transducer und wie ist die Komplexität aller betrachteten Algorithmen?

Da mit einem SST auch Endausgaben möglich sind, könnte es sequentielle Funktionen geben, deren minimaler ST mehr Zustände besitzt als der minimale SST. Mit Satz 3.31 zeige ich, dass dies nicht der Fall ist.

Lemma 3.30. *Zu jedem subsequentiellen Transducer, welcher eine sequentielle Funktion darstellt, gibt es einen äquivalenten sequentiellen Transducer mit gleicher Anzahl von Zuständen.*

Beweis: Sei $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma, \rho)$ ein SST für die sequentielle Funktion f . Ohne Einschränkung kann ich annehmen, dass die Endausgabe-Funktion ρ auf F total definiert ist. Ich betrachte den Ausgabe-Automaten von T , A_T , mit der Transitionsmenge E_A und fasse die Endausgaben als eigene Transitionen auf, zum Beispiel indem ich für alle Zustände q mit $\rho(q) \neq \varepsilon$ einen zusätzlichen Zustand q_+ definiere und jeweils die Transition $(q, \rho(q), q_+)$ zu E_A hinzu nehme. Zu diesem Automaten A_T konstruiere ich den Präfix (betrachte größte gemeinsame Präfixe von Pfaden zu F oder zu i), der ohne Einschränkung als schlank vorausgesetzt werden kann. Dabei wird die Ausgabe von Pfaden zu $F \cup \{i\}$ betrachtet. Ich erhalte einen äquivalenten Automaten, für den insbesondere gilt, dass Transitionen $(q \xrightarrow{\rho(q)} q_+)$ in Transitionen der Art $(q \xrightarrow{\varepsilon} q_+)$ übergehen, denn T beschreibt eine sequentielle Funktion und der konstruierte Präfix ist schlank. Falls i ein finaler Zustand ist, so gilt $\rho(i) = \varepsilon$, da $|T|$ sequentiell ist. Der zu dem resultierenden Automaten gehörende Transducer ist unter Vernachlässigung der Endausgabe der gesuchte äquivalente sequentielle Transducer zu T . Für alle Zustände $q \neq i$ ist die Präfix-Funktion gleich ε . \square

Satz 3.31. *Der minimale ST zu einer sequentiellen Funktion f besitzt genau so viele Zustände, wie der minimale SST, der f darstellt.*

Beweis: Aus dem minimalen SST zu f können wir mit Lemma 3.30 einen äquivalenten ST konstruieren. Außerdem ist jeder ST auch ein SST. \square

Damit gibt es nach Satz 3.13 und Satz 3.29 auch Funktionen aus $\mathcal{F}_{\lambda ST}$, deren minimaler SST einen Zustand mehr als der minimale λ ST besitzt.

3.5.1 Erweiterung der Algorithmen von Mohri und Béal/Caron

In [16] wird auf die Minimalisierung von λ - p -subsequentiellen Transducern eingegangen. Dies soll kurz vorgestellt werden. Dazu wird jedem λp SST ein λ ST

zugeordnet, der minimalisiert und anschließend in einen λp SST umgewandelt wird. Dieser ist der gesuchte minimale λp SST.

Sei $T = (\Sigma, \Delta, Q, i, F, \lambda, \delta, \sigma, \rho)$, mit der partiellen Funktion $\rho : F \rightarrow (\Delta^*)^p$ ($\rho = (\rho_1, \dots, \rho_p)$), ein λp SST. Für jeden finalen Zustand $q \in F$ sollen die Ausgabewörter $\rho_1(q), \dots, \rho_p(q)$ in steigender lexikographischer Ordnung stehen. T wird ein λ ST, $\Psi(T)$, zugeordnet:

- Erweiterung von Σ durch p neue Symbole $\phi_k, 1 \leq k \leq p$
- Einführung eines allgemeinen Endzustandes $f \notin Q$
- Definition neuer Transitionen von jedem Zustand aus F zu f mit Eingaben ϕ_k und Ausgaben $\rho_k, 1 \leq k \leq p$, entsprechend der Ordnung. Das Wort ϕ_1 ist also die Eingabe der Transition mit der ersten Ausgabe in lexikographischer Ordnung.

Es ist dann

$$\Psi(T) = (\Sigma \cup \{\phi_k\}_{1 \leq k \leq p}, \Delta, Q \cup \{f\}, i, \{f\}, \lambda, \delta', \sigma'), \delta' = \delta \cup \delta_1, \sigma' = \sigma \cup \sigma_1,$$

$$\forall q \in F \forall k \in \{1, \dots, p\} : \delta_1(q, \phi_k) = f, \sigma_1(q, \phi_k) = \rho_k(q).$$

Zu dieser Konstruktion kann man eine inverse Konstruktion Ψ^{-1} definieren, die rückwärts aus einem λ ST obiger Gestalt einen λp SST angibt. Sei \mathcal{F} die Menge der λ STs über $\Sigma \cup \{\phi_k\}_{1 \leq k \leq p}$ und Δ , die nur einen finalen Zustand f besitzen, für die zu f nur Kanten mit Eingaben aus $\{\phi_k\}_{1 \leq k \leq p}$ und Ausgaben, lexikographisch entsprechend der ϕ_k geordnet, führen. Die Menge der λp SSTs über Σ, Δ ist $\mathcal{F}_{\lambda p SST}$. Dann wird mit Ψ oder Ψ^{-1} eine Bijektion zwischen \mathcal{F} und $\mathcal{F}_{\lambda p SST}$ definiert [16].

Satz 3.32. (Mohri [16]) *Sei T ein λp SST, dessen Endausgaben lexikographisch geordnet sind. Der λ ST T' entstehe aus $\Psi(T)$ durch Minimalisierung nach Satz 3.22. Dann ist $\Psi^{-1}(T')$ ein minimaler λp SST zu T [16].*

Wenn wir nun den minimalen p SST zu einem λp SST konstruieren wollen, könnten wir das gleiche Prinzip wie Mohri anwenden und dabei die im Kapitel 3.2 in Vermutung 3.23 beschriebene Charakterisierung verwenden. Abbildung 13 zeigt eine subsequentielle Funktion f , deren minimaler λ SST T_2 einen Zustand weniger besitzt als der zugehörige minimale SST T_1 . Dieses Beispiel ist auch für allgemeine p SSTs anwendbar. Das konkrete p bleibt bei den Konstruktionen zunächst erhalten.

Vermutung 3.33. *Sei T ein p SST, dessen Endausgaben lexikographisch geordnet sind. T' entstehe aus $\Psi(T)$ durch Minimalisierung nach Vermutung 3.23. Dann ist $\Psi^{-1}(T')$ ein minimaler p SST zu T .*

Beweis: Analog zu Mohris Beweis für den vorhergehenden Satz, [16, S. 199] \square

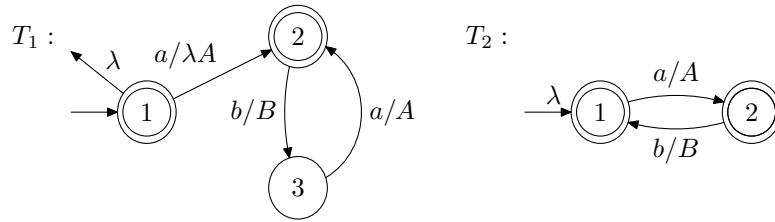


Abbildung 13: f als minimaler SST bzw. minimaler λ -SST

Béal und Carton weisen in [2] darauf hin, dass der beschriebene Algorithmus auch auf SSTs anwendbar ist. Wir wissen, dass dieser Algorithmus eigentlich den minimalen ST zu einem ST berechnet, für den die Einschränkung gilt, dass der größte gemeinsame Präfix von Ausgaben über Pfade beginnend im Initialzustand, gleich dem leeren Wort ist. Im Abschnitt 3.4 habe ich den angegebenen Algorithmus für die Minimalisierung sequentieller und λ -sequentieller Transducer verwendet und somit erweitert. Durch den Satz 3.32 bzw. die Vermutung 3.33 ist der Algorithmus in [2] auch für die Konstruktion des minimalen λp SSTs bzw. p SSTs zu einem gegebenen λp SST bzw. p SST anwendbar.

3.5.2 Choffruts Minimalisierungsverfahren

C. Choffrut [8] betrachtet die Minimalisierung von λ -subsequentiellen Transducern. Er bildet jeden Transducer, der schlank ist, auf den Minimal-Transducer ab. Dabei benutzt er die Relation R_f aus 3.2 und zeigt, dass eine Funktion $f : \Sigma^* \rightarrow \Delta^*$ genau dann λ -subsequentiell ist, wenn der Index von R_f endlich ist. Die Äquivalenzrelation ist also gerade stark genug, um λ -subsequentielle Funktionen zu beschreiben. Bei dem Algorithmus zur Minimalisierung eines λ SSTs wird aus $T = (\Sigma, \Delta, Q, i, F, \lambda, \delta, \sigma, \rho)$ ein Automat $A = (\Delta, Q \dot{\cup} \{s, f\}, s, \{f\}, \sigma \cup \sigma_A)$ mit $\sigma_A(s, \lambda) = i$ konstruiert, so dass für alle Zustände $q \in Q$ gilt: $\sigma_A(q, \rho(q)) = f$. Der Automat A ist also der Ausgabe-Automat von T zusammen mit neuem Initial und Endzustand, deren Transitionen λ und die Ausgabefunktion ρ widerspiegeln. Zu A wird nun der Präfix $P(A) = (\Delta, Q \dot{\cup} \{s, f\}, s, \{f\}, \sigma_{P(A)})$ berechnet und klassische Minimalisierung angewendet. Danach werden die beiden Zustände s, f wieder entfernt.

Genau nach diesem Prinzip könnten wir auch bei der Minimalisierung der SSTs, λp SSTs oder p SSTs vorgehen, so dass die lexikographische Ordnung dann keine Rolle spielt. Um für λp SSTs eine Charakterisierung der Anzahl der Zustände des Minimaltransducers zu erhalten, betrachten wir mit R_f jetzt Mengen von Ausgaben:

Für alle $(u, v) \in D(f) \times D(f)$ gilt genau dann uR_fv , wenn

$$\left(\begin{array}{l} \exists(u', v') \in \Delta^* \times \Delta^* : \\ \forall w \in \Sigma^* : uw \in \text{Dom}(f) \Leftrightarrow vw \in \text{Dom}(f), \\ uw \in \text{Dom}(f) \Rightarrow u'^{-1}f(uw) = v'^{-1}f(vw) \end{array} \right).$$

Dabei ist f eine Relation, die durch einen λp SST dargestellt wird, $u'^{-1}f(uw)$ also eine Menge von Wörtern. Die Anzahl der Zustände eines minimalen λp SST für f ist gleich dem Index von R_f .

Wenn wir uns die Frage nach der Minimalisierung der Transitionsmenge E bzw. des p stellen, müssten wir nach der Minimalisierung gleiche Endausgaben für ein und denselben Zustand streichen. Alle verbliebenen Endausgaben sind notwendig zur Beschreibung der Relation. Die maximale Anzahl solcher Endausgaben für einen Zustand gibt dann das kleinste p , so dass ein p SST oder ein λp SST die betrachtete Relation R_f beschreibt. Die Anzahl der Transitionen ist nach der Minimalisierung kleinst möglich, da den Abschluss eine Minimalisierung endlicher Automaten folgt.

Bei der Komplexität übernehme ich die Betrachtungen von Mohri und erhalte $\mathcal{O}(S + |E| \cdot (P_{max} + |Q| + 1))$, dabei ist S erneut die Summe aller Längen von Wörtern, die über Transitionen ausgegeben werden. Dazu zählen auch die Wörter, die über die Endausgabe-Funktion ρ ausgegeben werden, sie werden zu $|E|$ gezählt. Bei keinem der Algorithmen müssen wir voraussetzen, dass der betrachtete Transducer schlank ist, denn den Schluss der Algorithmen bildet immer eine Minimalisierung endlicher Automaten, deren Ergebnis schlank ist.

3.5.3 Der Suffix-Baum eines Baumes

D. Breslauer gibt in einer Arbeit [5] eine andere Idee für die Konstruktion des Präfixes eines Automaten. Er bezieht sich dabei direkt auf den Artikel von Mohri [13] und betrachtet die Minimalisierung sequentieller Transducer. Breslauer nimmt gleiche Bedingungen wie Mohri an (u. a. $P_T(i) = \varepsilon$), was zu gleichen Problemen wie in 3.2 führt. So muss erneut die Definition des Automaten auf Multimengen erweitert werden. Es soll nun seine eigentliche Idee vorgestellt werden. Sei dazu $T = (\Sigma, \Delta, Q, i, F, \delta, \sigma)$ ein sequentieller Transducer. Breslauer betrachtet den Suffix-Baum eines Baumes. Das ist eine Datenstruktur, die in geeigneter Weise alle Suffixe einer Menge von Wörtern enthält. Level-Probleme über gemeinsame Vorgänger von Knoten sind in konstanter Zeit lösbar (nachdem man den Baum in linearer Zeit zur Anzahl der Knoten überarbeitet hat) und spiegeln die Frage nach dem größten gemeinsamen Suffix zweier Suffixe der betrachteten Wörter wider. Die Vorgehensweise ist nun die folgende. Um für Zustände $q \in Q$ den größten gemeinsamen Präfix $P_T(q)$ von Ausgaben über Pfade zu F zu berechnen, könnte man genauso die Ausgabe $\sigma(\pi)$ über einen ganz bestimmten dieser Pfade von q berechnen und zusätzlich $|P_T(q)|$. Das Wort $P_T(q)$ ist dann der Präfix der Länge

$|P_T(q)|$ von $\sigma(\pi)$. Für die Länge der Präfixe ist ein „Kürzester Pfad-Problem“ zu lösen, was ausführlich in [5, S. 139ff.] beschrieben wird.

Die Komplexität des Algorithmus für die Konstruktion des Präfixes eines Automaten ist $\mathcal{O}(|Q| + |E| + L_{in} \cdot \log(|\Delta|))$, hierbei ist L_{in} die Summe über alle Längen von Ausgabewörtern. Wenn also die Anzahl der Kanten $|E|$ des Automaten klein ist, hat dieser Algorithmus eine bessere Komplexität als die bisher betrachteten. In den tatsächlich vorkommenden Systemen treten allerdings sehr viele Zustandsübergänge auf und P_{max} ist klein, so dass die Algorithmen der anderen Autoren fast linear sind. Ein direkter Komplexitätsvergleich ist nur schwer möglich, da unterschiedliche Parameter verwendet werden [16].

Den Algorithmus von Breslauer kann man auf die Minimalisierung sequentieller und λ -sequentieller Transducer erweitern, indem man die Präfix-Definition abändert (damit der Präfix zu einem äquivalenten Transducer führt) und gegebenenfalls zu λ STs übergeht. Ich möchte darauf nicht näher eingehen. Die Einschränkung $P_T(i) = \varepsilon$ wird im eigentlichen Algorithmus nicht benötigt, ist aber in Breslausers Publikation für die Äquivalenz der betrachteten Transducer notwendig. Man könnte mit seiner Idee auch (λ) - p -subsequentielle Transducer betrachten. Dazu führt man einen neuen Endzustand f ein und verbindet die p möglichen Endausgaben jeweils mit f . Es können höchstens $p \cdot |F|$ Transitionen dazukommen, die Komplexität für die Konstruktion des Präfixes eines p SST ist dann ebenso $\mathcal{O}(|Q| + |E| + L_{in} \cdot \log(|\Delta|))$, wobei für $|E|$ bzw. L_{in} die neu konstruierten Transitionen dazuzählen.

4 Gewichtete Automaten

Hier wird zunächst kurz in das Gebiet der gewichteten Automaten eingeführt. Daraufhin gebe ich eine rationale Funktion an, welche beschränkte Variation besitzt, jedoch nicht subsequentiell ist, und widerlege damit das Theorem 9 von Mohri [15].

Eine wichtige Aufgabe ist die Determinisierung gewichteter Automaten. Mohri gibt in [15] einen solchen Algorithmus an. Allerdings muss dieser für gewichtete Automaten, die eine sequentielle Funktion berechnen, nicht notwendigerweise terminieren. Es wird eine neue Klasse von gewichteten Automaten über \mathbb{R}_+ definiert. Für Automaten dieser Klasse terminiert (wenn der Automat zusätzlich schlank ist) der Algorithmus von Mohri genau dann, wenn sie determinisierbar sind, außerdem ist es entscheidbar, ob sie eine subsequentielle Funktion berechnen.

4.1 Hintergrund

Definition 4.1. Ein *gewichteter Automat* (kurz GA) $T = (\Sigma, Q, I, F, E, \lambda, \rho)$ ist ein 7-Tupel mit:

- Q , endliche Menge von Zuständen
- Eingabe-Alphabet Σ
- $I \subseteq Q, F \subseteq Q$, Mengen von Anfangs- bzw. Endzuständen
- $E \subseteq Q \times \Sigma \times \mathbb{R}_+ \times Q$, endliche Menge von Transitionen
- $\lambda : I \rightarrow \mathbb{R}_+$, Eingangsgewichtsfunktion
- $\rho : F \rightarrow \mathbb{R}_+$, Ausgangsgewichtsfunktion.

Dabei ist $\mathbb{R}_+ = \{r \in \mathbb{R} \mid r \geq 0\}$ sowie $\mathbb{R}_+^\infty = \mathbb{R}_+ \cup \{\infty\}$ mit den kanonischen Operationen $\min, +$ auf \mathbb{R}_+^∞ ($(\mathbb{R}_+^\infty, \min, +, \infty, 0)$ bezeichnet man auch als tropischen Semiring.). Für den gewichteten Automaten T kann man durch:

$$\forall (p, a) \in Q \times \Sigma, \delta(p, a) := \{q \mid \exists r \in \mathbb{R}_+ : (p, a, r, q) \in E\}$$

eine *Zustandsfunktion* $\delta : Q \times \Sigma \rightarrow \mathfrak{P}(Q)$ definieren, die auf $Q \times \Sigma^*$ oder auch $\mathfrak{P}(Q) \times \Sigma^*$ fortgesetzt wird. Die Ausgaben über Transitionen sind nun keine Wörter mehr, sondern Zahlen. Die Funktion $\sigma : E \rightarrow \mathbb{R}_+$, festgelegt durch:

$$\forall t = (p, a, r, q) \in E, \sigma(t) = r,$$

ist die *Ausgabefunktion* zu T . Ein *Pfad* π von $p \in Q$ nach $q \in Q$ der Länge n über dem Wort $a_0 a_1 \cdots a_{n-1}$ ist eine Folge $((q_0, a_0, r_0, q_1), \dots, (q_{n-1}, a_{n-1}, r_{n-1}, q_n))$ von

Transitionen aus E , so dass für alle $i = 0, \dots, (n-1)$ der Zustand q_{i+1} in $\delta(q_i, a_i)$ liegt. Die Ausgabefunktion kann mit $\sigma(\pi) = r_0 + \dots + r_{n-1}$ auf Pfade erweitert werden. Falls q_0 ein Anfangszustand und q_n ein Endzustand ist, so ist π *erfolgreich*, das Wort $a_0 \dots a_{n-1}$ ist dann das zugehörige *akzeptierte* Wort. Die Menge der Pfade über w von einem Zustand p zu einem Zustand q wird mit $p \xrightarrow{w} q$ bezeichnet. Die Funktion θ beschreibt das Minimum der Ausgaben über Pfade zwischen zwei Zuständen über einem konstanten Wort:

$$\theta(p, w, q) = \min\{\sigma(\pi) : \pi \in p \xrightarrow{w} q\}.$$

Das Minimum über der leeren Menge soll hier ∞ sein. Den *Definitionsbereich* $\text{Dom}(T)$ von T bilden die akzeptierten Wörter. Der gewichtete Automat T ordnet nun einen Wert wie folgt zu:

$$\begin{aligned} |T| : \text{Dom}(T) &\rightarrow \mathbb{R}_+ \text{ mit} \\ |T|(w) &= \min\{\lambda(i) + \theta(i, w, f) + \rho(f) \mid i \in I, f \in \delta(i, w) \cap F\}. \end{aligned}$$

Die Funktion $|T|$ ist die von T berechnete Funktion.

Definition 4.2. Ein *kritischer* Pfad π über $w \in \text{Dom}(T)$, der von einem Zustand i zu einem Zustand f führt, ist ein erfolgreicher Pfad für den gilt:

$$\lambda(i) + \sigma(\pi) + \rho(f) = |T|(w).$$

Bemerkung 4.3. Der Begriff „kritisch“ wurde in Anlehnung an die Netzplantechnik der Optimierung ausgewählt. Dort werden Graphen, die Ereignisse und deren Abhängigkeiten widerspiegeln, betrachtet. Die Lösungen der „kritischen Pfad-Methode“ (CPM, Critical path method) heißen kritische Pfade bzw. kritische Ereignisse.

Abbildung 14 ist ein Beispiel für einen GA. Der Begriff *schlank* soll erneut für diejenigen GAs eingeführt werden, die die Eigenschaft haben, dass jeder Zustand auf einem erfolgreichen Pfad liegt. Gewichtete Automaten berechnen formale Potenzreihen (PR), da sie partiell von einem freien Monoid in einen Semiring abbilden [4]. Formale Potenzreihen sind nach dem Theorem von Schützenberger [4] sogar genau dann rational, wenn sie durch einen GA dargestellt werden können. Im Weiteren betrachten wir nur Potenzreihen über dem tropischen Semiring.

Definition 4.4. Ein gewichteter Automat T über Σ ist *eindeutig*, falls es in T für jedes Eingabewort $w \in \Sigma^*$ höchstens einen erfolgreichen Pfad gibt.

4.2 Subsequentielle gewichtete Automaten

Eine Teilmenge der gewichteten Automaten stellen die subsequentiellen GAs dar. Sie werden wie folgt beschrieben.

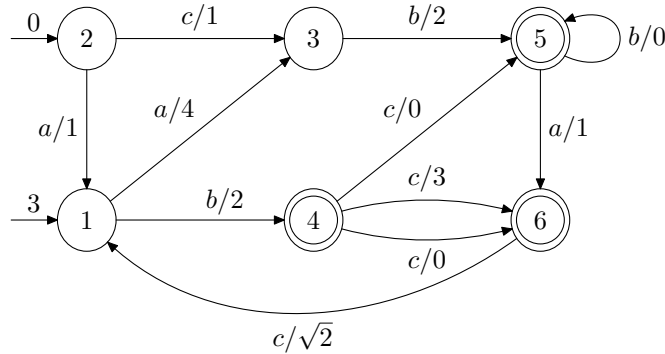


Abbildung 14: Ein schlanker gewichteter Automat

Definition 4.5. Ein gewichteter Automat $T = (\Sigma, Q, I, F, E, \lambda, \rho)$ heißt *subsequentuell*, falls gilt:

- $I = \{i\}$ (es gibt nur einen Anfangszustand) und
- Für $q, q_1, q_2 \in Q, a \in \Sigma, r_1, r_2 \in \mathbb{R}_+$ gilt:
Aus $(q, a, r_1, q_1), (q, a, r_2, q_2) \in E$ folgt $r_1 = r_2, q_1 = q_2$.

Für subsequentielle GAs ist die Zustandsfunktion δ deterministisch definiert, da mit jedem gegebenen Zustand sowie einem Buchstaben a aus Σ der Folgezustand eindeutig festgelegt ist (falls es überhaupt einen gibt). In diesem Falle identifiziere ich die Menge, die genau einen Zustand enthält, mit diesem Zustand. Man kann eine Funktion $\psi : Q \times \Sigma^* \rightarrow \mathbb{R}_+^\infty$ definieren, die die Ausgabe eines Pfades über einem Wort beschreibt, wobei der Pfad in einem bestimmten Zustand startet:

$$\psi(p, w) = \begin{cases} \sigma(\pi), & \text{falls: } \exists \pi, q \in Q : \pi \in p \xrightarrow{w} q \\ \infty, & \text{sonst.} \end{cases}$$

Jeder subsequentielle GA ist eindeutig, es gilt $\theta(p, w, q) = \psi(p, w)$ für alle Zustände $p, q \in Q$ und Wörter $w \in \Sigma^*$. Sie stellen *subsequentielle* Potenzreihen dar. Ein Beispiel ist die Abbildung 15.

Auf Σ^* kann man durch:

$$\forall u, v \in \Sigma^* : d(u, v) = |u| + |v| - 2|u \wedge v|$$

eine Metrik definieren. Dabei ist $|u|$ die Länge, also die Anzahl der Buchstaben, eines Wortes u und $u \wedge v$ der größte gemeinsame Präfix von u und v . Für zwei reelle Zahlen r_1, r_2 bezeichnet $|r_1 - r_2|$ die euklidische Metrik.

Definition 4.6. Eine partielle Funktion $\alpha : \Sigma^* \rightarrow \mathbb{R}_+$ hat *beschränkte Variation*, falls es für alle $k \geq 0$ ein $K \geq 0$ gibt mit:

$$\forall u, v \in \text{Dom}(\alpha) : d(u, v) \leq k \Rightarrow |\alpha(u) - \alpha(v)| \leq K.$$

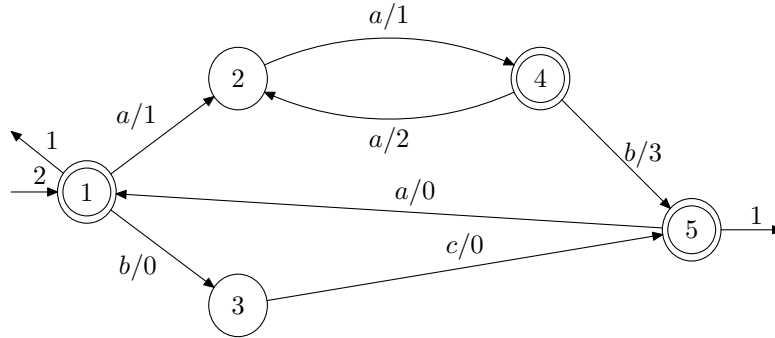


Abbildung 15: Ein subsequentieller GA

In [15] gibt Mohri auf Seite 283 durch sein Theorem 9 eine Charakterisierung derjenigen rationalen Potenzreihen, die subsequentiell sind, also für die es einen subsequentiellen GA gibt, der die betrachtete PR berechnet. Er behauptet, rationale Potenzreihen seien genau dann subsequentiell, wenn sie beschränkte Variation besitzen. Mit Satz 4.9 zeige ich, dass diese Behauptung von Mohri nicht gilt. Mohri nimmt an, dass es zu jedem GA einen äquivalenten gibt, der eindeutig ist. Es existieren jedoch rationale PRs, die beschränkte Variation haben, jedoch durch keinen subsequentiellen GA realisiert werden können. Dazu gebe ich eine spezielle Funktion an, welche durch keinen eindeutigen gewichteten Automaten darstellbar ist, deren Variation jedoch beschränkt ist. Für rationale Potenzreihen, die subsequentiell sind, folgt zwar die Eigenschaft der beschränkten Variation, die Rückrichtung gilt allerdings nicht. Die Idee zum Beweis des folgenden Lemmas stammt von D. Kirsten (Dresden).

Lemma 4.7. *Es gibt eine rationale Potenzreihe, die durch keinen eindeutigen gewichteten Automaten berechnet wird.*

Beweis: Sei $\Sigma = \{a, b\}$. Wir definieren $f : \Sigma^* \rightarrow \mathbb{R}_+$ durch:

$$w \in \Sigma^* : f(w) := \min\{|w|_a, |w|_b\}.$$

Dabei bezeichne $|w|_a$ die Anzahl der Vorkommnisse von $a \in \Sigma$ in w . Abbildung 16 stellt einen GA T_f dar, der f berechnet, somit ist f rational.

Angenommen, die definierte Funktion f ließe sich durch einen eindeutigen GA $S = (\Sigma, Q, I, F, E, \lambda, \rho)$ mit n Zuständen berechnen. Wir betrachten das Eingabewort $w = a^n b^n$ mit zugehörigem erfolgreichem Pfad π . Da S nur n Zustände besitzt, lässt sich nun w wie folgt zerlegen: $w = a^i a^j a^k b^x b^y b^z$ mit $i + j + k = x + y + z = n$ und $j \geq 1, y \geq 1$, sowie $i, j, k, x, y, z \in \mathbb{N}$, so dass π bei a^j bzw. b^y jeweils einen Zyklus durchläuft. Die Teile von π seien entsprechend $\pi = \pi_{a_i} \pi_{a_j} \pi_{a_k} \pi_{b_x} \pi_{b_y} \pi_{b_z}$. Dann darf sich, wegen der Eindeutigkeit der erfolgreichen Pfade, bei zweimaligem Durchlaufen der Zyklen die Ausgabe durch

S nicht erhöhen:

$$S(a^i a^{2j} a^k b^x b^y b^z) = S(a^i a^j a^k b^x b^y b^z) = n, \text{ also: } \sigma(\pi_{a_j}) = 0$$

$$S(a^i a^j a^k b^x b^{2y} b^z) = S(a^i a^j a^k b^x b^y b^z) = n, \text{ also: } \sigma(\pi_{b_y}) = 0.$$

Werden jedoch beide Zyklen zweifach durchlaufen, so muss sich die Ausgabe erhöhen:

$$S(a^i a^{2j} a^k b^x b^{2y} b^z) = n + 2 \cdot \min\{j, y\}, \text{ also: } \sigma(\pi_{a_j}) + \sigma(\pi_{b_y}) \neq 0.$$

Wegen der Eindeutigkeit der erfolgreichen Pfade ist dies ein Widerspruch. \square

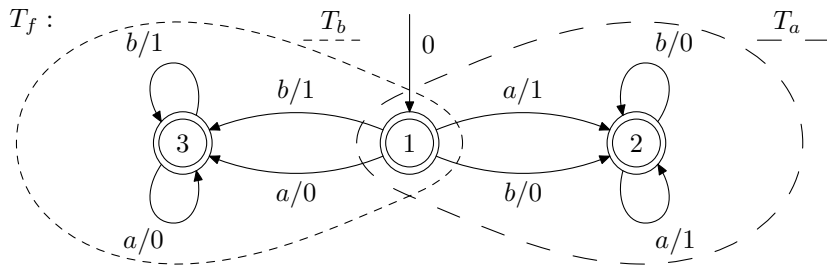


Abbildung 16: T_f berechnet die Funktion f .

Lemma 4.8. Die angegebene Funktion f aus Lemma 4.7 hat beschränkte Variation.

Beweis: Angenommen, f habe nicht beschränkte Variation. Das heißt, es gibt ein $k \geq 0$ und zwei Folgen $(u_i), (v_i), i \in \mathbb{N}$, aus Σ^* mit $d(u_i, v_i) \leq k$ und

$$|f(u_i) - f(v_i)| \xrightarrow{i \rightarrow \infty} \infty.$$

Man kann den in Abbildung 16 angegebenen GA betrachten, der f berechnet. Dieser setzt sich aus den beiden GAs T_a bzw. T_b zusammen, die für Wörter w jeweils $|w|_a$ bzw. $|w|_b$ zählen und deterministisch sind. Dann kann man leicht beweisen, dass f beschränkte Variation besitzt. Die Definition der Funktion f ist aber zunächst an keinen Automaten gebunden, deshalb möchte ich den Beweis automatenunabhängig führen.

Seien u, v zwei Wörter über $\{a, b\}$ mit $w = u \wedge v$, also:

$$\exists u', v' \in \Sigma^* : u = wu', v = wv'.$$

Wir zeigen: $|f(u) - f(v)| \leq d(u, v)$. Dazu nehme ich im ersten Fall an, dass in u und v durch f entweder nur Buchstaben a oder nur Buchstaben b gezählt werden, ohne Einschränkung zum Beispiel a . Es ist:

$$\begin{aligned} |f(u) - f(v)| &= ||u|_a - |v|_a| = ||u'|_a - |v'|_a| \\ &\leq |u'|_a + |v'|_a \leq |u'| + |v'| = d(u, v). \end{aligned}$$

Werden durch die Funktion f in u und v im zweiten Fall unterschiedliche Buchstaben gezählt, zum Beispiel in u das a und in v das b , so gilt:

$$|w|_a + |u'|_a \leq |w|_b + |u'|_b \quad (6)$$

$$|w|_b + |v'|_b \leq |w|_a + |v'|_a. \quad (7)$$

Dann kann ich die Differenz der Funktionswerte von u und v mit einer Fallunterscheidung aber ebenfalls abschätzen. Angenommen, $f(u) \geq f(v)$, dann ist:

$$\begin{aligned} |f(u) - f(v)| &= ||w|_a + |u'|_a - |w|_b - |v'|_b| = |w|_a + |u'|_a - |w|_b - |v'|_b \\ &\stackrel{6}{\leq} |w|_b + |u'|_b - |w|_b - |v'|_b = |u'|_b - |v'|_b \\ &\leq |u'| + |v'| = d(u, v). \end{aligned}$$

Andernfalls ($f(u) \leq f(v)$) gilt:

$$\begin{aligned} |f(u) - f(v)| &= ||w|_a + |u'|_a - |w|_b - |v'|_b| = |w|_b + |v'|_b - |w|_a - |u'|_a \\ &\stackrel{7}{\leq} |w|_a + |v'|_a - |w|_a - |u'|_a = |v'|_a - |u'|_a \\ &\leq |u'| + |v'| = d(u, v). \end{aligned}$$

Wenn ich nun $K := k$ setze, so erfüllt f die Bedingungen der Definition 4.6 und besitzt somit beschränkte Variation. \square

Satz 4.9. *Es gibt eine rationale Potenzreihe mit beschränkter Variation, die nicht subsequentiell ist.*

Beweis: Nach Lemma 4.7 und 4.8 ist f (Abbildung 16) eine rationale PR mit beschränkter Variation, die durch keinen eindeutigen GA darstellbar ist. Jeder subsequentielle GA ist auch eindeutig. Deshalb kann f nicht subsequentiell sein. \square

Im Weiteren werde ich eine Klasse von rationalen Potenzreihen angeben, für die die Behauptung von Mohri gilt. Diese Potenzreihen sind also genau dann subsequentiell wenn sie beschränkte Variation besitzen. Dazu gehe ich zunächst auf die Determinisierung von gewichteten Automaten ein und stelle in diesem Zusammenhang einige Resultate von Mohri dar [15]. Sie werden in einem Korollar zusammengefasst.

4.3 Ein Algorithmus zur Determinisierung von gewichteten Automaten

In [15, S. 285ff.] geht es um das Problem, wann es zu einem gewichteten Automaten einen äquivalenten geben kann, der subsequentiell ist und wie dieser möglicherweise berechnet werden könnte. Eine wichtige Eigenschaft dabei ist die folgende.

Definition 4.10. Sei $T = (\Sigma, Q, I, F, E, \lambda, \rho)$ ein GA mit zugehöriger Zustandsfunktion δ . Zwei Zustände $p, q \in Q$ heißen *Zwillinge*, falls für alle möglichen Wörter $u, v \in \Sigma^*$ gilt:

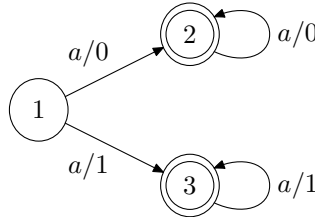
$$[p, q \in \delta(I, u), p \in \delta(p, v), q \in \delta(q, v)] \implies [\theta(p, v, p) = \theta(q, v, q)].$$

Ein gewichteter Automat T besitzt die *Zwillingeigenschaft*, falls je zwei Zustände von T Zwillinge sind.

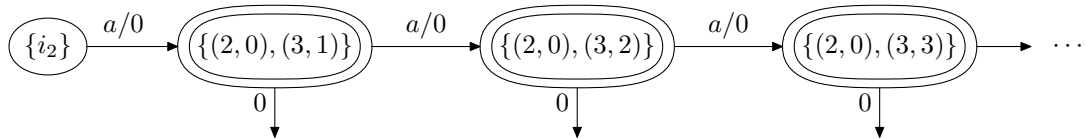
Ohne Einschränkung kann man bei obiger Definition annehmen, dass die Zustände p und q verschieden sind.

Auf Seite 285 in [15] wird ein konkreter Algorithmus, POWER SERIES DETERMINIZATION, für eine Determinisierung eines GA angegeben. Dieser hat als Eingabe einen GA $\tau_1 = (Q_1, \Sigma, I_1, F_1, E_1, \lambda_1, \rho_1)$ und konstruiert, falls er terminiert, einen äquivalenten subsequentiellen GA $\tau_2 = (Q_2, \Sigma, \{i_2\}, F_2, E_2, \lambda_2, \rho_2)$. Ich werde den Algorithmus von Mohri in dieser Arbeit PSDET nennen. Abbildung 17 stellt den Pseudocode des Algorithmus dar.

Es gibt gewichtete Automaten T , die eine subsequentielle Funktion berechnen, aber für die PSDET(T) nicht terminiert:



Der Algorithmus würde unendlich viele Zustände erzeugen und somit nicht anhalten:



Die entscheidende Frage ist, welche Klasse von gewichteten Automaten durch den Algorithmus PSDET determinisiert werden kann. Darauf gehe ich im zweiten Teil des nächsten Abschnittes ein. Mohri zeigt, wenn der gegebene GA die Zwillingeigenschaft besitzt, so hält PSDET an, außerdem ist das Ergebnis eines terminierenden Ablaufes des Algorithmus ein subsequentieller GA [15, S. 287ff.].

Bemerkung 4.11. In Abbildung 17 werden folgende Notation verwendet:

- $\Gamma(q_2, a) = \{(q, x) \in q_2 \mid (q, a, r, p) \in E_1\}$

- $\gamma(q_2, a) = \{(q, x, t) \in q_2 \times E_1 \mid t = (q, a, r, p) \in E_1\}$
- $\nu(q_2, a) = \{q' \mid \exists (q, x) \in q_2, \exists t = (q, a, r, q') \in E_1\}$.

Der Zustand q_2 ist der gerade betrachtete Zustand des neuen GA τ_2 . Er enthält Paar von Zuständen aus τ_1 und Zahlen. Es wird eine Schlange Q verwendet, die die noch zu untersuchenden Zustände des neuen GA enthält. Die Zustands- und Ausgabefunktionen der GAs τ_1 und τ_2 sind jeweils δ_1 und δ_2 bzw. σ_1 und σ_2 . Die Funktion der deterministischen Ausgabe für τ_2 ist ψ_2 . Die *Restausgabe* zu einem Zustand $q \in Q_1$ und einem Wort $w \in \Sigma^*$ ist definiert durch:

$$c(q, w) = \min_{i_1 \in I_1} \{\lambda_1(i_1) + \theta_1(i_1, w, q) - \psi_2(i_2, w) - \lambda_2\}.$$

Wie der angegebene Algorithmus intuitiv arbeitet, wird ausführlich in der Publikation von Mohri [15, S. 185f.] beschrieben.

PSDET(τ_1)

```

1   $F_2 \leftarrow \emptyset$ 
2   $\lambda_2 \leftarrow \min\{\lambda_1(i) \mid i \in I_1\}$ 
3   $i_2 \leftarrow \bigcup_{i \in I_1} \{(i, \lambda(i) - \lambda_2)\}$ 
4   $Q \leftarrow [i_2], Q_2 \leftarrow \{i_2\}, E_2 \leftarrow \emptyset$ 
5  while  $Q \neq \emptyset$ 
6  do  $q_2 \leftarrow \text{head}(Q)$ 
7      if (es gibt  $(q, x) \in q_2$  mit  $q \in F_1$ )
8      then  $F_2 \leftarrow F_2 \cup \{q_2\}$ 
9           $\rho_2(q_2) \leftarrow \min\{x + \rho_1(q) \mid q \in F_1, (q, x) \in q_2\}$ 
10     for (jedes  $a$  mit  $\Gamma(q_2, a) \neq \emptyset$ )
11     do  $\psi_2(q_2, a) \leftarrow \min\{[x + \min_{(q,a,\sigma_1(t),n_1(t)) \in E_1} \sigma_1(t)] \mid (q, x) \in \Gamma(q_2, a)\}$ 
12          $\delta_2(q_2, a) \leftarrow \bigcup_{q' \in \nu(q_2, a)} \{(q', \min_{(q,x,t) \in \gamma(q_2, a), n_1(t)=q'} (x + \sigma_1(t) - \sigma_2(q_2, a))\}$ 
13         if ( $\delta_2(q_2, a)$  ist neuer Zustand)
14         then (hänge  $\delta_2(q_2, a)$  an  $Q$  an)
15      $Q \leftarrow \text{tail}(Q), Q_2 \leftarrow Q_2 \cup \{\delta_2(q_2, a)\}, E_2 \leftarrow E_2 \cup \{(q_2, a, \psi_2(q_2, a), \delta_2(q_2, a))\}$ 

```

Abbildung 17: Der Algorithmus PSDET.

Die folgenden Ausführungen von Mohri werden von mir verwendet:

Lemma 4.12. (Mohri [15]) *Sei τ ein GA, so dass PSDET(τ) terminiert. Dann ist PSDET(τ) ein äquivalenter subsequentieller GA zu τ .*

Lemma 4.13. (Mohri [15]) *Sei τ ein GA, der die Zwillings-eigenschaft besitzt. Dann terminiert PSDET(τ).*

Lemma 4.14. (erster Teil des Beweises zu Theorem 9 [15]) *Sei f eine rationale Potenzreihe, die subsequentiell ist. Dann besitzt f beschränkte Variation.*

Im Beweis seiner Behauptung in Theorem 9 [15, S. 283] nimmt Mohri fälschlicherweise an, dass es zu jedem gewichteten Automaten einen äquivalenten eindeutigen GA gibt und zeigt dann, dass dieser, falls er beschränkte Variation besitzt, auch die Zwillingsseigenschaft hat. Die Aussage, dass aus der Eindeutigkeit eines GA, welcher beschränkte Variation hat, die Zwillingsseigenschaft folgt, ist dabei korrekt und führt zu diesem Lemma:

Lemma 4.15. *(Teile des Beweises zu Theorem 9 [15]) Sei τ ein eindeutiger und schlanker GA, der eine Funktion mit beschränkter Variation berechnet. Dann hat τ die Zwillingsseigenschaft.*

Wenn ein gewichteter Automat die Zwillingsseigenschaft hat, so ist er durch den Algorithmus PSDet determinisierbar und berechnet somit eine subsequentielle Funktion. Die Lemmata 4.15, 4.13 und 4.12 lassen sich also in dem folgenden Korollar zusammenfassen:

Korollar 4.16. *Sei τ ein eindeutiger GA, der eine Funktion mit beschränkter Variation berechnet. Dann ist $|\tau|$ subsequentiell.*

4.4 Pfadneutrale gewichtete Automaten

Ich möchte eine neue Klasse von GAs beschreiben. Dazu führe ich die folgende Definition eines *pfadneutralen* GA ein und zeige, dass es zu jedem solchen pfadneutralen gewichteten Automaten einen äquivalenten eindeutigen GA gibt.

Definition 4.17. Ein GA heißt *pfadneutral*, falls jeder erfolgreiche Pfad kritisch ist. Eine formale PR f heißt dann *pfadneutral darstellbar*, wenn es einen pfadneutralen GA gibt, der f berechnet.

Bemerkung 4.18. In diesem Teil verwende ich auch schlanke gewichtete Automaten. Zu jedem GA T gibt es einen äquivalenten GA, der schlank ist. Bei der Konstruktion „streicht“ man effektiv bestimmte Zustände, so dass Eigenschaften von T , zum Beispiel, dass er eindeutig oder pfadneutral ist, bei der Konstruktion erhalten bleiben.

Es gilt der folgende Satz [3, S. 111f.]:

Satz 4.19. *(Eilenberg) Seien X und Y zwei Alphabete und $\alpha : X^* \rightarrow Y^*$ ein Homomorphismus. Zu jeder rationalen Teilmenge $A \subseteq X^*$ existiert eine rationale Menge $B \subseteq A$, so dass α diese Menge B bijektiv auf $\alpha(A)$ abbildet.*

Lemma 4.20. *Zu jedem gewichteten Automaten $\tau = (\Sigma, Q, I, F, E, \lambda, \rho)$ mit $\varepsilon \notin \text{Dom}(\tau)$ existiert ein äquivalenter GA $\tau' = (\Sigma, Q', \{i\}, \{f\}, E', \lambda', \rho')$, wobei gilt: $\lambda'(i) = 0$ und $\rho'(f) = 0$.*

Bemerkung 4.21. Dieses Resultat ist wohlbekannt (siehe z. B. in [9]). Man kann zeigen, dass es für den Beweis des Lemmas 4.20 eine Konstruktion gibt, die die Eigenschaft, dass der gewichtete Automat pfadneutral ist, erhält und die außerdem angewandt auf einen GA T , der ε im Definitionsbereich hat, einen GA T' angibt mit $|T'| = |T|_{/\text{Dom}(T) \setminus \{\varepsilon\}}$.

Satz 4.22. *Zu jedem pfadneutralen GA existiert ein äquivalenter schlanker GA, der eindeutig ist.*

Beweis: Sei $\tau = (\Sigma, Q, I, F, E, \lambda, \rho)$ ein pfadneutraler GA. Zunächst betrachte ich den Fall, wie im obigen Lemma, dass $|\tau|(\varepsilon)$ nicht definiert ist. Ohne Einschränkung können wir also annehmen: $I = \{i\}, F = \{f\}, \lambda(i) = 0, \rho(f) = 0$. Ich definiere

$$R := \{r \in \mathbb{R}_+ \mid \exists p, q \in Q, a \in \Sigma \text{ mit } (p, a, r, q) \in E\}$$

als Menge der Ausgaben über Transitionen aus E . Dann ist E ein Alphabet über $Q \times \Sigma \times R \times Q$. Einen Homomorphismus $\alpha : E^* \rightarrow \Sigma^*$ lege ich durch

$$\alpha(p, a, r, q) := a, \quad \alpha(\varepsilon) := \varepsilon$$

fest (natürliche Fortsetzung auf der Menge der Wörter über E). Weiterhin sei

$$S := \{(p, a, r, p')(q, a', r', q') \in E^2 \mid p' \neq q\} \subseteq E^*$$

die Menge aller Paare aus E , die nicht aufeinander folgen. Dann ist S endlich und somit von einem DEA über E erkennbar. Die Menge der erfolgreichen Pfade in τ ist

$$P := [(\{i\} \times \Sigma \times R \times Q)E^* \cap E^*(Q \times \Sigma \times R \times \{f\})] \setminus E^*SE^*,$$

also ebenfalls erkennbar über E . Dann gilt

$$\text{Dom}(\tau) = \alpha(P). \tag{8}$$

Nach dem Zerlegungssatz von Eilenberg, hier der Satz 4.19, existiert eine rationale Sprache $K \subseteq P$, so dass α diese Sprache K bijektiv auf $\alpha(P) \stackrel{(8)}{=} \text{Dom}(\tau)$ abbildet. Ich kann also von der Menge der erfolgreichen Pfade von τ eine rationale Menge derart auswählen, dass es zu jedem Eingabewort genau einen Pfad gibt. Sei $A = (E, Q_A, i_A, F_A, E_A)$ ein DEA, welcher K erkennt. Ich definiere einen gewichteten Automaten $\tau' = (\Sigma, Q_A, \{i_A\}, F_A, E', 0, 0)$ durch:

$$E' := \{(p, a, r, q) \mid \exists z_1, z_2 \in Q_A \text{ mit } (p, (z_1, a, r, z_2), q) \in E_A\}. \tag{9}$$

Der GA τ' ist eindeutig, da τ nur kritische Pfade enthält. Jetzt zeige ich, dass τ' ein GA äquivalent zu τ ist.

Sei dazu $a = a_1 a_2 \dots a_n$ aus $\text{Dom}(\tau)$. Dann gibt es ein Element

$$\pi_\tau = (z_0, a_1, r_1, z_1)(z_1, a_2, r_2, z_2) \dots (z_{n-1}, a_n, r_n, z_n) \in K$$

mit $\alpha(\pi) = a$ und $|\tau(a)| = \sum_{i=1}^n r_i$. Damit existiert auch ein Pfad

$$\pi_A = (q_0, (z_0, a_1, r_1, z_1), q_1)(q_1, (z_1, a_2, r_2, z_2), q_2) \dots (q_{n-1}, (z_{n-1}, a_n, r_n, z_n), q_n)$$

in A über π_τ , da der Automat A die Menge K erkennt. Wegen (9) ist

$$\pi_{\tau'} = (q_0, a_1, r_1, q_1)(q_1, a_2, r_2, q_2) \dots (q_{n-1}, a_n, r_n, q_n)$$

ein Pfad in τ' und somit $a \in \text{Dom}(\tau')$ und (da τ' eindeutig ist) $|\tau'(a)| = \sum_{i=1}^n r_i = |\tau|(a)$.

Wenn umgekehrt $a = a_1 a_2 \dots a_n$ aus $\text{Dom}(\tau')$ ist, so existiert ein Pfad

$$\pi_{\tau'} = (q_0, a_1, r_1, q_1)(q_1, a_2, r_2, q_2) \dots (q_{n-1}, a_n, r_n, q_n)$$

in τ' , das Wort a wird durch τ' auf $\sum_{i=1}^n r_i$ abgebildet. Die Transitionen aus τ' sind über (9) entstanden, deshalb gibt es Zustände z_0, z_1, \dots, z_n aus Q und einen Pfad

$$\pi_A = (q_0, (z_0, a_1, r_1, z_1), q_1)(q_1, (z_1, a_2, r_2, z_2), q_2) \dots (q_{n-1}, (z_{n-1}, a_n, r_n, z_n), q_n)$$

in A für $\pi_{\tau'}$. So ist

$$(z_0, a_1, r_1, z_1)(z_1, a_2, r_2, z_2) \dots (z_{n-1}, a_n, r_n, z_n)$$

aus K und damit ein erfolgreicher Pfad aus τ . Es gilt $a \in \text{Dom}(\tau)$ und (da τ pfadneutral ist) $|\tau|(a) = \sum_{i=1}^n r_i = |\tau'(a)$.

Setze ich voraus, dass der Automat A schlank ist, so ist auch τ' schlank. Für den Fall, dass ε im Definitionsbereich von τ ist, konstruiere ich analog zum Lemma 4.20 und der darauf folgenden Bemerkung aus τ einen GA τ_1 mit:

$$\text{Dom}(\tau) \setminus \{\varepsilon\} = \text{Dom}(\tau_1) \text{ und } |\tau|(w) = |\tau_1|(w) \text{ für alle } w \neq \varepsilon$$

Nach dem Beweis für den ersten Fall und erneuter Anwendung des Lemmas 4.20 konstruiere ich aus τ_1 einen äquivalenten GA $\tau_2 = (Q_2, \Sigma, \{i_2\}, \{f_2\}, E_2, 0, 0)$, der eindeutig ist. Der GA $\tau' := (Q_2, \Sigma, \{i_3\}, \{f_2, i_3\}, E_2, \lambda, 0)$ mit $\lambda(i_3) = |\tau|(\varepsilon)$ ist dann ein äquivalenter eindeutiger gewichteter Automat zu τ . \square

Ich zeige nun, dass für die Klasse der pfadneutral darstellbaren Funktionen die Bedingung der beschränkten Variation mit der Eigenschaft, dass eine PR subsequentiell ist, gleichbedeutend ist.

Satz 4.23. *Die beiden Aussagen sind äquivalent:*

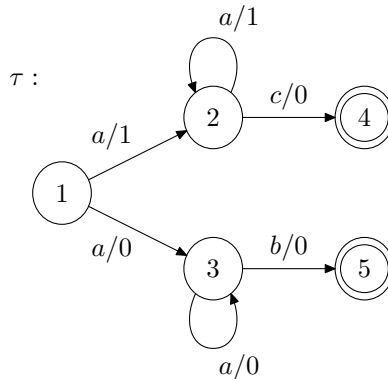
(1) *f ist subsequentiell.*

(2) *f ist pfadneutral darstellbar und besitzt beschränkte Variation.*

Beweis: (1) \Rightarrow (2): Die Funktion f ist subsequentiell, deshalb gibt es einen subsequentiellen GA, der f berechnet. Jeder subsequentielle GA ist aber auch pfadneutral. Außerdem besitzt f nach Lemma 4.14 beschränkte Variation.

(2) \Rightarrow (1): Da f pfadneutral darstellbar ist, gibt es nach Satz 4.22 einen eindeutigen GA, der f berechnet. Jede Funktion mit beschränkter Variation, die durch einen eindeutigen gewichteten Automaten realisiert wird, ist aber subsequentiell nach dem Korollar 4.16. \square

Jeder eindeutige GA ist pfadneutral. In der Abbildung 18 betrachte ich rationale Potenzreihen, die durch gewichtete Automaten berechnet werden. Die Menge F_{bV} bezeichne diejenigen PRs, welche beschränkte Variation haben, F_{sub} die subsequentiellen Funktionen, F_{ei} diejenigen, welche durch einen eindeutigen GA berechenbar sind, und F_{pn} die pfadneutral darstellbaren PR. Nach dem Satz 4.22 fallen F_{pn} und F_{eind} zusammen. Es gibt eine pfadneutral darstellbare Funktion, deren Variation nicht beschränkt ist:



Offensichtlich ist der GA τ pfadneutral. Ich definiere zwei Folgen von Wörtern über $\{a, b, c\}$:

$$(u_i) = (a^i c), (v_i) = (a^i b) \quad i := 1, 2, 3, \dots$$

Dann ist $d(u_i, v_i) = 2$, aber $||\tau|(u_i) - |\tau|(v_i)|| = i \xrightarrow{i \rightarrow \infty} \infty$. Die Funktion $|\tau|$ ist also nicht in F_{bV} .

Ferner gibt es nach den Lemmata 4.7 und 4.8 rationale Potenzreihen aus F_{bV} , die nicht eindeutig darstellbar sind. Ich kann nun durch den Satz 4.23 dieses Diagramm angeben:

Ich möchte untersuchen, für welche Klasse von gewichteten Automaten, die subsequentielle Funktionen beschreiben, der Algorithmus PSDDET wirklich anhält, also

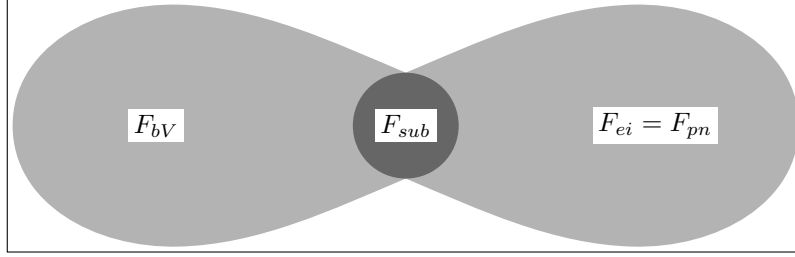


Abbildung 18: Beziehungen zwischen rationalen PR.

der Algorithmus „optimal“ in dem Sinne arbeitet, dass er für determinisierbare GAs einen subsequentiellen äquivalenten GA angibt. Dazu beweise ich die beiden nächsten Sätze und kann dann zeigen, dass F_{pn} eine solche Klasse beschreibt.

Satz 4.24. *Sei τ ein pfadneutraler und schlanker GA mit der Eigenschaft, dass $\text{PSDET}(\tau)$ terminiert. Dann besitzt τ die Zwillingseigenschaft.*

Beweis: Sei $\tau = (\Sigma, Q, I, F, E, \lambda, \rho)$ mit der Zustandsfunktion δ . Angenommen, τ habe nicht die Zwillingseigenschaft. Dann existieren $q_1, q_2 \in Q$ sowie $u, v \in \Sigma^*$ und $i_1, i_2 \in I$ mit:

$$\begin{aligned} \delta(i_1, u) \ni q_1, \delta(q_1, v) \ni q_1 & : i_1 \xrightarrow{u} q_1 \xrightarrow{v} q_1 \\ \delta(i_2, u) \ni q_2, \delta(q_2, v) \ni q_2 & : i_2 \xrightarrow{u} q_2 \xrightarrow{v} q_2 \\ \theta(q_1, v, q_1) & \neq \theta(q_2, v, q_2). \end{aligned}$$

$\text{PSDET}(\tau)$ sei der subsequentielle GA $\tau' = (\Sigma, Q', \{i'\}, F', \lambda', \rho')$ mit der Funktion $\psi' : Q' \times \Sigma^* \rightarrow \mathbb{R}_0^\infty$ für die Ausgabe (Bemerkung nach Definition 4.5) sowie der Zustandsfunktion δ' . Ich betrachte die gewichteten Zustandsmengen $\delta'(i', uv^k)$, $k \in \mathbb{N}$, welche durch PSDET konstruiert werden (Abbildung 17). Die Menge $\delta'(i', uv^k)$ enthält die Tupel $(q_1, c(q_1, uv^k))$ und $(q_2, c(q_2, uv^k))$ (Bemerkung 4.11). Da τ pfadneutral und schlank ist, gilt dann für alle $k \in \mathbb{N}$:

$$\begin{aligned} c(q_1, uv^k) &= \min_{j \in I} \{ \lambda(j) + \theta(j, u, q_1) \} + k\theta(q_1, v, q_1) - \psi'(i', uv^k) - \lambda' \\ c(q_2, uv^k) &= \min_{j \in I} \{ \lambda(j) + \theta(j, u, q_2) \} + k\theta(q_2, v, q_2) - \psi'(i', uv^k) - \lambda' . \end{aligned}$$

Seien λ_1 und θ_1 definiert durch:

$$\begin{aligned} \lambda_1 &:= \min_{j \in I} \{ \lambda(j) + \theta(j, u, q_1) \} - \min_{j \in I} \{ \lambda(j) + \theta(j, u, q_2) \} \\ \theta_1 &:= \theta(q_1, v, q_1) - \theta(q_2, v, q_2). \end{aligned}$$

$$\implies \forall k \in \mathbb{N} : c(q_1, uv^k) - c(q_2, uv^k) = \lambda_1 + k\theta_1.$$

Da $\theta_1 \neq 0$ ist, sind unendlich viele Zustandsmengen $\delta'(i', uv^k)$, ($k \in \mathbb{N}$) verschieden, ein Widerspruch zur Annahme, dass PSDET bei Eingabe von τ terminiert.

□

Satz 4.25. *Sei τ ein pfadneutraler und schlanker GA, der eine subsequentielle Funktion berechnet. Dann besitzt τ die Zwillingsseigenschaft.*

Beweis: Seien $q_1, q_2 \in Q, i_1, i_2 \in I$ und $u, v \in \Sigma^*$ mit:

$$q_1 \in \delta(i_1, u), q_2 \in \delta(i_2, u), q_1 \in \delta(q_1, v), q_2 \in \delta(q_2, v) .$$

Da τ schlank ist und beschränkte Variation (Lemma 4.14) besitzt, existieren $w_1, w_2 \in \Sigma^*$ mit:

$$f_1 \in \delta(q_1, w_1) \cap F \text{ und } f_2 \in \delta(q_2, w_2) \cap F$$

$$\exists K \geq 0 : [\forall k \geq 0 : ||\tau|(uv^k w_1) - |\tau|(uv^k w_2)| \leq K].$$

Da τ pfadneutral und schlank ist, nehmen alle Pfade, die zwischen zwei Zuständen das gleiche Wort einlesen, zum Beispiel $\pi \in q_1 \xrightarrow{v} q_1$ die gleiche minimale Ausgabe an, sind also konstant.

$$\begin{aligned} |\tau|(uv^k w_1) &= \lambda(i_1) + \theta(i_1, u, q_1) + \theta(q_1, w_1, f_1) + k \cdot \theta(q_1, v, q_1) + \rho(f_1) \\ &=: k \cdot \theta(q_1, v, q_1) + C_1 \\ |\tau|(uv^k w_2) &= \lambda(i_2) + \theta(i_2, u, q_2) + \theta(q_2, w_2, f_2) + k \cdot \theta(q_2, v, q_2) + \rho(f_2) \\ &=: k \cdot \theta(q_2, v, q_2) + C_2 \end{aligned}$$

$$\implies \forall k \geq 0 : |C_1 - C_2 + k(\theta(q_1, v, q_1) - \theta(q_2, v, q_2))| \leq K$$

$$\implies \theta(q_1, v, q_1) - \theta(q_2, v, q_2) = 0$$

□

Satz 4.26. *Sei τ ein pfadneutraler und schlanker gewichteter Automat. Dann sind äquivalent:*

- (1) $|\tau|$ ist subsequentiell.
- (2) $\text{PSDET}(\tau)$ terminiert.
- (3) τ besitzt die Zwillingsseigenschaft.
- (4) $|\tau|$ hat beschränkte Variation.

Beweis: (2) \implies (3): Nach Satz 4.24 folgt aus der Terminierung von PSDET , angewandt auf einen gewichteten Automaten τ , die Zwillingsseigenschaft für τ , falls dieser schlank und pfadneutral ist.

(3) \implies (2): Für jeden GA, der die Zwillingsseigenschaft hat, terminiert PSDET nach dem Lemma 4.13.

(1) \Leftrightarrow (4): Diese Äquivalenz beschreibt gerade Satz 4.23.

(2) \implies (1): Falls $\text{PSDET}(\tau)$ anhält berechnet τ eine subsequentielle Funktion (Lemma 4.12).

(1) \implies (3): Hier verwende ich den eben bewiesenen Satz 4.25. □

Der angegebene Algorithmus arbeitet für schlanke GAs, die pfadneutral sind, bestmöglich in dem Sinne, dass er, wenn die dargestellte Funktion subsequentiell ist, auch wirklich terminiert und einen deterministischen GA angibt, der die Funktion berechnet.

Im Folgenden werde ich auf die Frage nach der Entscheidbarkeit, ob eine PR subsequentiell ist, eingehen. Dazu zeige ich, dass es für pfadneutrale und schlanke GAs entscheidbar ist, ob sie die Zwillingsseigenschaft besitzen und benötige dazu ein Lemma von Mohri [15, S. 289, Lemma 1]. Dieses ist eine Eigenschaft gewichteter Automaten, verwendet aber in der eigentlichen Aussage sowie im Beweis lediglich den zugrunde liegenden endlichen Automaten und ist daher ebenso auf endliche Automaten anwendbar. Ich gebe das allgemeine Resultat an:

Lemma 4.27. (Mohri [15]) *Sei $A = (Q, \Sigma, I, F, \delta)$ ein NNEA derart, dass es Zustände $p, p', q, q' \in Q$, ein Wort $w \in \Sigma^*$ und Pfade $\pi \in p \xrightarrow{w} q$, $\pi' \in p' \xrightarrow{w} q'$ gibt mit der Eigenschaft, dass die Länge von w größer als $|Q|^2$ ist. Dann existieren Wörter $u_1, u_2, u_3 \in \Sigma^*$, sowie Zustände $z, z' \in Q$, so dass sich die Pfade π und π' wie folgt zerlegen lassen:*

$$w = u_1 u_2 u_3, \quad \pi \in p \xrightarrow{u_1} z \xrightarrow{u_2} z' \xrightarrow{u_3} q, \quad \pi' \in p' \xrightarrow{u_1} z' \xrightarrow{u_2} z' \xrightarrow{u_3} q'.$$

Dabei ist $|u_2| \geq 1$ und $|u_3| \geq 1$.

Beweis: Sei $w = a_1 a_2 \cdots a_{|w|}$ mit $a_i \in \Sigma$. Ich betrachte die endliche Folge von Paaren von Zuständen aus Q^2 :

$$(q_0, q'_0), (q_1, q'_1), \dots, (q_{|w|}, q'_{|w|}) \text{ mit } (q_0, q'_0) = (p, p') \text{ und}$$

$$q_{i+1} = \delta(q_i, a_{i+1}) \text{ entlang } \pi, \quad q'_{i+1} = \delta(q'_i, a_{i+1}) \text{ entlang } \pi', \quad i := 0, \dots, (|w| - 1).$$

Sei j nun die kleinste Zahl mit $(q_j, q'_j) = (q_k, q'_k)$ für ein $k < j$. Das Wort u_1 ist der Teil des Pfades π von q_0 zu q_k , das Wort u_2 der Teil von q_k zu q_j und u_3 der Teil von Zustand q_j zu $q_{|w|}$.

Dann erfüllen $z := q_j$, $z' := q'_j$ sowie u_1, u_2, u_3 die Behauptung des Lemmas, denn es gilt: $j \leq |Q|^2 < |w|$, also $|u_2| \geq 1$ und $u_3 \neq \varepsilon$. \square

Bemerkung 4.28. Solche „langen“ Pfade über dem gleichen Wort w haben also Kreise über gleiche Teile von w . Das Lemma gibt die Grenze $|Q|^2$ an, damit eine echte Faktorisierung konstruiert werden kann ($u_2 \neq w$), was ich im Beweis des nächsten Satzes verwende.

Satz 4.29. *Sei $\tau = (\Sigma, Q, I, F, E, \lambda, \rho)$ ein GA mit der Zustandsfunktion δ , der schlank und pfadneutral ist. Dann sind die beiden Aussagen äquivalent:*

- (1) τ besitzt die Zwillingsseigenschaft.
- (2) Es gilt für alle $u, v \in \Sigma^*$ mit $|uv| \leq 2|Q|^2$:
 $[\{p, q\} \in \delta(I, u), p \in \delta(p, v), q \in \delta(q, v)] \implies [\theta(p, v, p) = \theta(q, v, q)].$

Beweis: Ich verfolge den Beweis von Mohri zu seinem Lemma 2 [15, S. 291f.]. Dort wird dieser Satz für schlanke und eindeutige GAs bewiesen.

Aus (1) folgt sofort (2), denn wenn τ die Zwillingsseigenschaft hat, dann gilt (2) insbesondere für endlich viele spezielle Werte u bzw. v .

(2) \Rightarrow (1): Es ist zu zeigen:

$$u, v \in \Sigma^*, p, q \in Q \text{ mit } p, q \in \delta(I, u), p \in \delta(p, v), q \in \delta(q, v) \Rightarrow \theta(p, v, p) = \theta(q, v, q).$$

Der Beweis erfolgt per Induktion über $|uv|$. Dazu kann wegen (2) angenommen werden, dass $|uv| > 2|Q|^2$ ist. Dann gilt aber $|u| > |Q|^2$ oder $|v| > |Q|^2$.

1. Fall: $|u| > |Q|^2$.

Es gibt Pfade π, π' , sowie Zustände $i, i' \in I$, so dass $\pi \in i \xrightarrow{u} p$ und $\pi' \in i' \xrightarrow{u} q$. Der zugrunde liegende Automat von τ ist ein NNEA. Wir wenden Lemma 4.27 an und erhalten:

$$\exists u_1, u_2, u_3 \in \Sigma^*, |u_2| > 0, z, z' \in Q, u_1 u_2 u_3 = u,$$

so dass die Pfade π, π' über dem gleichen Wort u_2 Kreise haben:

$$\pi \in i \xrightarrow{u_1} z \xrightarrow{u_2} z \xrightarrow{u_3} p, \pi' \in i' \xrightarrow{u_1} z' \xrightarrow{u_2} z' \xrightarrow{u_3} q.$$

Da $|u_1 u_3 v| < |uv|$ folgt nach der Induktionsvoraussetzung: $\theta(p, v, p) = \theta(q, v, q)$.

2. Fall: $|v| > |Q|^2$.

Seien $\pi \in p \xrightarrow{v} p$ und $\pi' \in q \xrightarrow{v} q$ zwei Pfade mit minimaler Ausgabe, also entsprechend $\theta(p, v, p)$ und $\theta(q, v, q)$. Ich wende erneut das Lemma 4.27 an.

$$\Rightarrow \exists v_1, v_2, v_3 \in \Sigma^*, |v_2| > 0, v_3 \neq \varepsilon, z, z' \in Q, v_1 v_2 v_3 = v \text{ mit}$$

$$\pi \in p \xrightarrow{v_1} z \xrightarrow{v_2} z \xrightarrow{v_3} p, \pi' \in q \xrightarrow{v_1} z' \xrightarrow{v_2} z' \xrightarrow{v_3} q$$

Nun gilt aber $|uv_1 v_3| < |uv|$ und nach Induktionsvoraussetzung somit:

$\theta(p, v_1 v_3, p) = \theta(q, v_1 v_3, q)$. Wegen $v_3 \neq \varepsilon$ können wir mit $|uv_1 v_2| < |uv|$ auf $\theta(z, v_2, z) = \theta(z', v_2, z')$ schließen.

Da τ pfadneutral und schlank ist, ist die minimale Ausgabe über einem Wort zwischen zwei Zuständen gleich der Ausgabe eines speziellen Pfades zwischen diesen Zuständen zu dem Wort. Also ist

$$\theta(p, v, p) = \theta(p, v_1 v_3, p) + \theta(z, v_2, z)$$

$$\theta(q, v, q) = \theta(q, v_1 v_3, q) + \theta(z', v_2, z')$$

und deshalb $\theta(p, v, p) = \theta(q, v, q)$. \square

Die Bedingung (2) des obigen Satzes entspricht endlich vielen Aussagen, die jeweils entscheidbar sind. Deshalb ist es für schlanke und pfadneutrale gewichtete Automaten entscheidbar, ob sie die Zwillingsseigenschaft besitzen. Da nach Satz 4.26 für einen schlanken und pfadneutralen GA τ , die Bedingung, dass $|\tau|$ subsequentiell ist, mit der Eigenschaft, dass τ die Zwillingsseigenschaft hat, zusammen fällt, kann ich auch entscheiden, ob τ subsequentiell ist:

Satz 4.30. *Gegeben sei ein pfadneutraler gewichteter Automat τ . Es ist entscheidbar, ob τ eine subsequentielle Funktion berechnet und ob $|\tau|$ beschränkte Variation hat. Falls τ zusätzlich schlank ist, so ist auch entscheidbar, ob $\text{PSDET}(\tau)$ terminiert und ob τ die Zwillingsseigenschaft besitzt.*

Beweis: Aus dem gewichteten Automaten τ können wir effektiv einen äquivalenten schlanken GA τ' konstruieren, der ebenfalls pfadneutral ist (Bemerkung 4.18). Nach dem Lemma 4.27 ist es entscheidbar, ob τ' die Zwillingsseigenschaft besitzt. Die Zwillingsseigenschaft von $|\tau|$ ist für die betrachtete Potenzreihe aber durch Satz 4.26 äquivalent dazu, dass $|\tau|$ subsequentiell ist, dass $\text{PSDET}(\tau')$ terminiert oder, dass $|\tau|$ beschränkte Variation besitzt. \square

5 Ausblick

Die Gleichungssysteme von Mohri in Abschnitt 3.3 bzw. von Béal/Caron in 3.4 zur Berechnung größter gemeinsamer Präfixe (innerhalb einer Komponente des Komponentengraphen) von Zuständen eines Automaten stellen *Fixpunkt-Gleichungen* dar. Es wurde diskutiert, dass es, wenn es in dem betrachteten Automaten ε -Kreise gibt, mehrere Lösungen geben kann, wobei die maximale Lösung (also der längste Präfix) gesucht ist. Demnach wird der maximale Fixpunkt der Gleichungen benötigt. Dieser Ansatz ist sehr einsichtig und simpel. Für die Berechnung des maximalen Fixpunktes einer Gleichung gibt es bereits viele Verfahren, die dann Anlass zu neuen Konstruktionen des Präfixes eines Automaten geben.

Ferner ist ein allgemeines Verfahren für die Konstruktion des minimalen sequentiellen Transducers zu einer sequentiellen Funktion gesucht. Dabei sind die Vermutungen 3.23 am Ende des Abschnitts 3.2 und 3.33 in 3.5.1 zu beachten.

In Kapitel 4 habe ich gezeigt, dass es für pfadneutrale und schlanke gewichtete Automaten entscheidbar ist, ob sie eine subsequentielle Funktion berechnen. Eine pfadneutral darstellbare Funktionen ist genau dann subsequentiell, wenn sie beschränkten Variation besitzt. Es wäre interessant die definierte Klasse von Funktionen so zu vergrößern (oder sogar zu maximieren), dass einige der aufgeführten Eigenschaften noch immer gelten. Außerdem sind Resultate von Kapitel 4 auf andere Semiringe verallgemeinerbar.

6 Abkürzungsverzeichnis

DEA deterministischer endlicher Automat, S. 21

GA gewichteter Automat, S. 49

GS Gleichungssystem, S. 38

λp **SST** λ - p -subsequentieller Transducer, S. 43

λ **SST** λ -subsequentieller Transducer, S. 43

λ **ST** λ -sequentieller Transducer, S. 23

LST links-sequentieller Transducer, S. 11

LSST links-subsequentieller Transducer, S. 13

NEA nichtdeterministischer endlicher Automat, S. 21

NNEA nichtdeterministischer endlicher Automat mit nichtdeterministischer Initialzustandsmenge, S. 21

PR Potenzreihe, S. 50

p **SST** p -subsequentieller Transducer, S. 14

SCC strongly connected component (starke Zusammenhangskomponente), S. 37

ST sequentieller Transducer, S. 12

SST subsequentieller Transducer, S. 13

Danksagung

Zuallererst danke ich meinem Betreuer Prof. Dr. Manfred Droste für viele sehr hilfreiche Gespräche und Hinweise. Ohne ihn hätte diese Arbeit nie stattgefunden. Weiterhin gilt mein Dank Dr. Daniel Kirsten, Andreas Zollmann, Thomas Schneider und meinem Vater für ihre Diskussionen zum Thema und Kommentare über frühere Versionen dieser Arbeit sowie Heiko Reppe, der in dieser Zeit seine Diplomarbeit geschrieben hat und mir über die Monate Mut gab.

Literatur

- [1] A. V. Aho; J. E. Hopcroft; J. D. Ullman. The Design and Analysis of Computer Algorithms. *Addison Wesley, Reading, MA*, 1974.
- [2] M. P. Béal; O. Carton. Computing the Prefix of an Automaton. *RAIRO Inform. Théor. Applic., to appear*, 2001.
- [3] J. Berstel. Transductions and Context-free Languages. *B. G. Teubner*, 1979.
- [4] J. Berstel. Rational Series and their Languages. *Springer*, 1988.
- [5] D. Breslauer. The Suffix Tree of a Tree and Minimizing Sequential Transducers. *Theoret. Comput. Sci.*, Bd. 191, S. 131–144, 1998.
- [6] J. A. Brzozowski. Conical Regular Expressions and Minimal State Graphs for Definite Events. *Mathematical Theory of Automata*, Bd. 12, S. 529–561, 1962.
- [7] C. Choffrut. A Generalisation of Ginsburg and Rose’s Characterisation of GSM Mappings. *ICALP’79, Lect. Notes Comp. Sci.*, Bd. 71, S. 88–103, 1979.
- [8] C. Choffrut. Minimizing Subsequential Transducers: A Survey. *preprint, LIAFA, Université Paris 7, 2*, 2001.
- [9] M. Droste; P. Gastin. The Kleene-Schützenberger Theorem for Formal Power Series in Partially Commuting Variables. *Information and Computation*, Bd. 153, S. 47–80, 1999.
- [10] S. Ginsburg; G.F. Rose. A Characterization of Machine Mappings. *Can. J. of Math.*, Bd. 18, S. 381–388, 1966.
- [11] J. E. Hopcroft; J. D. Ullman. Introduction to Automata Theory, Languages and Computation. *Addison Wesley, Reading, MA*, 1979.
- [12] M. Mohri. Compact Representations by Finite-state Transducers. *Proceedings of the 32th Annual Meeting, Association for Computational Linguistics*, 1994.
- [13] M. Mohri. Minimization of Sequential Transducers. *Lect. Notes Comp. Sci.*, Bd. 807, S. 151–163, 1994.
- [14] M. Mohri. On some Applications of Finite-state Automata Theory to Natural Language Processing. *Journal of Natural Language Engineering*, Bd. 2, S. 1–20, 1996.
- [15] M. Mohri. Finite-state Transducers in Language and Speech Processing. *Comp. Linguistics*, Bd. 23, S. 269–311, 1997.

- [16] M. Mohri. Minimization Algorithms for Sequential Transducers. *Theoret. Comput. Sci.*, Bd. 234, S. 177–201, 2000.
- [17] S. Yu. Regular Languages, in: G. Rozenberg, A. Salomaa (Eds.), *Handbook of Formal Languages*, Bd. 1, S. 90–99, *Springer*, 1997.

Erklärung

Hiermit erkläre ich, dass ich die am heutigen Tage eingereichte Diplomarbeit zum Thema „Zur Minimalisierung und Determinisierung von sequentiellen Transducern“ unter der Betreuung von Prof. Dr. Droste selbständig erarbeitet, verfasst und Zitate kenntlich gemacht habe. Andere als die angegebenen Hilfsmittel wurden nicht verwendet.

Datum

Unterschrift