NICHTSEQUENTIALITÄT IN DER SPRACHVERARBEITUNG MIT FST

Günther Wirsching

Katholische Universität Eichstätt-Ingolstadt
guenther.wirsching@ku.de

Kurzfassung: Ausgehend von der Beobachtung, dass ein Worthypothesengraph die mathematische Struktur eines allgemeinen nichtsequentiellen Wortes hat, wird untersucht, welche Bedeutung die Nichtsequentialität in der Sprachverarbeitung hat. Dabei stellt sich heraus, dass die der Spracherkennung angemessene Struktur die *Multisequentialität* ist, die zwischen der Sequentialität (z.B. einer Signalfolge) und allgemeiner Nichtsequentialität (z.B. einer beschrifteten partiell geordneten Menge) liegt. Auch auf semantischer Ebene sind die Daten noch multisequentiell und nicht beliebig nichtsequentiell. Allerdings treten auf der semantischen Ebene logische Strukturen auf, die die benötigten FST kompliziert machen und deshalb die Untersuchung einer anderen Modellierungssprache wie z.B. Petrinetze motivieren.

1 Das Vorgehen in der Sprachverarbeitung

In der automatischen Sprachverarbeitung geht es unter anderem darum, gesprochene Sprache dem Inhalt nach zu verstehen. Eine wichtige Technologie, die hier zum Einsatz kommt, sind die gewichteten *finite state transducer* (abgekürzt FST), siehe z. B. Mohri [4] für eine ausführliche Darstellung. Das Verstehen gesprochener Sprache geschieht in hierarchischen Systemen wie z. B. dem an der TU Dresden entwickeltem UASR [1] in mehreren aufeinanderfolgenden Schritten: zunächst Signal- und Merkmalanalyse, anschließend eine phonetische, lexikalische und syntaktische Analyse. Die verarbeiteten Daten sind bei der Signal- und Merkmalanalyse noch einfach sequentiell, aber schon die phonetische Analyse liefert häufig mehrere mögliche Phonemsequenzen, und das Ergebnis der syntaktischen Analyse ist im allgemeinen eine Bestenliste, die aus mehreren Erkennalternativen besteht. Die Daten sind also ab der dritten Ebene *multisequentiell*, und werden der Einfachheit halber in einem *Phonem-* oder *Worthypothesengraph* zusammenfassend dargestellt. Ein Worthypothesengraph ist aber bereits ein „nichtsequentielles Wort“, wie im folgenden gezeigt und analysiert wird.

In diesem Beitrag betrachte ich Fragen der Sequentialität und Nichtsequentialität; daher werden bei der Betrachteung von FSTs mögliche Gewichte im Folgenden außer Acht gelassen – obwohl sie in Anwendungen eine entscheidende Rolle spielen.

2 Mathematische Analyse

2.1 Lposets und azyklische Hypothesengraphen

Eine *partielle Ordnung* oder *Partialordnung* auf einer Menge M ist eine Relation $R \subseteq M \times M$ mit den folgenden Eigenschaften:

- *Reflexivität:* $\forall x \in M : xRx$,

- *Transitivität:* $\forall x,y,z \in M : (xRy \wedge yRz \Rightarrow xRz)$.
- *Antisymmetrie:* $\forall x,y \in M : (xRy \wedge xRy \Rightarrow x = y)$.

Um eine endliche partiell geordnete Menge, einen endlichen *Poset* (abkürzend für *partially ordered set*), bildlich darzustellen, benutzt man in der Regel ein *Hasse-Diagramm*, das ist eine Relation $H \subseteq M \times M$ mit den Eigenschaften:

- *Irreflexivität:* $\forall x \in M : (x,x) \notin H$,
- *Intransitivität:* $\forall x,y,z \in M : (xHy \wedge yHz \Rightarrow (x,z) \notin H)$,
- *Azyklizität:* $\forall x_0,\ldots,x_n \in M : \big((\forall i \in \{1,\ldots,n\} : x_{i-1}Hx_i) \Rightarrow x_0 \neq x_n\big)$.

Ein Hasse-Diagramm H stellt eine partielle Ordnung R dar, wenn $R = H^*$ die reflexiv-transitive Hülle von H ist. Man kann beweisen, dass es zu jeder endlichen Partialordnung R ein durch R eindeutig bestimmtes Hasse-Diagramm H_R gibt, welches diese darstellt.

Ein *Digraph* (oder *gerichteter Graph*) $\mathscr{G} = (E,K)$ besteht aus einer *Eckenmenge E* und einer *Kantenmenge* $K \subseteq E \times E$. Ein *Zyklus* in einem Digraphen (E,K) ist eine endliche Folge von Ecken $(e_1,\ldots,e_n)$ mit

$$n \geq 2 \text{ und } \forall i \in \{1,\ldots,n-1\} : \quad (e_i,e_{i+1}) \in K.$$

Ein Digraph heißt *azyklisch*, wenn er keinen Zyklus besitzt.

Satz 1. *Jedes Hasse-Diagramm kann als azyklischer, gerichteter Graph aufgefasst werden, aber nicht umgekehrt.*

Eine beschriftete partiell geordnete Menge, ein *Lposet* (abkürzend für *labeled partially ordered set*), über einem Alphabet A ist ein Quadrupel $\mathscr{L} = (M,R,A,\ell)$, bestehend aus einer (Ecken-)Menge M, einer partiellen Ordnung $R \subseteq M \times M$, und einer Abbildung $\ell : M \to A$.

Ein *(Kanten-)beschrifteter Digraph* über einem Alphabet A ist Quadrupel (E,K,A,ℓ), wobei (E,K) ein Digraph und $\ell : K \to A$ eine beliebige Beschriftungsabbildung ist. Beispiel: Jeder Worthypothesengraph ist ein beschrifteter Digraph.

Stellt man sich den Lposet als Hasse-Diagramm mit Eckenbeschriftungen vor, dann erscheinen die beiden Konzepte dual zueinander: beim Lposet sind die Ecken beschriftet, beim beschrifteten Digraphen die Kanten. Tatsächlich sind die beiden Konzepte beinahe dual zueinander, aber man muss beim Vertauschen von „Ecken" und „Kanten" etwas aufpassen.

Ist $\mathscr{B} = (E,K,A,\ell)$ ein beschriftete Digraph, dann ist durch

$$M := K \quad \text{und} \quad (x,w)R(y,z) :\Longleftrightarrow \big((x,w) = (y,z) \vee (w,y) \in K^*\big),$$

wobei K^* die *reflexiv-transitive Hülle* der Relation $K \subseteq E \times E$, also die Menge der Wege im Digraphen (E,K), bezeichnet, ein Lposet $L(\mathscr{B}) := (M,R,A,\ell)$ gegeben.

Ist umgekehrt $\mathscr{L} = (M,R,A,\ell)$ ein Lposet, so gibt es mehrere gleichermaßen „natürliche" Möglichkeiten, dazu einen beschrifteten Digraphen zu assoziieren. Eine Möglichkeit ist, zunächst zur Eckenmenge M des Lposet eine *initiale Ecke* $i \notin M$ hinzuzufügen und die Partialordnung um alle Paare (i,x) zu erweitern,

$$M' := M \cup \{i\}, \qquad R' := R \cup \{(i,x) : x \in M'\},$$

dann das zu R' assoziierte Hasse-Diagramm $H_{R'}$ als Digraphen $D(\mathscr{L}) = (M', H_{R'})$ interpretieren, und schließlich die Beschriftungen von jeder Ecke $y \in M$ auf alle in y endenden Pfeile $(x,y) \in H'$ zu verschieben:

$$\forall (x,y) \in H_{R'}: \quad \eta(x,y) := \ell(y).$$

Das ergibt einen zum Lposet $\mathscr{L}$ assoziierten beschrifteten Digraphen $B(\mathscr{L}) := (M', H_{R'}, A, \eta)$. Die Abbildungen $\mathscr{L} \mapsto B(\mathscr{L})$ und $\mathscr{B} \mapsto L(\mathscr{B})$ ergeben keine strenge Dualität, denn es gilt im Allgemeinen weder $\mathscr{L} = L(B(\mathscr{L}))$ noch $\mathscr{B} = B(L(\mathscr{B}))$.

2.2 Maximale Ketten und Sequentialisierungen

Zu einem beliebigen Lposet $\mathscr{L} = (M, \leq, A, \ell)$ (die partielle Ordnung wird ab jetzt mit $\leq$ bezeichnet) kann man zwei verschiedene Mengen von A-Folgen assoziieren:

1. Die Menge $K(\mathscr{L})$ der *maximalen Ketten* $k_1 \ldots k_n \in A^+$ mit der Bedingung:

   $$\exists x_1, \ldots, x_n \in M : \big(x_1 < \ldots < x_n \text{ nicht verlängerbar} \wedge \forall i \in \{1, \ldots, n\} : k_i = \ell(x_i)\big).$$

   Einige Eigenschaften der maximalen Ketten:

   (a) Aus der Menge $K(\mathscr{L})$ der maximalen Ketten kann der Lposet nicht rekonstruiert werden; aus $K(\mathscr{L}_1) = K(\mathscr{L}_2)$ folgt nicht $\mathscr{L}_1 = \mathscr{L}_2$, wie das folgende Beispiel zeigt:

   $$\mathscr{L}_1: \begin{tikzcd} \boxed{a} \arrow[r] \arrow[dr] & \boxed{b} \\ & \boxed{c} \end{tikzcd} \qquad \mathscr{L}_2: \begin{tikzcd} \boxed{a} \arrow[r] & \boxed{b} \\ \boxed{a} \arrow[r] & \boxed{c} \end{tikzcd}$$

   Der Lposet $\mathscr{L}_1$ mit der Grundmenge $M_1 = \{1,2,3\}$, der Partialordnung, die von $1 \leq 2$ und $1 \leq 3$ erzeugt wird, und den Beschriftungen $\ell(1) = a$, $\ell(2) = b$, $\ell(3) = c$, hat die maximalen Ketten $K(\mathscr{L}_1) = \{ab, ac\}$.

   Der Lposet $\mathscr{L}_2$ mit der Grundmenge $M_2 = \{1,2,3,4\}$, der Partialordnung, die von $1 \leq 2$ und $3 \leq 4$ erzeugt wird, und den Beschriftungen $\ell(1) = \ell(3) = a$, $\ell(2) = b$, $\ell(4) = c$, hat die gleiche Menge maximaler Ketten $K(\mathscr{L}_2) = \{ab, ac\}$.

   (b) Die Menge der maximalen Ketten kann ziemlich groß werden. Z.B. besitzt das „Kreuzband"

   $$\begin{tikzcd} \boxed{a} \arrow[dr] & & \boxed{a} \arrow[dr] & & \boxed{a} \arrow[dr] & & \boxed{a} \\ & \boxed{b} \arrow[ur] \arrow[dr] & & \cdots \arrow[ur] \arrow[dr] & & \boxed{b} \arrow[ur] \arrow[dr] & \\ \boxed{c} \arrow[ur] & & \boxed{c} \arrow[ur] & & \boxed{c} \arrow[ur] & & \boxed{c} \end{tikzcd}$$

   mit $m = |M|$ Ecken und den skizzierten Beschriftungen insgesamt $2^{(m+1)/3}$ verschiedene maximale Ketten.

   (c) In der Sprachverarbeitung werden die maximalen Ketten eines Worthypothesengraphen häufig als Erkennalternativen betrachtet, die sich gegenseitig ausschließen.

2. Die Menge $S(\mathscr{L})$ der *Sequentialisierungen* $k_1 \ldots k_m \in A^+$ mit der Bedingung:

   $$M = \{x_1, \ldots, x_m\} : \big(\forall j \in \{1, \ldots, m-1\} : x_j \not\geq x_{j+1} \wedge \forall j \in \{1, \ldots, m\} : k_j = \ell(x_j)\big).$$

(a) Es ist möglich, aus der Menge $S(\mathscr{L})$ den Lposet $\mathscr{L}$ exakt zu rekonstruieren.

(b) Die beste allgemeine Größenabschätzung ist $|S(\mathscr{L})| = O(m!)$.
Die Gleichheit $|S(\mathscr{L})| = m!$ wird genau dann erreicht, wenn die Partialordnung nur aus der Diagonalen $\Delta = \{(x,x) \mid x \in M\}$ besteht.

(c) In der Sprachverarbeitung haben die Sequentialisierungen eines Worthypothesengraphen meistens keine Bedeutung.

Man bezeichnet eine endliche oder unendliche Menge von Lposet über einem gemeinsamen Alphabet A als *partielle Sprache über* A. Sind alle zugrundeliegenden Posets totale Ordnungen, dann heißt die Sprache *sequentiell* und kann als Teilmenge von A^* betrachtet werden; z. B. sind Sprachen, deren Worte Signalfolgen sind, sequentiell. Zur Unterscheidung von sequentiellen Sprachen $\mathscr{S} \subset A^*$ nennt man eine Sprache, bei der mindestens einem Wort eine nicht-totale partielle Ordnung zugrunde liegt, eine *nichtsequentielle* Sprache. Begrifflich zwischen den sequentiellen Sprachen und den allgemeinen nichtsequentiellen Sprachen liegen die *multisequentiellen* Sprachen, deren Worte *endliche Teilmengen* von A^* sind (die sich natürlich auch als Lposet darstellen lassen, siehe etwa das obige Beispiel $\mathscr{L}_2$).

3 FST-Technologie

In der automatischen Spracherkennung hat es sich gezeigt, dass die Erkennraten besser werden, wenn man in den mittleren Hierarchieebenen mehrere Erkennalternativen verarbeitet, denn die dadurch erreichte Flexibilität kann auf den höheren Hierarchieebenen genutzt werden. Zur effizienten Darstellung einer größeren Menge von Erkennalternativen werden dabei Phonem-, Wort- und Satzhypothesengraphen genutzt. Die Verwendung von derartigen *Hypothesengraphen*, also nichtsequentiellen Worten, war eine wichtige Motivation zur Entwicklung der FST-Technologie.

Die innere Struktur eines FST zur Übersetzung einer Sprache über A in eine Sprache über B ist die eines Kanten-beschrifteten Digraphen, dessen Ecken *Zustände* genannt werden, und bei dem gewisse *Eingangszustände* und gewisse *Ausgangszustände* markiert sind. Die Beschriftungen der Kanten enthalten jeweils einen Eingangsbuchstaben $a \in A \cup \{\varepsilon\}$, wobei $\varepsilon \in A^*$ das leere Wort bezeichnet, und einen Ausgangsbuchstaben $b \in B \cup \{\varepsilon\}$. Bei den in der automatischen Spracherkennung verwendeten *gewichteten* FST enthält jede Kantenbeschriftung zusätzlich ein Gewicht aus einem geeigneten Gewichtshalbring. Gemäß der im vorigen Abschnitt vorgestellten Begriffsbildung ergibt sich daraus der folgende Sachverhalt.

Satz 2. Ein azyklischer FST mit Eingangsalphabet A und Ausgangsalphabet B ist ein nichtsequentielles Wort über dem Alphabet $(A \cup \{\varepsilon\}) \times (B \cup \{\varepsilon\})$.

Aus funktionaler Sicht ist ein FST eine Blackbox T, die eine effiziente Übersetzung eines Wortes $w \in A^*$ aus einer Eingangssprache $\mathscr{S}_A \subseteq A^*$ über einem gegebenen Alphabet A in eine (leere oder nicht-leere) Menge $X \subseteq B^*$ von Worten aus einer Ausgangssprache $\mathscr{S}_B \subseteq B^*$ über einem Alphabet B übersetzt (siehe Mohri [4]):

$$w \quad \overset{T}{\longmapsto} \quad X\,. \tag{1}$$

Zunächst ist es nicht ausgeschlossen, dass die von T zu einem Eingabewort $w \in A^*$ produzierte Wortmenge $X \subseteq B^*$ endlich ist. Da das Eingabewort w endlich ist, kann eine unendliche Ausgabemenge höchstens dann auftreten, wenn T nichtdeterministisch ist und eine ε-*Schleife* enthält

– hierbei ist eine ε-Schleife ein Zykel, bei dem zu jeder Kante das leere Eingabewort ε gehört. In der Sprachverarbeitung ist es häufig sinnvoll, unendliche Ausgabemengen von vorne herein auszuschließen.

Auf der anderen Seite wird es sich insbesondere im Hinblick auf die Semantik als sinnvoll erweisen, Transduktoren einzubeziehen, die zu einem Eingabewort mehrere Ausgabewörter assoziieren können, die also im automatentheoretischen Sinn *nichtdeterministisch* sind. Die nun folgenden Betrachtungen beziehen sich auf diejenigen FST, die keine ε-Schleifen enthalten; nichtdeterministische Transduktoren sind explizit enthalten.

Verarbeitet man eine endliche Menge $W \subseteq \mathscr{S}_A$ wortweise, so ergibt sich aus (1) eine Übersetzung von Teilsprachen in Teilsprachen:

$$\mathscr{S}_A \supseteq W \quad \overset{T}{\longmapsto} \quad X \subseteq \mathscr{S}_B\,; \tag{2}$$

dabei werden endliche Wortmengen auf endliche Wortmengen abgebildet. Ein FST ist also, funktional gesehen, die Implementierung einer Mengenabbildung

$$T : \mathscr{P}_0(\mathscr{S}_A) \longrightarrow \mathscr{P}_0(\mathscr{S}_B), \qquad X \longmapsto W = T(X)\,, \tag{3}$$

wobei für eine Menge M durch $\mathscr{P}_0(M)$ die Menge ihrer endlichen Teilmengen bezeichnet wird. Damit implementiert ein FST – funktional gesehen – die Übersetzung einer multisequentiellen Sprache in eine andere multisequentielle Sprache.

Hat man zwei Transduktoren T_{AB} und T_{BC}, bei denen die Ausgangssprache des ersten in der Eingangssprache des zweiten enthalten ist, dann kann man diese als Mengenabbildungen komponieren:

$$\mathscr{P}(\mathscr{S}_A) \quad \overset{T_{AB}}{\longrightarrow} \quad \mathscr{P}(\mathscr{S}_B) \quad \overset{T_{BC}}{\longrightarrow} \quad \mathscr{P}(\mathscr{S}_C)\,. \tag{4}$$

Mohri [4] beschreibt auch eine *Komposition* von FST, das ist ein Algorithmus, der aus zwei von den Sprachen her passende FST T_{AB} und T_{BC} einen FST $T_{AC} = T_{BC} \circ T_{AB}$ macht, der funktional das Gleiche leistet, wie die Hintereinanderausführung (4). Die Komposition von FST hat die folgende Eigenschaft: Ist T_{AB} azyklisch, und enthält T_{BC} keine „ε-Schleifen“, dann ist T_{AC} ebenfalls azyklisch.

Prinzipiell können zwei FST mit unterschiedlicher innerer Struktur die gleiche Mengenabbildung (3) realisieren. Mohris Kompositionsalgorithmus konstruiert einen konkreten FST, aber aus funktionaler Sicht ist die Komposition zweier FST nicht eindeutig festgelegt. Zusammenfassend erhalten wir das folgende Resultat.

Satz 3. Sei T ein FST mit Eingangsalphabet A und Ausgangsalphabet B (ohne ε-Schleifen).

1. In funktionaler Hinsicht übersetzt T eine multisequentielle Sprache über A in eine multisequentielle Sprache über B.

2. Jeder Kompositionsalgorithmus ergibt durch Komposition mit T einen Übersetzer von einer nichtsequentiellen Sprache über A in eine nichtsequentielle Sprache über B.

4 Die semantische Ebene

Es gibt mehrere Versuche, FST als semantische Träger zu verwenden [5, 3] – wie sich dabei gezeigt hat, ist das nicht ganz unproblematisch. Einige der Schwierigkeiten scheinen mit der „natürlichen“ Nichtsequentialität semantischer Daten zusammenzuhängen.

4.1 Darstellung semantischer Daten

Eine ziemlich allgemeine Form, semantische Daten darzustellen, ist die Verwendung von *Merkmal-Werte-Relationen* (MWR) [2]. Mathematisch besteht eine MWR aus einer beschrifteten Knotenmenge M und einer azyklischen Relation $R \subseteq M \times M$; in den in [2] betrachteten Fällen ist diese Relation zusätzlich irreflexiv und intransitiv, also ein Hasse-Diagramm. Aus der Azyklizität von R folgt, dass die reflexiv-transitive Hülle R^* auch antisymmetrisch ist, also kann eine MWR als Lposet aufgefasst werden.

Ein Beispiel: Gegeben sei die Äußerung:

$$\text{„\textit{Treffpunkt um zwölf Uhr am Domplatz}“,} \tag{5}$$

eine sinnvolle Semantik wäre z. B.:

$$\begin{tikzcd}
& \boxed{\text{Ort}} \arrow[r] & \boxed{\text{Domplatz}} \\
\boxed{\text{Treffpunkt}} \arrow[ur] \arrow[dr] & & \\
& \boxed{\text{Zeit}} \arrow[r] & \boxed{\text{12 Uhr}}
\end{tikzcd} \tag{6}$$

Das ist ein nichtsequentielles Wort über dem Alphabet

$$\big\{\text{Treffpunkt, Ort, Zeit}\big\} \cup \big(\text{Ortsbezeichnungen}\big) \cup \big(\text{mögliche Uhrzeiten}\big)\,.$$

Dieses nichtsequentielle Wort soll in einem Sprachdialogsystem durch eine Folge von FST erzeugt werden. Da ein FST in funktionaler Hinsicht nur multisequentiell arbeitet, sollte auch die Ausgabe nur multisequentiell interpretiert werden:

$$\left\{\begin{array}{l}
\boxed{\text{Treffpunkt}} \longrightarrow \boxed{\text{Ort}} \longrightarrow \boxed{\text{Domplatz}} \\
\boxed{\text{Treffpunkt}} \longrightarrow \boxed{\text{Zeit}} \longrightarrow \boxed{\text{12 Uhr}}
\end{array}\right. \tag{7}$$

4.2 Mehrere Erkennalternativen

Wie das obige Beispiel zeigt, besteht *eine* Semantik häufig aus *mehreren* maximalen Ketten. Damit entsteht ein logisches Problem, wenn mehrere Erkennalternativen berücksichtigt werden sollen: die zu *einer* Erkennalternativen gehörigen Ketten müssen mit „und“ verbunden werden, während zu *unterschiedlichen* Erkennalternativen gehörige Ketten mit dem exklusiven „oder“ zu verbinden sind.

Betrachten als Beispiel eine Ergänzung von (5) um eine zweite Erkennalternative:

$$\text{„\textit{Treffpunkt um elf Uhr am Nordplatz}“,} \tag{8}$$

dann erhält man als Gesamtsemantik:

$$\text{1. Erkennalternative:} \quad \left\{\begin{array}{l}
\boxed{\text{Treffpunkt}} \longrightarrow \boxed{\text{Ort}} \longrightarrow \boxed{\text{Domplatz}} \\
\boxed{\text{Treffpunkt}} \longrightarrow \boxed{\text{Zeit}} \longrightarrow \boxed{\text{12 Uhr}}
\end{array}\right.$$

$$\text{2. Erkennalternative:} \quad \left\{\begin{array}{l}
\boxed{\text{Treffpunkt}} \longrightarrow \boxed{\text{Ort}} \longrightarrow \boxed{\text{Nordplatz}} \\
\boxed{\text{Treffpunkt}} \longrightarrow \boxed{\text{Zeit}} \longrightarrow \boxed{\text{11 Uhr}}
\end{array}\right.$$

Diese Gesamtsemantik besteht, multisequentiell gedacht, aus vier Sequenzen, von denen je zwei als zu einer Erkennalternative gehörig zusammengefasst werden müssen. Eine solche Konstruktion nenne ich *geschachtelt multisequentiell*, um die beschriebene Interpretation anzudeuten.

4.3 Konstruktion der Semantik im hierarchischen System

In der automatischen Spracherkennung muss die Semantik aus dem akustischen Signal mit Hilfe des Kontextes gewonnen werden. Die relevante Kontextinformation besteht dabei aus

1. Sprachdaten zur Modellierung der Aussprache,
2. linguistischen Daten zur Modellierung der Formulierungen,
3. einer an die jeweilige Situation angepasste MWR zur Modellierung derjenigen Semantiken, die das System verstehen können soll [6].

Bei einem hierarchischen System geschieht die Analyse einer Äußerung in mehreren aufeinander aufbauenden Schritten:

$$\begin{array}{cl}
\boxed{\text{akustisches Signal}} & \text{(sequentiell)} \\
\downarrow & \\
\boxed{\text{Folge von Merkmalvektoren}} & \text{(sequentiell)} \\
\downarrow & \\
\boxed{\text{Phonemhypothesengraph}} & \text{(multisequentiell)} \\
\downarrow & \\
\boxed{\text{Worthypothesengraph}} & \text{(multisequentiell)} \\
\downarrow & \\
\boxed{\text{Semantikhypothesengraph}} & \text{(geschachtelt multisequentiell)}
\end{array}$$

Dabei wird jeder Analyseschritt von einem geeigneten FST durchgeführt; durch FST-Komposition können mehrere Schritte zusammengefasst werden. Die Semantik wird erst im letzten Schritt aufgebaut, wo die in [6] formulierten Überlegungen zum Tragen kommen.

Der funktionale Standpunkt impliziert, dass von der Ausgabe eines FST nur die multisequentielle Information weiterbenutzt werden darf. Man kann die Sequenzen prinzipiell so gestalten, dass bei jeder einzelnen Sequenz die Information über die Zugehörigkeit zu den Erkennalternativen enthalten ist. Damit lässt sich die Ausgabe dann logisch analysieren, wie es für eine Weiterverarbeitung erforderlich ist.

5 Fazit

Zusammenfassung der erzielten Ergebnisse zur Nichtsequentialität der Daten auf der Semantik-Ebene und deren Behandlung in automatischen Sprachdialogsystemen:

1. Bei korrekter Verwendung von FSTs ist nur die funktionale Betrachtung zulässig. Der Einsatz von FSTs als nichtsequentielle Transduktoren ist nur mit einem festgelegten Kompositionsalgorithmus wohldefiniert.
2. Da man die Semantik so gestalten kann, dass auf Semantik-Ebene nur die maximalen Ketten der auftretenden Lposet benötigt werden, ist ein funktionaler FST-Einsatz prinzipiell möglich.
3. Bei komplexeren Systemen treten auf der Semantik-Ebene Probleme mit der Logik auf, die eine komplizierte FST-Architektur erforderlich machen.
4. Eine vielversprechende Möglichkeit, der strukturellen Nichtsequentialität der Semantik-Daten Rechnung zu tragen, ist der Einsatz von Petrinetzen, der in weiteren Forschungsarbeiten untersucht werden soll.

Literatur

[1] HOFFMANN, R., M. EICHNER und M. WOLFF: *Analysis of verbal and nonverbal acoustic signals with the Dresden UASR system.* In: AL., A. E. ET (Hrsg.): *Verbal and Nonverbal Communication Behaviours*, S. 200–218, Berlin etc., 2007. Springer.

[2] HUBER, M., C. KÖLBL, R. LORENZ, R. RÖMER und G. WIRSCHING: *Semantische Dialogmodellierung mit gewichteten Merkmal-Werte-Relationen.* In: HOFFMANN, R. (Hrsg.): *Elektronische Sprachsignalverarbeitung 2009. Tagungsband der 20. Konferenz. Dresden, 21. bis 23. September 2009*, Bd. 53 d. Reihe *Studientexte zur Sprachkommunikation*, S. 25–32. TUDpress, Sep. 2009.

[3] HUBER, M., C. KÖLBL und G. WIRSCHING: *Endliche gewichtete Transduktoren als semantischer Träger.* In: KRÖGER, B. J. und P. BIRKHOLZ (Hrsg.): *Elektronische Sprachsignalverarbeitung 2011. Tagungsband der 22. Konferenz. Aachen, 28. bis 30. September 2011*, Bd. 61 d. Reihe *Studientexte zur Sprachkommunikation*, S. 176–183, Sep. 2011.

[4] MOHRI, M.: *Weighted Automata Algorithms.* In: DROSTE, M., W. KUICH und H. VOGLER (Hrsg.): *Handbook of Weighted Automata*, S. 213–256. Springer, 2009.

[5] RAYMOND, C., F. BÉCHET, R. D. MORI und G. DAMNATI: *On the use of finite state transducers for semantic interpretation.* Speech communication, 48(3-4):288–304, March-April 2006.

[6] WIRSCHING, G. und C. KÖLBL: *Language Modeling with Utterance-Meaning Pairs.* Techn. Ber. 2011–12, Angewandte Informatik, Universität Augsburg, 2011.