

Übertragung automatentheoretischer Sätze auf Chomsky-Sprachen¹

Von

G. Hotz, Saarbrücken

(Eingegangen am 10. Juni 1968)

¹ Hauptvortrag auf der Jahrestagung der GAMM 1968 in Prag.

Zusammenfassung — Summary

Übertragung automatentheoretischer Sätze auf Chomsky-Sprachen. Für endliche Automaten hat man ihrem Wesen nach zwei verschiedene Äquivalenzdefinitionen, die sich dann als gleich herausstellen. Die eine Definition beruht auf der Wirkung des Automaten nach außen, die andere geht von der inneren Struktur des Automaten aus. Letztere Definition wird hier R(eduktion)-Äquivalenz genannt.

Bei der Übertragung dieser Äquivalenzbegriffe auf CHOMSKY-Sprachen und verwandte formale Sprachen ergibt sich für die *R*-Äquivalenz eine ganze Reihe von Möglichkeiten. Es wird hier über die Ergebnisse berichtet, die in diesem Zusammenhang bis jetzt erzielt wurden.

Transfer of Automaton-theoretical Propositions to Chomsky Languages. For finite automata one has two different comprehensions of equivalence which prove to be equal. One of the definitions called *R*-equivalence rests on the internal structure of the automata. The other equivalence concerns those automata which cannot be distinguished by any experiments.

The extension of these equivalences to the CHOMSKY languages and related languages leads to different possibilities for the definitions of *R*-equivalences. The paper summarizes the results which are proved at this moment.

Einleitung

Die in der Überschrift gemeinten Sätze sind die Struktursätze über endliche Automaten. Es ist bekannt, daß Zerlegungen von endlichen Automaten in direkte Produkte zu günstigen technischen Realisierungen führen. Hierin besteht die Motivation der Theorie der Reduktion endlicher Automaten; zu einer Zerlegung eines endlichen Automaten in ein direktes Produkt gehört ein Paar von Reduktionen mit bestimmten Eigenschaften, und umgekehrt liefert jedes solches Reduktionspaar eine Zerlegung in ein direktes Produkt. Häufig ist der gegebene Automat nicht sofort zerlegbar, wohl aber ein „äquivalenter“, den man aus dem gegebenen durch eine „Erweiterung“ gewinnen kann, häufig verbunden mit „Einbettungen“. Nun lassen sich endliche Automaten als Sonderfälle von CHOMSKY-Sprachen beschreiben. Dies wirft die Frage auf, inwieweit sich diese algebraische

Strukturtheorie der endlichen Automaten auf diese Sprache übertragen läßt; vielleicht, daß sich die Analyse eines Wortes in einer Sprache zurückführen läßt auf die Analyse eines Wortes in zwei einfacheren Sprachen, die man durch „Reduktionen“ aus der gegebenen gewinnt.

Eine weitere Motivation für diese Übertragung liefert das Bedürfnis, die Äquivalenz von CHOMSKY-Sprachen so zu fassen, daß die Struktur der Sprache dabei eine wesentliche Rolle spielt.

Worin besteht das Äquivalenzproblem?

Ein Grund für das Interesse an CHOMSKY-Sprachen ist die Hoffnung der Linguisten, Grammatiken von gesprochenen Sprachen formal zu beschreiben, so daß sie einer exakten Untersuchung zugänglich werden. Angenommen, es liegen zwei nicht identische formale Beschreibungen der Grammatik der deutschen Sprache vor, dann wird man sich zuerst fragen, ob durch beide Beschreibungen die gleichen Sätze als formal richtig angesehen werden. Ist dies der Fall, dann nennt man beide formalen Sprachen (schwach) äquivalent. Nun passiert es häufig, daß Sätze verschiedene Deutungen zulassen; d. h. es läßt sich nicht immer eindeutig festlegen, was z. B. Subjekt und Objekt eines Satzes ist. Verlangt man nun auch noch, daß die formal beschriebenen Grammatiken für gleiche Sätze die gleichen Interpretationen zulassen, dann erhält man kleinere Äquivalenzklassen, bei deren Definition die Struktur der definierenden Sprache wesentlich eingeht.

Für Anwendungen wichtiger ist das entsprechende Problem für Programmierungssprachen. Man wünscht sich, formale vollständige Beschreibungen der Programme, die von einer Maschine bzw. von einem Compiler als formal richtig angenommen werden, um zu wissen, auf welchen Maschinen ein existierendes Programm eingegeben werden kann. Programmierungssprachen, die in dieser Weise übereinstimmen, heißen (schwach) äquivalent. Natürlich wünscht man sich mehr, nämlich daß die verschiedenen Maschinen die angenommenen Programme auch in gleicher Weise interpretieren. Die Interpretation der Programme geht aber über den „Analyse“-Prozeß, d. h. man sucht die „Ableitung“ des Programmes in der formalen Sprache und interpretiert das Programm unter Verwendung dieser Ableitungen; die Bedeutung (Interpretation) der einzelnen Schritte ist der Maschine mitgeteilt worden; aus diesen setzt sich die Gesamtinterpretation zusammen. Es ist klar, daß bei dieser Art Programme zu interpretieren, die Struktur der formalen Sprache ganz wesentlich eingeht.

Im folgenden Kapitel soll zunächst die Theorie der endlichen Automaten soweit skizziert werden, wie wir sie für die Übertragung auf CHOMSKY-Sprachen im Auge haben. Im darauffolgenden Kapitel skizzieren wir anhand des einfachsten Typs der CHOMSKY-Sprachen verschiedene Möglichkeiten der Übertragung. Im abschließenden Kapitel teilen wir in knapper Weise den jetzigen Stand des Versuches mit. Das hier geschilderte Programm wurde in [13] meines Wissens zum ersten Mal in strenge mathematische Form gebracht.

1. Automatentheorie

Physikalisch ist ein endlicher Automat (MOORE, MEALY) ein meist elektrisches Gerät, das eine Eingabevorrichtung, Ausgabevorrichtung und einen Speicher hat. Im Unterschied zu einer normalen elektronischen Rechenanlage geschieht die Aufnahme eines Signales, die Ausgabe eines Signales und eine eventuelle Änderung des Speicherinhaltes stets gleichzeitig, so daß die Länge der eingegebenen Signalfolgen stets gleich der Länge der zugehörigen Ausgabefolgen ist. Dies mag zunächst nur als formaler Unterschied erscheinen. Man kann aber der normalen Rechenmaschine formal ähnlichere Modelle von Automaten angeben (z. B. den „linear bounded automat“ von MYHILL [21] oder Mehrbänderautomaten von RABIN und SCOTT [22]), die Funktionen berechnen, die von keinem endlichen Automaten berechnet werden können. Mathematisch definiert man einen *endlichen Automaten* als ein Quintupel: $\mathfrak{A} = (X, Y, Z, \delta, \lambda)$, worin X, Y, Z endliche Mengen sind und $\delta: X \times Z \rightarrow Z$ und $\lambda: Z \rightarrow Y$ Abbildungen. X heißt das Eingabe-, Y das Ausgabe-Alphabet und Z die Zustandsmenge von \mathfrak{A} .

Wir zeichnen noch einen Anfangszustand $z_0 \in Z$ des Speichers aus, in den der Automat beim Einschalten übergeht. Ein solcher Automat wird beschrieben durch $\mathfrak{A} = (X, Y, Z, \delta, \lambda, z_0)$ und wird *initialer Automat* genannt.

Wir setzen

$$X^* = \{(x_1, \dots, x_n) \mid x_i \in X, n = 0, 1, 2, \dots\}$$

und definieren

$$(x_1, \dots, x_n) \cdot (x_{n+1}, \dots, x_m) = (x_1, \dots, x_m).$$

Dann wird X^* ein freies Monoid über X . Um die Wirkung von Eingangsfolgen auf den Automaten \mathfrak{A} zu beschreiben, setzen wir δ und λ zu Abbildungen $\lambda: X^* \times Z \rightarrow Z$ und $\mu: X^* \rightarrow Y$ fort, indem wir definieren:

$$\delta(e, z) = z, \mu(e) = \lambda(z_0),$$

$$\delta(x \cdot w, z) = \delta(x, \delta(w, z)), \mu(w) = \lambda(\delta(w, z))$$

für $x \in X, w \in X^*$.

Darin ist e die Einheit von X^* bzw. Y^* , d. h. jeweils die Folge der Länge 0. Nach Eingabe des Wortes w geht \mathfrak{A} in den Zustand $\delta(w, z_0)$ über und gibt beim nächsten Schritt $\mu(w)$ aus. Zwei Automaten \mathfrak{A} und \mathfrak{A}' sind von einem Benutzer nicht unterscheidbar und deshalb austauschbar, wenn $\mu = \mu'$ ist. Dies nehmen wir als Grund für eine erste Äquivalenzdefinition:

Definition: \mathfrak{A} heißt äquivalent zu \mathfrak{A}' genau dann, wenn $\mu = \mu'$ ist.

Für einen Ingenieur ist ein Automat von der Aufgabenstellung her also nur bis auf diese Äquivalenz definiert. (Wir sehen dabei ab von einer Reihe weiterer Bedingungen, die durch die Physik hereinkommen.) Damit stellt sich für ihn die Aufgabe, aus den unendlichen Äquivalenzklassen

einen Repräsentanten als vorteilhaft auszuwählen. Hierbei spielt natürlich die innere Struktur des Automaten die entscheidende Rolle. Um Automaten bezüglich ihrer Struktur vergleichen zu können, definiert man strukturerhaltende Abbildungen, die hier Reduktionen heißen.

Definition: Eine Abbildung $\varphi: \mathcal{A} \rightarrow \mathcal{A}'$ heißt eine Reduktion von $\mathcal{A} = (X, Y, Z, \delta, \lambda, z_0)$ auf $\mathcal{A}' = (X, Y, Z', \delta', \lambda', z'_0)$, wenn (1) und (2) gilt.

- (1) $\varphi: Z \rightarrow Z'$ ist surjektiv und $\varphi(z_0) = z'_0$.
- (2) a) $\varphi(\delta(x, z)) = \delta'(x, \varphi(z))$,
- b) $\lambda'(x, \varphi(z)) = \lambda(x, z)$.

Wir schreiben dafür kurz $\varphi: \mathcal{A} \rightarrow \mathcal{A}'$.

Man sieht sofort:

Satz 1: Aus $\varphi: \mathcal{A} \rightarrow \mathcal{A}'$ folgt, \mathcal{A} äquivalent \mathcal{A}' .

Wir setzen nun

$$\mathfrak{R}(\mathcal{A}) = \{\mathcal{A}' \mid \text{Es gibt } \varphi: \mathcal{A} \rightarrow \mathcal{A}'\}.$$

Dann gilt

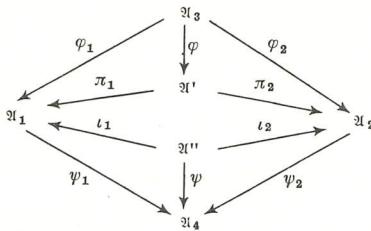
Hilfssatz 1: Zu je zwei Automaten $\mathcal{A}_1, \mathcal{A}_2 \in \mathfrak{R}(\mathcal{A})$ gibt es bis auf Isomorphie eindeutig bestimmte Automaten \mathcal{A}' und \mathcal{A}'' und Reduktionen

$$\mathcal{A}_1 \xleftarrow{\pi_1} \mathcal{A}' \xrightarrow{\pi_2} \mathcal{A}_2 \quad \text{und} \quad \mathcal{A}_1 \xrightarrow{\iota_1} \mathcal{A}'' \xleftarrow{\iota_2} \mathcal{A}_2,$$

so daß es zu jedem Reduktionspaar

$$\mathcal{A}_1 \xleftarrow{\varphi_1} \mathcal{A}_3 \xrightarrow{\varphi_2} \mathcal{A}_2 \quad \text{und} \quad \mathcal{A}_1 \xrightarrow{\psi_1} \mathcal{A}_4 \xleftarrow{\psi_2} \mathcal{A}_2$$

je genau eine Reduktion $\varphi: \mathcal{A}_3 \rightarrow \mathcal{A}'$ und $\psi: \mathcal{A}'' \rightarrow \mathcal{A}_4$ gibt, so daß das folgende Diagramm kommutativ ist:



Mit anderen Worten: $\mathfrak{R}(\mathcal{A})$ ist bei Identifizierung der isomorphen Elementen und den Verknüpfungen $\mathcal{A}_1 \cup \mathcal{A}_2 = \mathcal{A}''$ und $\mathcal{A}_1 \cap \mathcal{A}_2 = \mathcal{A}'$ ein Verband.

Wir definieren nun mittels den Reduktionen eine Äquivalenz.

Definition: \mathcal{A} heißt R(eduktions)-äquivalent zu \mathcal{A}' genau dann, wenn es eine Kette $\mathcal{A}_1 = \mathcal{A}, \dots, \mathcal{A}_k = \mathcal{A}'$ von Automaten mit Reduktion $\varphi_i: \mathcal{A}_i \rightarrow \mathcal{A}_{i+1}$ oder $\varphi_i: \mathcal{A}_{i+1} \rightarrow \mathcal{A}_i$ gibt.

Diese Äquivalenz gründet sich auf die innere Struktur der endlichen Automaten. Aus Satz 1 und Hilfssatz 1 folgt:

Satz 2: Ist \mathfrak{A} R-äquivalent zu \mathfrak{A}' , dann folgt:

- 1) \mathfrak{A} äquivalent \mathfrak{A}' .
- 2) Zu \mathfrak{A} und \mathfrak{A}' gibt es einen Automaten \mathfrak{A}'' und Reduktionen φ, φ' mit $\varphi: \mathfrak{A} \rightarrow \mathfrak{A}'', \varphi': \mathfrak{A}' \rightarrow \mathfrak{A}''$.

Man kommt bei der R-Äquivalenz also immer mit Ketten der Länge 3 aus. Nun erhebt sich die Frage nach der Umkehrung dieses Satzes. Ist Satz 2.1 umkehrbar, dann erhalten wir eine vollständige Übersicht über die Strukturen einander äquivalenter Automaten. Im Falle der nicht initialen Automaten ist die Umkehrung richtig, in unserem Falle noch nicht. Um die Umkehrung formulieren zu können, treffen wir eine weitere

Definition: $\mathfrak{A} = (X, Y, Z, \delta, \lambda, z_0)$ heißt zusammenhängend genau dann, wenn zu $z \in Z$ ein $w \in X^*$ existiert mit $\delta(w, z_0) = z$.

Das heißt, daß jeder mögliche Zustand des Speichers bei geeigneten Eingangsfolgen auch angenommen wird.

Nun kann man zeigen, daß gilt:

Satz 3: Sind \mathfrak{A} und \mathfrak{A}' zusammenhängend und ist \mathfrak{A} äquivalent zu \mathfrak{A}' , dann ist auch \mathfrak{A} R-äquivalent zu \mathfrak{A}' .

Nun gibt es zu jedem Automaten \mathfrak{A} einen zusammenhängenden Unterautomaten \mathfrak{A}' .

Definition: $\mathfrak{A}' = (X, Y, Z', \delta', \lambda', z_0)$ heißt Unterautomat von $\mathfrak{A} = (X, Y, Z, \delta, \lambda, z_0)$ genau dann, wenn 1) und 2) gilt.

- 1) $Z' \subset Z$,
- 2) $\delta'(x, z) = \delta(x, z), \lambda(z) = \lambda(z')$ für $z \in Z'$.

In Zeichen: $\mathfrak{A}' \subset \mathfrak{A}$.

Hilfssatz 2: Ist $\mathfrak{A}' \subset \mathfrak{A}$, dann ist \mathfrak{A}' äquivalent zu \mathfrak{A} . Zu jedem \mathfrak{A} gibt es einen zusammenhängenden Unterautomaten $\mathfrak{A}' \subset \mathfrak{A}$.

Zusammenfassend können wir folgenden Satz aussprechen:

Satz 4: \mathfrak{A} ist genau dann äquivalent zu \mathfrak{A}' , wenn es Automaten $\mathfrak{A}_1, \mathfrak{A}_2, \mathfrak{A}_3$ gibt und Reduktionen φ, ψ , so daß gilt:

$$\mathfrak{A} \xrightarrow{\varphi} \mathfrak{A}_1 \xleftarrow{\psi} \mathfrak{A}_2 \subset \mathfrak{A}'.$$

Nennen wir Automaten *verwandt*, die sich durch eine Kette dieser Art, d. h. durch „Einbettungen“ und Reduktionen, verbinden lassen, dann haben wir: \mathfrak{A} ist genau dann äquivalent zu \mathfrak{A}' , wenn \mathfrak{A} und \mathfrak{A}' verwandt sind. Im nächsten Abschnitt interessiert uns nur der Sonderfall $Y = \{0, 1\}$.

Es gilt:

Hilfssatz 3: $\mathfrak{A} = (X, \{0, 1\}, Z, \delta, \lambda, z_0)$ ist genau dann äquivalent zu $\mathfrak{A}' = (X, \{0, 1\}, Z', \delta', \lambda', z'_0)$, wenn $\mu^{-1}(1) = \mu'^{-1}(1)$ ist.

Damit haben wir bei gleicher Bezeichnung:

Satz 5: \mathfrak{A} ist genau dann verwandt zu \mathfrak{A}' , wenn $\mu^{-1}(1) = \mu'^{-1}(1)$ ist.

Aus Hilfssatz 1 ergibt sich nun, daß sich die ganze Äquivalenzklasse eines Automaten aus einem Einzigen durch „Erweiterungen“ (= Umkehrung der Reduktion) und Einbettungen aufzählen läßt. Hieran schließen im Grunde genommen die Algorithmen zur „Realisierung“ von endlichen Automaten an, die von J. HARTMANIS, R. E. STEARNS, Z. KOHAVI, G. BEYER, S. GINSBURG und H. WALTER entwickelt wurden [1], [9], [10], [11], [16], [27]. Zur algebraischen Theorie der endlichen Automaten sehe man z. B. [3], [7], [17].

2. Einseitig lineare Sprachen

Wir geben nun eine andere Beschreibung der Mengen $\mu^{-1}(1)$, die einen Sonderfall der CHOMSKY-Sprachen darstellt. Diesen Sonderfall verallgemeinern wir in diesem Abschnitt auf die einseitig linearen CHOMSKY-Sprachen, auf die wir dann die Ergebnisse des vorherigen Kapitels übertragen.

Wir legen dem Folgenden das freie Monoid A^* und $A = X \cup Z$ zugrunde und setzen stets $X \cap Z = \emptyset$ voraus. Sei $\mathfrak{A} = (X, \{0, 1\}, Z, \delta, \lambda, z_0)$ gegeben, dann definieren wir:

$$P_{\mathfrak{A}} = \{(z, x z') \mid z \in Z, z' \in Z, x \in X, \delta(x, z) = z'\} \cup \{(z, e) \mid \lambda(z) = 1\}.$$

Es ist $P_{\mathfrak{A}} \subset (Z \times X \cdot Z) \cup (Z \times \{e\})$.

Wir schreiben auch $z \rightarrow x z$ und $z \rightarrow e$ für $(z, x z) \in P_{\mathfrak{A}}$ bzw. $(z, e) \in P_{\mathfrak{A}}$. Weiter schreiben wir $z \rightarrow x_1 \dots x_k z_k$ genau dann, wenn es eine Kette $z \rightarrow x_1 z_1, z_1 \rightarrow x_2 z_2, \dots, z_{k-1} \rightarrow x_k z_k$ gibt. Wir schreiben $z \rightarrow x_1 \dots x_k$ genau dann, wenn es ein z' gibt mit $z \rightarrow x_1 \dots x_k z'$ und $z' \rightarrow e$. Man erhält also $x_1 \dots x_k$ aus z , indem man „Produktionen“ aus P in geeigneter Weise ineinander einsetzt. Eine Folge von solchen Einsetzungen heißt eine *Ableitung*.

Man nennt nun

$$\mathfrak{L}_{\mathfrak{A}} = (A, X, P_{\mathfrak{A}}, z_0)$$

eine *links lineare CHOMSKY-Sprache* [5, 6], A das Alphabet, X das Endalphabet, $P_{\mathfrak{A}}$ das Produktionssystem und z_0 die Verankerung oder das Anfangselement der Sprache. Die Menge

$$|\mathfrak{L}_{\mathfrak{A}}| = \{w \in X^* \mid z_0 \rightarrow w\}$$

nennen wir die *Satzmenge* der Sprache.

Dann erkennt man ohne weiteres die Richtigkeit von

Satz 6:

$$|\mathfrak{L}_{\mathfrak{A}}| = \mu^{-1}(1).$$

Man sieht, daß sich aus $\mathfrak{L}_{\mathfrak{A}}$ der Automat \mathfrak{A} zurückgewinnen läßt. δ und λ werden durch $P_{\mathfrak{A}}$ in eindeutiger Weise beschrieben. Will man also strukturerhaltende Abbildungen auf solchen Sprachen erklären, wird anstelle der Verträglichkeit mit δ und λ eine mit $P_{\mathfrak{A}}$ zu treten haben.

Wir verallgemeinern zunächst:

Definition: $\mathfrak{L} = (A, X, P, z_0)$ heißt eine rechtslineare CHOMSKY-Sprache, wenn 1) und 2) gilt:

$$1) \quad X \neq \emptyset, z_0 \in A - X = Z, X \subset A,$$

$$2) \quad P \subset Z \times X^* \cdot Z \cup Z \times X^* \text{ ist eine endliche Menge.}$$

Es gilt $w_0 z \xrightarrow[P]{} w_0 w_1 \dots w_k z'$ mit $z' \in Z \cup \{e\}$, $z \in Z$, $w_i \in X^*$ genau dann, wenn es eine Kette

$$(z, w_1 z_1), \dots, (z_{k-1}, w_k z') \in P \text{ gibt.}$$

Die Folge

$$w_0 z \xrightarrow[P]{} w_0 w_1 z_1 \xrightarrow[P]{} w_0 w_1 w_2 z_2 \xrightarrow[P]{} \dots \xrightarrow[P]{} w_0 w_1 \dots w_k z'$$

heißt eine Ableitung von $w_0 w_1 \dots w_k z'$ aus $w_0 z$.

$$|\mathfrak{L}| = \{w \in X^* \mid z_0 \xrightarrow[P]{} w\}$$

heißt die Satzmenge von \mathfrak{L} .

Definition: Seien $\mathfrak{L}, \mathfrak{L}'$ rechtslineare CHOMSKY-Sprachen. \mathfrak{L} heißt äquivalent zu \mathfrak{L}' genau dann, wenn $|\mathfrak{L}| = |\mathfrak{L}'|$ ist.

Für den Sonderfall, daß es Automaten \mathfrak{A} und \mathfrak{A}' gibt mit $\mathfrak{L}_{\mathfrak{A}} = \mathfrak{L}$ und $\mathfrak{L}_{\mathfrak{A}'} = \mathfrak{L}'$, stimmt diese Äquivalenzdefinition nach Satz 5 mit der für endliche Automaten überein. Nun übertragen wir den Begriff der Reduktion.

Sei $A = X \cup Z$, $A' = X \cup Z'$, $X \cap Z = X \cap Z' = \emptyset$ und $\varphi'_1: A \rightarrow A'$ eine surjektive Abbildung mit $\varphi'_1(x) = x$ für $x \in X$. Wir setzen φ'_1 zu dem (einzig bestimmen) Homomorphismus $\varphi_1: A^* \rightarrow A'^*$ fort. Weiter definieren wir $\varphi_2(w_1, w_2) = (\varphi_1(w_1), \varphi_1(w_2))$ für $w_1, w_2 \in A^*$ und erhalten so eine Abbildung $\varphi_2: A^* \times A^* \rightarrow A'^* \times A'^*$.

Definition: $\varphi = (\varphi_1, \varphi_2)$ heißt eine Reduktion von \mathfrak{L} auf \mathfrak{L}' ($\varphi: \mathfrak{L} \rightarrow \mathfrak{L}'$), wenn (R 1), (R 2), (R 3) erfüllt ist.

$$(R \ 1) \quad \varphi_1(Z) = Z', \varphi_1(z_0) = z'_0,$$

$$(R \ 2) \quad \varphi_2(P) = P',$$

(R 3) Aus $\varphi(z_1) = \varphi(z_2)$ und $(z_1, w) \in P$ folgt, daß es ein $(z_2, w') \in P$ gibt mit $\varphi_2(z_2, w') = \varphi_2(z_1, w)$.

Man könnte zunächst geneigt sein, auf (R 3) zu verzichten. Das folgende Beispiel zeigt, daß dann der unten folgende Satz 7 nicht gilt.

Beispiel:

$$(1) \quad X = \{x\}, Z = \{z_0, z_1\},$$

$$P = \{(z_0, x z_1), (z, e), (z_0, e)\}, \mathfrak{L} = (X \cup Z, X, P, z_0),$$

$$(2) \quad \mathfrak{L}' = (X \cup Z', X, P', z'_0), Z'_0 = \{z'_0\}, P' = \{(z'_0, x z'_0), (z'_0, e)\}.$$

Setzen wir nun

$$\varphi_1(z_0) = \varphi_1(z_1) = z'_0,$$

dann wird (R 1) und (R 2) erfüllt, aber (R 3) ist verletzt, da in P keine Produktion $(z_1, x z_0)$ oder $(z_1, x z_1)$ auftritt. Man erkennt $|\mathfrak{L}| = \{e, x\}$ und $|\mathfrak{L}'| = \{e, x, x^2, x^3, \dots\} = \{x\}^*$.

Man kommt also mit (R 1) und (R 2) nicht aus, aber es genügen auch schon Abschwächungen von (R 3), wie man aus einer Ergänzung von P durch $(z_0, x z_0)$ erkennt. Wir werden hierauf später zurückkommen. Unsere Definition erlaubt es uns aber, den folgenden Satz zu beweisen:

Satz 7: Ist $\varphi: \mathfrak{L} \rightarrow \mathfrak{L}'$, dann ist $|\mathfrak{L}| = |\mathfrak{L}'|$; d. h. \mathfrak{L} äquivalent \mathfrak{L}' .

Wir definieren nun die Reduktionsäquivalenz in Analogie zu Abschnitt 1.

Definition: \mathfrak{L} heißt R-äquivalent zu \mathfrak{L}' genau dann, wenn es eine Kette $\mathfrak{L}_0 = \mathfrak{L}, \mathfrak{L}_1, \dots, \mathfrak{L}_k = \mathfrak{L}'$ und Reduktionen $\varphi_i: \mathfrak{L}_i \rightarrow \mathfrak{L}_{i+1}$ oder $\varphi_i: \mathfrak{L}_{i+1} \rightarrow \mathfrak{L}_i$ ($i = 0, \dots, k-1$) gibt.

Korollar zu Satz 7: Ist \mathfrak{L} R-äquivalent zu \mathfrak{L}' , dann ist \mathfrak{L} äquivalent zu \mathfrak{L}' .

Wie wir in Abschnitt 1 gesehen haben, ist dieser Satz nicht ohne weiteres umkehrbar. Wir definieren zu diesem Zweck analog, wann \mathfrak{L} Untersprache von \mathfrak{L}' ist.

Definition: $\mathfrak{L} = (A, X, P, z_0)$ heißt Untersprache von $\mathfrak{L}' = (A', X, P', z_0)$, wenn (U 1) und (U 2) gilt.

(U 1) $Z \subset Z'$.

(U 2) Ist $z \in Z$ und $w \in A^*$ und gilt $z_1 \xrightarrow{P'} w$, dann gilt auch $z_1 \xrightarrow{P} w$.

Zu Zeichen $\mathfrak{L} \subset \mathfrak{L}'$.

In [13] fehlte (U 2).

Definition: \mathfrak{L} heißt zusammenhängend, falls es zu jedem $z \in Z$ ein $w \in X^*$ gibt mit $z_0 \xrightarrow{P} w z$ und ein $u \in X^*$ mit $z \xrightarrow{P} u$.

Es gilt:

Hilfssatz 4: Ist $\mathfrak{L} \subset \mathfrak{L}'$, dann ist $|\mathfrak{L}| = |\mathfrak{L}'|$.

Hilfssatz 5: Zu jeder Sprache \mathfrak{L} gibt es eine zusammenhängende Untersprache \mathfrak{L}' .

Definition: \mathfrak{L} heißt R-verwandt zu \mathfrak{L}' genau dann, wenn es eine Kette $\mathfrak{L}_0 = \mathfrak{L}, \mathfrak{L}_1, \dots, \mathfrak{L}_k = \mathfrak{L}'$ gibt mit $\mathfrak{L}_i \subset \mathfrak{L}_{i+1}$ oder $\mathfrak{L}_{i+1} \subset \mathfrak{L}_i$ oder \mathfrak{L}_i R-äquivalent zu \mathfrak{L}_{i+1} ($i = 0, \dots, k-1$).

Nun können wir den Hauptsatz dieses Kapitels formulieren, der die angestrebte Übertragung des Satzes 4 auf die rechtslinearen CHOMSKY-Sprachen enthält.

Satz 8: Sind \mathfrak{L} und \mathfrak{L}' rechtslineare CHOMSKY-Sprachen, dann gilt: \mathfrak{L} ist genau dann äquivalent zu \mathfrak{L}' , wenn \mathfrak{L} R-verwandt zu \mathfrak{L}' ist.

Man kann zeigen, daß für \mathfrak{L} äquivalent \mathfrak{L}' stets eine Verwandtschaft folgender Art besteht:

$$\mathfrak{L} \supset \mathfrak{L}_1 \subset \mathfrak{L}_2 \leftarrow \mathfrak{L}_3 \rightarrow \mathfrak{L}_4 \Leftrightarrow \mathfrak{L}'_4 \leftarrow \mathfrak{L}'_3 \rightarrow \mathfrak{L}'_2 \supset \mathfrak{L}'_1 \subset \mathfrak{L}'.$$

Jedes Glied \mathfrak{L}_i ist aus \mathfrak{L} und jedes Glied \mathfrak{L}'_i aus \mathfrak{L}' effektiv konstruierbar. Zu \mathfrak{L}_4 bzw. \mathfrak{L}'_4 gibt es Automaten \mathfrak{A} und \mathfrak{A}' mit $\mathfrak{L}_{\mathfrak{A}} = \mathfrak{L}_4$ und $\mathfrak{L}_{\mathfrak{A}'} = \mathfrak{L}'_4$, so daß \mathfrak{L}_4 isomorph \mathfrak{L}'_4 entscheidbar ist.

Im nächsten Abschnitt untersuchen wir die Übertragung dieser Begriffe auf allgemeine CHOMSKY-Sprachen.

3. Verallgemeinerung auf beliebige Chomsky-Sprachen

Sei $X \subset A$, A eine endliche Menge, $X \neq \emptyset$, $z_0 \in Z = A - X$. Weiter sei $P \subset (Z^* - \{e\}) \times A^*$ eine endliche Menge. Mittels P definieren wir eine Relation \xrightarrow{P} auf A^* : Es gilt für $w, w' \in A^*$ genau dann $w \xrightarrow{P} w'$, wenn es eine Folge $w = w_1, \dots, w_k = w'$ gibt mit $w_1 = u_1 p_1 v_1$, $w_2 = u_1 q_1 v_1 = u_2 p_2 v_2, \dots, w_{k-1} = u_{k-1} p_{k-1} v_{k-1}$, $w_k = u_{k-1} q_{k-1} v_{k-1}$, worin $(p_i, q_i) \in P$ ist für $i = 1, \dots, k-1$, oder wenn $w = w'$ ist.

Definition: $\mathfrak{L} = (A, X, P, z_0, \xrightarrow{P})$ oder kürzer $\mathfrak{L} = (A, X, P, z_0)$ heißt CHOMSKY-Sprache vom Typ 0. Abkürzung $\mathfrak{L} \in Ch\text{-}0$.

$|\mathfrak{L}| = \{w \in X^* \mid z_0 \xrightarrow{P} w\}$ heißt die Satzmengen von \mathfrak{L} .

CHOMSKY hat noch die folgenden Unterklassen betrachtet:

Definition:

- 1) $\mathfrak{L} \in Ch\text{-}1 \subset Ch\text{-}0 \iff ((p, q) \in P \implies p = u z v, q = u w v \text{ mit } u, v, w \in A^*, z \in Z, w \neq e)$.
- 2) $\mathfrak{L} \in Ch\text{-}2 \subset Ch\text{-}1 \iff ((p, q) \in P \implies p \in Z, q \in A^*, q \neq e)$.
- 3) $\mathfrak{L} \in (Ch\text{-linear}) \subset Ch\text{-}2 \iff ((p, q) \in P \implies p \in Z, q = u z' v, u, v \in X^*, z' \in Z \cup \{e\})$.
- 4) *Ch-3 ist die Klasse der einseitig linearen Sprachen, d. h. der linkslinearen oder rechtslinearen, die wir im vorherigen Abschnitt behandel haben.*

Man läßt bei *Ch-2* und *Ch-lin.* oft auch $w = e$ bzw. $z' = e$ zu. Der Unterschied ist bezüglich der erzeugten Satzmengen geringfügig: Beide Klassen definieren die beiden Satzmengen, wenn man davon absieht, daß in den Satzmengen der einen Klassen e vorkommen kann und in denen der anderen Klassen nicht.

Wie man die Äquivalenz von Sprachen zu definieren hat, ist klar. Zwei Sprachen \mathfrak{L} und \mathfrak{L}' heißen äquivalent, wenn $|\mathfrak{L}| = |\mathfrak{L}'|$ ist.

Seien \mathfrak{L} und \mathfrak{L}' CHOMSKY-Sprachen und $\varphi_1: A^* \rightarrow A'^*$ und $\varphi_2: (A^* \times A^*) \rightarrow (A'^* \times A'^*)$ wie im vorigen Kapitel erklärt.

Definition: $\varphi = (\varphi_1, \varphi_2)$ heißt eine (n, l) -Reduktion, wenn (H 1), (H 2) und (H 3) gilt:

- (H 1) $\varphi_1(Z) = Z'$, $\varphi_1(z_0) = z'_0$, $\varphi_1(x) = x$ für $x \in X$.
(H 2) $\varphi_2(P) = P'$.
(H 3) Ist $(p, q) \in P$ und $\varphi_1(p) = \varphi_1(p')$, dann existiert $(p', q') \in P$ mit $\varphi_2(p, q) = \varphi_2(p', q')$.

Wir schreiben in diesem Fall $\varphi: \mathfrak{L} \rightarrow \mathfrak{L}'$.

Die Bezeichnung (n, l) -Reduktion werde ich später erläutern. Es gilt der zu Satz 7 analoge

Satz 9: Aus $\varphi: \mathfrak{L} \rightarrow \mathfrak{L}'$ folgt $|\mathfrak{L}| = |\mathfrak{L}'|$.

Ebenso leicht wie die Reduktion überträgt sich der Begriff der Untersprache:

Definition: $\mathfrak{L} \subset \mathfrak{L}'$ gilt genau dann, wenn (U 1) und (U 2) gilt:

- (U 1) $X = X'$, $z_0 = z'_0$, $Z \subset Z'$.
(U 2) Ist $w_1, w_2 \in Z^*$ und ist $w_1 \xrightarrow{P'} w_2$, dann gilt auch $w_1 \xrightarrow{P} w_2$.

Die Definition der Reduktionsäquivalenz und der R-Verwandtschaft lassen sich wörtlich übernehmen.

Man kann anhand einfacher Beispiele zeigen, daß die (n, l) -Reduktionen einer Sprache keinen Verband bilden, weshalb sich die zu R-Äquivalenzen gehörigen Reduktionsketten nicht ohne weiteres verkürzen lassen. Durch eine zusätzliche Bedingung kann man die Menge der Reduktionen einschränken, daß man einen Verband erhält. In [15] sind diese Reduktionen als (n, l) -Reduktion bezeichnet worden. (H 1), (H 2), (H 3) entsprechen den Bedingungen (P 1), (P 2) dort. (P 3) ist die zusätzliche weitere Bedingung. Es führt in diesem Vortrag zu weit, diese zu formulieren.

Für welche dieser Klassen der Hauptsatz des vorherigen Kapitels gilt, ist noch offen. Es liegen in dieser Richtung aber Teilergebnisse vor, über die nun noch berichtet werden soll. Hierzu ist eine weitere Verfeinerung unserer Begriffe notwendig, die wir nun durchführen.

Definieren wir für $(w_1, v_1), (w_2, v_2) \in A^* \times A^*$ die Verknüpfung „ \circ “ durch $(w_1, v_1) \circ (w_2, v_2) = (w_1, v_2)$ für $v_1 = w_2$, dann erhalten wir eine Kategorie [2] bezüglich „ \circ “. Setzen wir $(w_1, v_1) \times (w_2, v_2) = (w_1 w_2, v_1 v_2)$, dann erhalten wir bezüglich „ \times “ ein Monoid. Beide Verknüpfungen genügen der Relation

$$\begin{aligned} ((w_1, v_1) \times (w_2, v_2)) \circ ((v_1, u_1) \times (v_2, u_2)) &= \\ = ((w_1, v_1) \circ (v_1, u_1)) \times ((w_2, v_2) \circ (v_2, u_2)). \end{aligned}$$

Eine solche Kategorie \mathfrak{C} ist ein triviales Beispiel für eine Monoidkategorie oder X -Kategorie [12], [14]. Man sieht leicht, daß „ \rightarrow “ bezüglich dieser Verknüpfungen abgeschlossen ist, und darüber hinaus, daß „ \rightarrow “ aus P mittels den beiden Verknüpfungen erzeugt wird. Die durch P erzeugte Unterkategorie mit A^* als Objektmenge nennen wir \mathfrak{C}_P . Faßt man den Begriff der „Ableitung“, der für Anwendungen sehr wichtig ist, genau,

dann wird man in natürlicher Weise zu der Konstruktion einer über der vorigen X -Kategorie *freien X-Kategorie* \mathfrak{F} geführt, die durch (A, X, P) bis auf Isomorphie eindeutig bestimmt ist.

Die oben definierte Reduktion einer Sprache \mathfrak{L}_1 auf eine Sprache \mathfrak{L}_2 läßt sich nun beschreiben durch einen *Funktor* zwischen den zugehörigen Kategorien \mathfrak{F}_1 und \mathfrak{F}_2 . Die oben definierte Reduktion heißt l (ängenerhaltende) Reduktion, weil sie die Länge der Ableitungen, d. h. der Morphinismen der freien Kategorie enthält. Das n kommt von normal und dient der Auszeichnung einer speziellen Klasse von l -Reduktionen, für die der zu Satz 9 analoge Satz gilt.

Wir betrachten noch einige allgemeinere Fassungen des Reduktionsbegriffes:

Eine Reduktion soll erstens die Satzmenge einer Sprache festlassen und zweitens die Struktur der Sprache in gewissem Umfang erhalten.

Dementsprechend definieren wir: Sind \mathfrak{L} und \mathfrak{L}' zwei CHOMSKY-Sprachen und \mathfrak{F} bzw. \mathfrak{F}' die zugehörigen X -Kategorien, dann definieren wir: Ein Funktor $\varphi = (\mathfrak{F}_1, \mathfrak{C}_2, \varphi_1, \varphi_2)$ bzw. $\varphi = (\mathfrak{F}_1, \mathfrak{F}_2, \varphi_1, \varphi_2)$ heißt eine schwache Reduktion bzw. eine Reduktion von \mathfrak{L}_1 auf \mathfrak{L}_2 , falls $\varphi_1(z_0) = z'_0$ ist, $\varphi_1 | X = i d_X$ und $|\mathfrak{L}_1| = |\mathfrak{L}_2|$.

Die beiden Reduktionen unterscheiden sich dadurch, daß die zweite die Ableitungsstruktur stärker berücksichtigt.

Die unschöne Bedingung $|\mathfrak{L}_1| = |\mathfrak{L}_2|$ kann man durch Bedingungen über φ_2 ersetzen.

Hinreichend ist die Forderung

$$\varphi_2(Mor \mathfrak{F}_1(z_0, X^*)) = Mor \mathfrak{F}_2(z'_0, X^*).$$

Die Forderung ist nicht notwendig. Sie enthält noch eine Bedingung über die Ableitungsstruktur [23].

Eine weitere Verschärfung erhält man, indem man zusätzlich fordert, daß die Bildmenge von \mathfrak{F}_1 wieder eine Kategorie [15] ist. In diesem Fall kann man den Funktor zerlegen in ein Produkt eines Funktors mit surjektiver Abbildung φ_2 mit einer *vollen Einbettung* [19]. Wir nennen \mathfrak{L} eine *Untersprache* von \mathfrak{L}' , wenn sich die zu \mathfrak{L} gehörige X -Kategorie voll in die zu \mathfrak{L}' gehörige einbetten läßt ($\mathfrak{L} \subset \mathfrak{L}'$).

Man kann nun folgende Sätze zeigen:

1. Für *Ch-lin.* lassen sich die Reduktionen effektiv angeben. Die Sprache \mathfrak{L}_4 in der Verwandtschaftskette am Ende von Abschnitt 2 ist „determiniert“. Dieser Begriff läßt sich auf *Ch-lin.* so übertragen, daß die analoge Konstruktion

$$\mathfrak{L} \supset \mathfrak{L}_1 \subset \mathfrak{L}_2 \leftarrow \mathfrak{L}_3 \rightarrow \mathfrak{L}_4$$

effektiv durchführbar ist. Darin ist „ \subset “ die volle Einbettung. Es gilt aber nicht allgemein \mathfrak{L}_4 isomorph \mathfrak{L}'_4 . Ob \mathfrak{L} und \mathfrak{L}' verwandt sind, wenn sie äquivalent sind, ist noch offen [15].

2. Für *Ch-2* kann man noch generell entscheiden, ob ein über das Erzeugendensystem definierter längenerhaltender Funktor zwischen den

freien X -Kategorien surjektiv ist. Man kann weiter generell entscheiden $\varphi_2(Mor \mathfrak{F}_1(z_0, X^*)) = Mor \mathfrak{F}_2(z'_0, X^*)$ [23]. Die längenerhaltende Reduktion einer Sprache dieses Typs läßt sich also noch generell durchführen.

3. Für $Ch\text{-}1$ ist die Reduktion nicht mehr generell durchführbar [24].

Die in dem ersten Teil des Vortrages aufgeworfene Frage nach Zerlegungen von CHOMSKY-Sprachen in direkte Produkte wird in [25], [26] untersucht. Es hat sich ergeben, daß in der Kategorie der X -Kategorien mit den Funktoren als Morphismen kein direktes Produkt existiert, wohl aber pull-back-Konstruktionen über der Kategorie der Netze [12] möglich sind. Verschiedene andere Typen von Zerlegungen formaler Sprachen werden in [15a] untersucht, die auf der Konstruktion der direkten Summe beruhen, die in dieser Kategorie existieren.

Bemerkung:

Die zuletzt genannten Arbeiten von C.-P. SCHNORR und H. WALTER wurden von der Deutschen Forschungsgemeinschaft unter Ho 251/2 gefördert.

Auf der GAMM-Tagung machte mich anlässlich dieses Vortrages Herr K. CULIK auf eine Homomorphiedefinition für CHOMSKY-Sprachen aufmerksam in „On Some Transformations in Contextfree Grammers and Languages“ (Czechoslovak Mathematical Journal 17 (1967). Dieser Homomorphiebegriff ist ein Sonderfall unserer (n, l) -Reduktion.)

Literatur

- [1] BEYER, G.: Die Erweiterung von Schaltwerken. Techn. Bericht der Fa. Telefunken 1965.
- [2] BRINKMANN, H.-B., und D. PUPPE: Kategorien und Funktoren. Lecture Notes in Mathematics. (Ed. A. DOLD, R. ECKMANN). Springer-Verlag. 1966.
- [3] BÜCHT, J. R.: Theorie der endlichen Automaten. Vorlesung an der Universität in Mainz (1961–1962). Nicht veröffentlicht.
- [4] CHOMSKY, N.: Three Models for the Description of Language. IRE Trans. on Inform. Theory, IT-2, 1956.
- [5] CHOMSKY, N., and G. A. MILLER: Finite State Languages. Information and Control 1, (1958).
- [6] CHOMSKY, N., and M. P. SCHÜTZENBERGER: The Algebraic Theory of Context-free Languages, in: Computer Programming and Formal Systems. (Ed.: BRAFFORT and HIRSCHBERG). Amsterdam: North-Holland Publishing Comp. 1963.
- [7] DEUSSEN, P.: On the Algebraic Theorie of Finite Automata. ICC Bulletin 4, (1966).
- [8] EICKEL, J., W. LOHNER und H. LANGMAACK: Kapitel über Mehrdeutigkeit von CHOMSKY-0-Sprachen. Erscheint in Lecture Notes in Mathematics. (Ed.: A. DOLD, R. ECKMANN). Springer-Verlag.
- [9] GINSBURG, S.: An Introduction to Mathematical Machine Theory. Addison-Wesley. 1962.
- [10] HARTMANIS, J.: On the State Assignment Problem for Sequential Machines I, IRE-Trans. EC-10, (1961).
- [11] HARTMANIS, J., and R. E. STEARNS: On the State Assignment Problem for Sequential Machines II. IRE-Trans. EC-10, (1961).
- [12] HÖRZ, G.: Algebraisierung des Syntheseproblems von Schaltkreisen. EIK 1, (1965).

- [13] HOTZ, G.: Homomorphie und Äquivalenz formaler Sprachen. 3. Kolloquium über Automatentheorie. (Ed.: W. HÄNDLER, E. PESCHL, H. UNGER). Basel: Birkhäuser-Verlag, 1967.
- [14] HOTZ, G.: Eindeutigkeit und Mehrdeutigkeit formaler Sprachen. EIK 2, (1966).
- [15] HOTZ, G.: Sprachen mit endlich vielen Zuständen. Math. Zeitschrift 104, (1968).
- [15a] HOTZ, G.: Erzeugung formaler Sprachen durch gekoppelte Ersetzungen. 4. Kolloquium über Automatentheorie vom 5.—6. 10. 1967 in München. (Ed.: F. L. BAUER, K. SAMELSON. Math. Institut der TH München).
- [16] KOHAVI, Z.: Secondary State Assignment for Sequential Machines. IEEE-Trans. EC-13, (1964).
- [17] KROHN, K. B., and J. L. RHODES: Algebraic Theorie of Machines, in: Proceedings of the Symposium on Mathematical Theory of Automata, 1962. Polytechnic Press of the Polytechnic Institute of Brooklyn.
- [18] MEALY, G. H.: Method for Synthesizing Sequential Circuits. Bell System Techn. J. 34, (1955).
- [19] MITCHELL, B.: Theory of Categories. New York: Academic Press. 1965.
- [20] MOORE, E. F.: Gedankenexperiments on Sequential Machines, in G. E. SHANNON and J. McCARTHY: Automata Studies. Princeton University Press. 1956.
- [21] MYHILL: Linear Bounded Automata. WADD Techn. Note, No. 60-165. Wright-Patterson Air Force Base. Ohio. 1960.
- [22] RABIN, M. O., and D. SCOTT: Finite Automata and Their Decision Problems. IBM Journal 2, 114—125 (1959).
- [23] SCHNORR, C. P.: Homomorphismen kontextfreier Sprachen. Erscheint demnächst.
- [24] SCHNORR, C. P.: Vier Entscheidbarkeitsprobleme für kontextsensitive Sprachen. Erscheint demnächst in der EIK.
- [25] SCHNORR, C. P., und H. WALTER: Pullbackkonstruktionen bei Semi-Thue-systemen. Erscheint demnächst in der EIK.
- [26] WALTER, H.: Pullbackkonstruktionen bei Semi-Thuesystemen. 4. Kolloquium über Automatentheorie 4, (Ed. BAUER und SAMELSON, München).
- [27] WALTER, H.: Erweiterungen von endlichen Automaten. Erscheint demnächst in der EIK.
- [28] YOELI, M.: The Cascade Decomposition of Sequential Machines. IRE Trans. EC-10, (1961).

Prof. Dr. Günter Hotz
Institut für Angewandte Mathematik und Rechenzentrum
Universität des Saarlandes
D-66 Saarbrücken
Bundesrepublik Deutschland