KYBERNETIKA ČÍSLO 1, ROČNÍK 1/1965

Zur syntaktischen Synthese*

JÜRGEN KUNZE

Neben einigen Betrachtungen über Abhängigkeitsbäume und Bemerkungen über den allgemeinen Ablauf einer Automatischen Übersetzung enthält dieser Artikel eine Übersicht über Unterordnungscharakteristiken für das Deutsche sowie einen Algorithmus für die Herstellung der deutschen Wortreiehenfolge, der die Unterordnungscharakteristiken benutzt.

In diesem Vortrag wird eine Methode für die syntaktische Synthese in der automatischen Übersetzung dargestellt, die auf einigen einfachen Prinzipien aufbaut und einen liecht überblickbaren formalen Ablauf besitzt. Diese Methode ist eine Weiteentwicklung derjenigen, die in einem Übersetzungsexperiment Englisch-Deutsch im Jahre 1963 an der Arbeitsstelle für Mathematische und Angewandte Linguistik und Automatische Übersetzung der Deutschen Akademie der Wissenschaften verwendet wurde. Über die Unterschiede wird später zu sprechen sein.

Alle Erläuterungen und Beispiele werden am Deutschen gegeben. Das allgemeine Verfahren ist jedoch auch für andere Sprachen anwendbar, speziell für Englisch, Französisch, Russisch usw.

Die wesentliche Voraussetzung für die hier geschilderte Methode sind die Unterordnungscharakteristiken (im folgenden mit UC abgekürzt). Sie müssen bereits vorhanden sein, damit der zu schildernde Vorgang ablaufen kann.

Befassen wir uns daher zunächst mit diesen UC. Sie können als Verallgemeinerung und formale Auffassung des Begriffs *syntaktisches Verhältnis* verstanden werden. Das bedeutet, daß sich z.B. Beziehungen wie *Subjekt, Objekt, Attribut, adverbiale Bestimmung, Nebensatz* usw. als UC wiederfinden werden. Die Aufstellung eines vollständigen Systems von UC, mit denen man alle syntaktischen Beziehungen beschreiben kann, hängt natürlich vom Ziel der Untersuchungen ab. So gibt Meltschuk

* Vortrag am *Prager Koloquium über algebraische Linguistik und Maschienenübersetzung*, 18.—22. September 1964.

86 in seiner Arbeit [1] ein System von grammatischen Beziehungen an. Jedoch ist seine Klassifizierung anders als die, die wir benötigen werden.

Bevor wir nun die Einzelheiten erläutern, müssen wir erst einmal die zugrunde gelegte Auffassung der Grammatik beschreiben: Sätze (sowohl einfache als auch zusammengesetzte) werden als *Abhängigkeitsbäume* dargestellt, und nicht als *Ableitungsbäume*. Das bedeutet, daß die Knoten des Baumes Wörter des Satzes sind und keine Elemente des sogenannten Hilfsvokabulars (wie etwa *NP*, *VP*, *Adv* usw. nach Chomsky). An der Spitze des Baumes steht das abschließende Satzzeichen, also ein Punkt (oder Semikolon), ein Fragezeichen oder ein Ausrufungszeichen. (Damit wird in gewisser Weise die Art des Satzes schon bestimmt: Aussagesatz bzw. Fragesatz bzw. Befehls- oder Ausrufesatz.) Dieses Satzzeichen wird als Wort des Satzes aufgefaßt. Die inneren Satzzeichen, also Kommata usw., werden nicht als Elemente des Baumes aufgenommen! Alle übrigen Wörter sind in folgender Weise dargestellt: Zu jedem vom Schlußsatzzeichen verschiedenen Wort W_1 des Satzes S existiert genau ein Wort $W_2 \in S$, so daß W_1 dem Wort W_2 untergeordnet ist. Wir schreiben dafür:

$$W_2 = M(W_1)\,,$$

W_2 ist das W_1 direkt übergeordnete Wort.

Die Bedingung, daß jedes Wort nur einem anderen untergeordnet wird, bringt natürlich gewisse Einschränkungen mit sich, da einige Wörter real direkte grammatische Beziehungen zu mehreren Wörtern haben, z.B. die Relativpronomina. Jedoch geben die UC diese Beziehungen wieder, wenn das Wort auch nur einem anderen untergeordnet ist.

Die Funktion $M(W)$ wird zur Funktion $M_h(W)$ verallgemeinert durch folgende induktive Definition:

$$M_0(W) = {}_{\text{Def}} W\,,$$

$$M_{h+1}(W) = {}_{\text{Def}} \begin{cases} M(M_h(W)), \text{ falls } M_h(W) \neq Sp, Sf, Sa, \\ \text{nicht definiert, falls } M_h(W) = Sp, Sf \text{ oder } Sa. \end{cases}$$

Dabei bezeichnen Sp, Sf und Sa in dieser Reihenfolge den Punkt, das Fragezeichen und das Ausrufungszeichen. Als generelle Abkürzung verwenden wir für diese drei Sz. Es ist z. B. $M_2(W)$ das Wort, das $M(W)$ übergeordnet ist. Die Funktionen M und M_1 stimmen überein, was man sofort aus der Definition entnimmt, wenn man $h = 0$ setzt. $M_1(Sp) = M(Sp)$, $M(Sf)$ und $M(Sa)$ sind nicht definiert.

Die Forderung, daß der Baum keine geschlossenen Wege enthalten soll, läßt sich so ausdrücken:

$$\text{Für jedes } W \in S \text{ gilt: Wenn } M_h(W) = W, \text{ so ist } h = 0\,.$$

Alle Wörter $W \in S$, die von Sz verschieden sind, sollen Sz direkt oder indirekt untergeordnet sein, d.h., zu jedem $W \in S$ gibt es ein $h \geqq 0$ mit

$$M_h(W) = Sz\,,$$

$h = 0$ bei $W = Sz$,
$h = 1$ bei denjenigen W, die direkt von Sz abhängen,
$h > 1$ bei denjenigen W, die indirekt von Sz abhängen.

Aus der Bedingung, daß der Baum keine geschlossenen Wege enthält, folgt sofort, daß h durch W eindeutig bestimmt ist. Daher liegt eine Funktion $h(W)$ vor, die wir die Hierarchie des Wortes W nennen. Es gilt

$$h(W) \geqq 0\,, \quad M_{h(W)}(W) \equiv Sz\,.$$

$M_\kappa(W)$ existiert für $0 \leqq x \leqq h(W)$, $M_\kappa(W)$ existiert nicht für $x > h(W)$.

Durch diese Forderungen ist die Struktur eines Baumes $B(S)$, der einen Satz S wiedergibt, im wesentlichen festgelegt:

$B(S)$ enthält als Elemente die eigentlichen Wörter von S und das Satzzeichen Sz am Schluß des Satzes.

Zu jedem $W \in B(S)$ mit $W \neq Sz$ existiert ein $M(W) \in B(S)$. $M(W)$ ist das Wort, dem W untergeordnet ist.

$B(S)$ enthält keine geschlossenen Wege, d.h., für jedes W gilt: Wenn $M_h(W) = W$, so $h = 0$.

Zu jedem $W \in B(S)$ existiert genau eine Zahl $h(W) \geqq 0$, die Hierarchie von W, die durch $M_{h(W)}(W) = Sz$ definiert ist. Die Funktion $h(W)$ ist durch die Funktion $M(W)$ implizit gegeben. Offenbar gilt:

$$h(M(W)) = h(W) - 1\,.$$

Nun reicht die reine Struktur eines Satzes, so wie sie eben geschildert wurde, sicher nicht aus, um alle relevanten grammatischen Beziehungen ablesen zu können, d. h., sie sind noch nicht in der reinen Struktur explizit enthalten. Es ist vielmehr notwendig, auch die Wortklassen der Wörter zu kennen. Wir vervollständigen daher unser Modell durch folgenden Zusatz: Zu jedem $W \in B(S)$ ist eine Wortklasse $K(W) \in \mathfrak{K}$ gegeben. $\mathfrak{K}$ ist dabei die Menge aller Wortklassen und stellt eine nicht notwendig disjunkte Zerlegung der Menge aller Wörter der betrachteten Sprache in Untermengen dar. $K(W)$ sei dabei die aktuelle Wortklasse, wenn W, isoliert für sich betrachtet, mehreren Wortklassen angehören kann. Die morphologischen Merkmale des Wortes W seien in $K(W)$ mit enthalten. Das Problem, wie man bei Wörten, deren Wortklasse nicht eindeutig bestimmt ist, die aktuelle Wortklasse findet, umgehen wir hier, ebenso die Frage, wie das System $\mathfrak{K}$ der Wortklassen aufzustellen ist. Dies sind vor allem Fragen der Analyse.

Man überzeugt sich, daß auch diese Informationen noch nicht ausreichend sind; denn es ist z. B. in gewissen Fällen nicht möglich, einen adverbialen Akkusativ von einem Akkusativobjekt zu unterscheiden, ebenso verhält es sich mit Subjekt und Prädikatsnomen. Dieser Mangel wird dadurch behoben, daß jedes Wort außerdem noch eine Information darüber erhält, welche Funktion es zu dem ihm übergeordneten Wort einnimmt, d. h., in welchem syntaktischen Verhältnis es zu ihm steht. Diese

Information ist gerade die UC des Wortes, abgekürzt wiedergegeben durch $C(W)$. Mit diesen UC wollen wir uns nun ausführlicher befassen. Wir geben zunächst die wichtigsten UC für das Deutsche an und erklären sie. Die aufgeführten UC stellen jedoch nur einen Ausschnitt dar, es sind für das Deutsche, vor allem für die syntaktische Synthese, bedeutend mehr davon nötig.

aa Charakteristik des Ausrufungszeichens *Sa* an der Spitze des Baumes. Der gesamte abhängige Satz wird dadurch als Befehls- oder Ausrufesatz gekennzeichnet.

ff Charakteristik des Fragezeichens *Sf* an der Spitze. Der Satz ist ein Fragesatz.

pp Charakteristik des Punktes *Sp* an der Spitze. Der Satz ist ein Aussagesatz.

AA Adverb als Attribut zum Adjektiv oder Adverb.
(die sehr schöne Oper; sehr oft fahren)

AB Präpositionalgruppe als adverbiale Bestimmung unter Verb. (Vgl. *FO*)
(fahren mit dem Auto)

AD Adverb als adverbiale Bestimmung unter Verb, außer adverbialer Verneinung. (Vgl. *AN.*)
(sehr oft fahren)

AF Präpositionalgruppe als Attribut zum Adjektiv.
(unabhängig von der Methode)

AN Adverbiale Verneinung.
(nicht fahren)

DO Substantiv als direktes Objekt unter transitivem Verb. (Vgl. *IO*, *DR.*)
(schenken dem Freund das Buch)

DR Relativpronomen oder Substantiv, dem ein Relativpronomen untergeordnet ist, als direktes Objekt unter transitivem Verb.
(das Buch, das ich dem Freund schenke; der Komponist, dessen Oper die Besucher hören)

DS Vollständiger Konjunktionalsatz als Objektsatz.(Vgl. *NS* und *PS.*)
(wir sehen, daß der Freund kommt)

FE Substantiv unter Präposition.
(von der Methode)

FO Präpositionalgruppe als präpositionales Objekt unter Verb.
(abhängen von der Methode)

IO Substantiv als indirektes Objekt.
(schenken dem Freund das Buch; danken dem Freund)

IR Relativpronomen oder Substantiv, dem ein Relativpronomen untergeordnet ist, als indirektes Objekt unter Verb.
(der Freund, dem ich das schenke)

KS Subordinierende Konjunktion des Konjunktionalsatzes unter dem Verb dieses Satzes.
(..., daß der Freund kommt)

NA Adjektiv als Attribut zum Substantiv.
(das neue Buch)

NC Possessivisches Relativpronomen als Attribut zum Substantiv.
(der Komponist, dessen Oper die Besucher hören)

ND Artikel als Attribut zum Substantiv.
(das neue Buch; ein neues Buch)

NF Präpositionalgruppe als präpositionales Attribut zum Substantiv.
(das Buch im Schrank)

NG Substantiv als Genitivattribut zum Substantiv.
(die Oper des Komponisten)

NR Relativsatz als Attributsatz zum Substantiv, Verb des Relativsatzes unter dem Beziehungswort.
(der Komponist, dessen Oper die Besucher hören)

NS Vollständiger Konjunktionalsatz als Attributsatz zum Substantiv, Verb des Konjunktionalsatzes unter dem Beziehungswort.
(der Wunsch, daß der Freund kommt)

PA Adjektiv als Prädikatsnomen unter Kopula.
(die Lösung ist unabhängig von ...)

PN Substantiv als Prädikatsnomen unter der Kopula.
(der Freund ist Lehrer)

PR Abtrennbares Präfix unter dem Stammverb.
(die Lösung hängt von der Methode ab)

PS Vollständiger Konjunktionalsatz als adverbialer Nebensatz, Verb des Konjunktionalsatzes unter Verb des übergeordneten Satzes.
(wir schreiben, wenn der Freund kommt)

SN Substantiv als Subjekt unter Verb.
(die Lösung ist unabhängig von ...; der Wunsch, daß der Freund kommt; der Komponist, dessen Oper die Besucher hören)

SR Relativpronomen oder Substantiv, dem ein Relativpronomen untergeordnet ist, als Subjekt unter Verb.
(der Freund, der kommen wird; der Komponist, dessen Oper uns gefällt)

90 *VF* Finites temporales Hilfsverb unter Vollverb.
(der Freund ist gekommen; der Freund hat gesagt; der Freund, der kommen wird)

VV Vollverb des Hauptsatzes unter dem Satzzeichen.
(der Freund kommt •)

Zu dem UC *AB* und *AD* ist zu bemerken, daß sie unbedingt noch weiter differenziert werden müssen, nämlich in kausale, temporale, lokale, Richtungsbestimmungen usw. Dies ist vor allem notwendig, weil zwischen den verschiedenen adverbialen Bestimmungen auch gewisse Einschränkungen für ihre Reihenfolge im Satz bestehen. Die Abgrenzung zwischen *AB* und *FO* ist mitunter schwierig. In dem später dargestellten Algorithmus wird sich jedoch zeigen, daß aus Gründen der Wortstellung ein Unterschied zwischen *AB* und *FO* gemacht werden muß.

Aus der Beschreibung der UC ergibt sich, daß man eine solche UC, ganz grob, als eine mit einer grammatischen Funktion versehene Wortklasse ansehen kann.

UC eines Wortes im Satz = Wortklasse des Wortes im Satz
+ Funktion des Wortes im Satz.

Zu den Relativpronomina ist noch folgendes zu bemerken: In dem Satz *Der Freund, dem ich das Buch schenke, ...* hat das Relativpronomen *dem* grammatische Beziehung zum Verb *schenken* (durch den Dativ ausgedrückt) und zum Substantiv *Freund* (durch Genus = mask. und Numerus = singl. ausgedrückt). Die Beziehung zum Verb besteht unmittelbar (vgl. *IR*), die Beziehung zum Substantiv ergibt sich auf folgende Weise nach einer generellen Regel:

Jedes Relativpronomen, gleich, in welcher grammatischen Beziehung es im Relativsatz steht, ist direkt oder indirekt der Spitze dieses Relativsatzes untergeordnet, die die UC *NR* hat. Das Wort, dem dieses Wort *W* mit $C(W) = NR$ untergeordnet ist, ist das Beziehungswort (vgl. *NR*).

Der eigentliche Grund für die Differenzierung der UC *SR, IR, DR* und *NC* gegenüber *SN, IO, DO* und *NA* (bzw. *ND*) besteht jedoch darin, daß die Satzglieder, die das Relativpronomen in irgendeiner Form enthalten, immer am Beginn des Relativsatzes stehen, bei der Herstellung der Wortreihenfolge also nicht wie gewöhnliche Satzglieder behandelt werden dürfen.

Dies mag als Überblick für die Darstellungsweise genügen, die wir für die Sätze verwenden. Wir wollen nun kurz die Frage streifen, wie man zu einem vorgegebenen Satz den entsprechenden Baum durch einen Algorithmus erhält. Syntaktische Mehrdeutigkeiten interessieren hier nicht. Betrachten wir eine bilaterale Übersetzung. Ein Satz S_Q der Quellensprache sei vorgegeben. Die Behandlung der syntaktischen Probleme (nur diese behandeln wir hier) zerfällt in zwei große Abschnitte: ***Syntaktische Analyse*** und ***Syntaktische Synthese.***

Die syntaktische Analyse ordnet S_Q den entsprechenden Baum der Quellensprache zu:

$$S_Q \rightarrow B_Q(S_Q).$$

Die syntaktische Synthese zerfällt in zwei Unterabschnitte: *Die Konstruktionsumwandlungen* und *die Herstellung der Wortreihenfolge in der Zielsprache.* Die Konstruktionsumwandlungen verwandeln $B_Q(S_Q)$ in $B_Z(S_Z)$, die Herstellung der Wortreihenfolge ordnet den Baum $B_Z(S_Z)$ nach den Regeln der Zielsprache um, so daß man S_Z erhält, den entsprechenden Satz in der Zielsprache. $B_Z(S_Z)$ ist der Baum des „übersetzen" Satzes S_Z in der Zielsprache.

Durch eine Vorstufe der syntaktischen Analyse oder während der syntaktischen Analyse selbst werden die aus dem Analyse-Wörterbuch stammenden Informationen über die möglichen Wortklassen eindeutig gemacht, d. h., die aktuellen Wortklassen werden festgelegt. Dann kennen wir für jedes Wort W die Wortklasse $K(W)$. Es fehlen also noch die Informationen $M(W)$ und $C(W)$, d. h., es muß bestimmt werden, welchem Wort jedes Wort mit welcher UC untergeordnet ist. Danach ist $B_Q(S_Q)$ bestimmt.

Durch die natürliche Reihenfolge der Wörter in S_Q ist eine Abzählung der Wörter dieses Satzes vorgegeben. Zu jedem $W \in S_Q$ existiert also eine Zahl $n(W)$. Für das erste Wort W ist $n(W) = 01$, für das zweite Wort W $n(W) = 02$ usw. Wir können $n(W)$ als weiteres Bestimmungsstück zu unseren Bäumen mit hinzunehmen. Wir haben dann einen geordneten Baum.

Befassen wir uns daher noch etwas mit geordneten Bäumen. Eine Teilmenge $E \leqq B$ eines Baumes B heißt ein Abschnitt, wenn folgendes gilt: Wenn $W_1 \in E$, $W_2 \in E$ und $W_3 \in B$ ein Wort mit $n(W_1) \leqq n(W_3) \leqq n(W_2)$ ist, so gilt $W_3 \in E$. (Wir nehmen *OBdA* $n(W_1) \leqq n(W_2)$ an.) Abschnitte sind also Teilmengen von B, die aus aufeinander folgenden Wörtern bestehen, im Sinne der Ordnung $n(W)$. Ein einzelnes Wort ist offenbar immer ein Abschnitt.

Unter $A(W)$ verstehen wir die Menge aller Wörter, die direkt oder indirekt von W abhängen, einschließlich W selbst:

$$W_1 \in A(W) =_{\text{Def}} \text{Es gibt ein } h \geqq 0 \text{ mit } M_h(W_1) = W.$$

Wir sagen, daß ein Baum einfach projektiv geordnet ist, wenn folgendes gilt:

Für jedes $W \in B$ ist $A(W)$ ein Abschnitt.

Anschaulich bedeutet dies folgendes: Ordnet man die Wörter W des Baumes von oben nach unten nach wachsenden $h(W)$ und von rechts nach links nach wachsenden $n(W)$, verbindet man jedes Wort W mit seinem übergeordneten Wort $M(W)$ durch eine Strecke und zieht man von jedem W aus eine senkrechte Linie nach unten bis zu einer gewissen Grundlinie, so schneiden sich alle diese Linien nicht (Abb. 1).

92 Betrachtet man eine bestimmte Sprache, so kann man sich die Frage vorlegen, ob alle Sätze dieser Sprache eine Darstellung durch einen einfach projektiv geordneten Baum besitzen. Nun ist diese Frage nicht a priori zu beantworten, sondern man muß zuerst eine Art der Darstellung durch Bäume festlegen, wie wir dies im Auszuge getan haben. Erst dann kann auf die gestellte Frage eine Antwort gegeben werden.

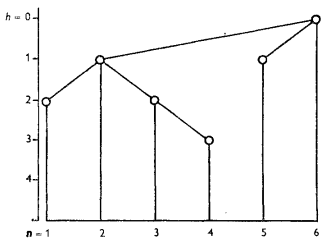

Abb. 1.

Es ist jedoch von Nutzen, die Grundkonzeption nach Möglichkeit so zugestalten, daß die Projektivität so weit wie möglich gesichert wird. Es zeigt sich dabei, daß bei gewissen Entscheidungen, wie man unterzuordnen hat, die Lösung, die der Projektivität entspricht, auch die linguistisch bessere ist. Ich verweise heir auf Untersuchungen von Fitialov in Leningrad.

Es ist aber keineswegs so, daß man die Projektivität hundertprozentig erreichen kann. So gibt es im Deutschen (wie auch im Englischen) Wortstellungen, die nicht projektiv sind. Als Beispiel werden wir die verzögerten Attributsätze behandeln. Hier müßte man die wahren grammatischen Beziehungen verfälschen, wenn man Projektivität erzwingen wollte. Der Begriff *einfach projektiv geordnet* muß dann entsprechend verallgemeinert werden. Wir gehen hier jedoch nicht darauf ein.

Es gilt folgender Satz, der hier ohne Beweis mitgeteilt sei: Ein Baum B ist genau dann einfach projektiv geordnet, wenn für drei beliebige und paarweise verschiedene Wörter W_1, W_2 und W_3 aus B folgendes gilt:

Wenn

$$W_1 = M(W_2) \quad \text{und} \quad n(W_1) < n(W_2) \quad \text{und} \quad n(W_1) < n(W_3) < n(W_2),$$

so ist

$$n(W_1) \leqq n(M(W_3)) \leqq n(W_2) \quad \text{und} \quad h(W_3) \leqq h(W_2).$$

Wenn

$$W_1 = M(W_2) \quad \text{und} \quad n(W_1) > n(W_2) \quad \text{und} \quad n(W_2) < n(W_3) < n(W_1),$$

so ist

$$n(W_2) \leqq n(M(W_3)) \leqq n(W_1) \quad \text{und} \quad h(W_3) \leqq h(W_2).$$

Dieser Satz besagt, anders ausgedrückt: Zwischen zwei Wörtern W_1 und W_2, von denen eines dem anderen untergeordnet, ist, können nur Wörter W stehen, deren übergeordnetes Wort $M(W)$ ebenfalls zwischen W_1 und W_2 (diese beiden mit eingeschlosen) steht und deren Hierarchie $h(W)$ das Minimum von $h(W_1)$ und $h(W_2)$ nicht übertrifft.

Für die syntaktische Analyse läßt sich dieser Sachverhalt in folgender Weise ausnutzen: Man kann sämtliche UC daraufhin untersuchen, ob sie projektiv in folgendem Sinne sind:

Eine UC XY heißt *projektiv*, wenn für jeden Baum $B(S)$ der betrachteten Sprache folgendes gilt: Sind W_1, W_2 und W_3 beliebige Wörter aus $B(S)$ und ist außerdem

$$W_1 = M(W_2),\ C(W_2) = XY,\ n(W_1) < n(W_2)$$

und

$$n(W_1) < n(W_3) < n(W_2)\,,$$

so ist

$$n(W_1) \leqq n(M(W_3)) \leqq n(W_2)$$

und

$$h(W_3) \leqq h(W_2)\,.$$

Entsprechendes gilt wieder, wenn $n(W_1) > n(W_2)$ ist.

Sind alle Bäume einfach projektiv geordnet, so sind auch alle UC projektiv und umgekehrt. Unter den oben aufgeführten UC für das Deutsche sind z.B. *NR* und *NS* nicht projektiv in diesem Sinne. Jedoch sind die weitaus meisten UC projektiv.

Für die projektiven UC lassen sich die Unterordnungsregeln in der syntaktischen Analyse in einer einfachen Weise aufstellen und anordnen. Die Nachteile eines solchen Verfahrens bestehen einmal darin, daß der Reihenfolge der Regeln eine große Bedeutung zukommt, womit eine gewisse Fehleranfälligkeit beim Aufstellen des Algorithmus verbunden ist, und zum zweiten darin, daß viele Regeln bei der Abarbeitung mehrmals wiederholt werden müssen. (Einige Erörterungen dazu finden sich im Bericht unserer Arbeitsstelle aus dem Jahre 1963.) Die nicht-projektiven UC müssen auf andere Weise behandelt werden.

Soviel zur syntaktischen Analyse. Die Konstruktionsumwandlungen verwandeln den Baum $B_Q(S_Q)$ in $B_Z(S_Z)$. Sie verwandeln, grob gesprochen, diejenigen grammatischen Konstruktionen der Quellensprache, die nicht unmittelbar in die Zielsprache übernommen werden können, in entsprechende Konstruktionen der Zielsprache. Insbesondere können Wörter (Funktionswörter, Pronomina usw.) eingefügt oder gestrichen werden. Auch Änderungen von Wortklassen können vorkommen.

Nach den Konstruktionsumwandlungen haben wir den Baum $B_Z(S_Z)$. Er enthält für jedes Wort $W \in B_Z(S_Z)$ folgende Informationen:

Die Wortklasse	$K(W)$,
das übergeordnete Wort	$M(W)$ (für $W \neq Sz$),
die Unterordnungscharakteristik	$C(W)$ und
die Nummer in S_Q	$n(W)$.

(In den Konstruktionsumwandlungen eingefügte Wörter stehen am Schluß.)

Wir beziehen nun alle diese Informationen auf die Abzählung $n(W)$, die Funktion $n(W)$ wird zur Funktion $W(n)$ umgekehrt:

$$W(n) = n\text{-tes Wort des Baumes}.$$

Zuvor rücken wir den Satz zusammen, so daß die durch Streichungen entstandenen Leerstellen verschwinden. N sei die Anzahl der Elemente von $B = B_Z(S_Z)$. Die Informationen erhalten dann für $1 \leqq n \leqq N$ folgendes Aussehen:

$K(n)$ = Wortklasse des n-ten Wortes,
$M(n)$ = das dem n-ten Wort übergeordnete Wort,
$C(n)$ = UC des n-ten Wortes.

Ferner erhalten wir aus $M_{h(n)}(n) = Sz$ die Funktion $h(n)$ für $1 \leqq n \leqq N$. Es sei

$$I(j) = {}_{\text{Def}}\, n(M(j)) \quad \text{für} \quad 1 \leqq j \leqq N.$$

$I(j)$ ist die Nummer des übergeordneten Wortes, die der Unterordnungsindex (abgekürzt UI) des j-ten Wortes genannt wird. Wegen $M(j) = W(n(M(j))) = W(I(j))$ wird die Funktion $M(j)$ dann entbehrlich, so daß wir $M(j)$ durch $I(j)$ ersetzen können. Für $W = Sz$ und $n(Sz) = n_0$ sei $I(n_0) = I(n(Sz)) = 00$, die Spitze hat also den UI 00.

Da $B_Z(S_Z)$ bereits die grammatischen Beziehungen der Zielsprache wiedergibt, bleibt nur noch die Reihenfolge der Wörter, wie sie der Zielsprache entspricht, herzustellen; denn n ist die Reihenfolge in der Quellensprache, die durch die Konstruktionsumwandlungen sowieso ihren Sinn verloren hat. Mit anderen Worten:

Zu $K(n)$, $I(n)$ und $C(n)$ ist eine weitere Funktion $Z(n)$ zu bilden, die folgende Bedeutung hat:

$$Z(n) = \text{Stelle des } n\text{-ten Wortes von } B_Z(S_Z) \text{ in } S_Z.$$

Ein Beispiel erläutere dies kurz:

	I	*have*	*seen*	*him*	*yesterday*	.
n	01	02	03	04	05	06
$Z(n)$	01	02	05	03	04	06

da es im Deutschen heißt:

	Ich	*habe*	*ihn*	*gestern*	*gesehen*	.
$Z(n)$	01	02	03	04	05	06

Zum Algorithmus für die Herstellung der Wortreihenfolge gehören: Ein Flußdiagramm, ein System von Ordnungsreihen, ein System von Bedingungen, ein spezieller Speicher (push-down store), einige Arbeitszellen.

Wir erklären zunächst die sechs Ordnungsreihen:

$q = 01$:	01	02
	pp	*ZZ*
	02	
	0	

$q = 02$:	01	02	03
	VV	*MM*	*ZZ*
	03	00	
	0	0	

$q = 03$:	01	02	03	04	05	06	07	08	09	10	11	12	13	14
	KS	*SR*	*IR*	*DR*	*PS*	*AB*	*AD*	*SN*	*VF*	*MM*	*SN*	*AB*	*AD*	*IO*
	00	04	04	04	03	05	06	04	00	00	04	05	06	04
	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	16	17	18	19	20	21	22	23	24	25	26	27
	DO	*AN*	*PN*	*PA*	*FO*	*PR*	*MM*	*VF*	*NR*	*NS*	*DS*	*PS*	*ZZ*
	04	00	04	06	05	00	00	00	03	03	03	03	
	0	0	0	0	0	0	0	0	1	1	0	0	

$q = 04$:	01	02	03	04	05	06	07	08	09
	NC	*ND*	*NA*	*MM*	*NG*	*NF*	*NR*	*NS*	*ZZ*
	00	00	06	00	04	05	03	03	
	0	0	0	0	0	0	0	0	

$q = 05$:	01	02	03
	MM	*FE*	*ZZ*
	00	04	
	0	0	

$q = 06$:	01	02	03	04	05
	AF	*AA*	*MM*	*AF*	*ZZ*
	05	06	00	05	
	0	0	0	0	

Die Ordnungsreihen (abgekürzt durch OR) werden mit dem Zähler q abgezählt. ($q = 01, \ldots, 06$) Innerhalb einer OR läuft der Zähler p (erste Zeile). Die zweite Zeile gibt die entsprechende UC an. Sie ist durch p und q bestimmt, und wir schreiben

$$C_q(p) = p\text{-te UC der } q\text{-ten Ordnungsreihe.}$$

So ist z. B.

$$C_{03}(11) = SN\,, \quad C_{04}(05) = NG\,.$$

Die dritte Zeile gibt die Nummer der OR an, die als von dem entsprechenden Wort abhängige zu bearbeiten ist. Wir schreiben dafür:

$$Q_q(p)\,.$$

Beispielsweise ist:

$$Q_{03}(11) = 04\,, \quad Q_{04}(05) = 04\,.$$

Die vierte Zeile gibt an, ob das bearbeitete Wort direkt oder indirekt von demjenigen Wort abhängen soll, das als übergeordnetes Wort festgelegt wurde: 0 bei direkter

Abhängigkeit, 1 bei indirekter Abhängigkeit. Der zweite Fall deutet auf nicht-projektive Wortstellung hin. Wir schreiben

$$T_q(p) = 0 \quad \text{oder} \quad 1 .$$

Damit hat jede OR die Gestalt:

01	02	...	p	...
$C_q(01)$	$C_q(02)$		$C_q(p)$	
$Q_q(01)$	$Q_q(02)$		$Q_q(p)$	
$T_q(01)$	$T_q(02)$		$T_q(p)$	

Die OR bedeuten inhaltlich folgendes: (Man vergleiche dazu die Liste der UC!) *MM* steht für das Wort, das als übergeordnetes aller gerade zu bearbeitenden Wörter festgelegt wurde, *ZZ* markiert das Ende der OR. Die erste OR ($q = 01$) besagt, daß im Baum ein Wort mit der UC *pp* zu suchen ist. (Wir behandeln die Frage- und Befehlssätze hier nicht!) Dies ist das abschließende Satzzeichen. Hat man es gefunden, so gehe man zur Reihe $q = 02$ über. ($Q_{01}(01) = 02$) Das eben gefundene Wort wird nun als dasjenige Wort festgelegt, von dem alle anderen abhängen sollen, die als nächste bearbeitet werden. Wir suchen also jetzt ein Wort mit der UC *VV*, das direkt von diesem Wort abhängt. (Es ist $T_{02}(01) = 0$, daher direkte Abhängigkeit!) Ist dieses Wort mit allen davon abhängenden bearbeitet, so wird als letztes der Punkt gesetzt; denn *MM* steht für dieses Wort. – Haben wir *VV* als UC bei einem Wort im Satz gefunden und hängt dieses Wort direkt vom Punkt ab, so gehen wir zur Reihe $q = 03$ über, da $Q_{02}(01) = 03$ ist. Das Wort mit der UC *VV* wird jetzt das Wort, von dem alle diejenigen abhängen sollen, die nach dieser Reihe gesucht werden. Diese Reihe ist die Satz-Reihe, sie enthält die Satzglieder. Finden wir etwa ein Wort mit der UC *SN*, das direkt von dem Wort mit der UC *VV* abhängt, so gehen wir zur Reihe $q = 04$ über. ($Q_{03}(08) = 04$) Die Reihe $q = 04$ stellt die Nominalphrase dar. Ist sie vollständig bearbeitet, so gehen wir in der Reihe $q = 03$ weiter und suchen *VF*, dann das Wort mit der UC *SN* noch einmal, nachdem wir das Wort mit der UC *VV* bearbeitet haben, für das *MM* steht, usw. – Der genaue Ablauf ergibt sich aus dem Flußdiagramm (Abb. 2).

Bei denjenigen Werten von p, die unterstrichen sind (z. B. in $q = 03 : p = 05, \ldots, \ldots, 10$), müssen außerdem gewisse Bedingungen geprüft werden, die später geschildert werden.

Wie diese Such- und Entscheidungsprozesse mit der Herstellung der Wortreihenfolge, d.h. mit der Aufstellung der Funktion $Z(n)$ zusammenhängen, werden wir später sehen: Ein gefundenes Wort wird dann mit einer Nummer $Z(n)$ versehen, wenn an der entsprechenden Stelle in der OR $Q_q(p) = 00$ ist.

Für das Flußdiagramm sind zusätzlich noch folgende Symbole zu erklären:

z ist ein Zähler, der angibt, wieviele Wörter bereits mit der Information $Z(n)$ versehen wurden.

m ist die Nummer n desjenigen Wortes, von dem die in der gerade bearbeiteten OR gesuchten Wörter abhängen sollen. Dieses Wort W mit $n(W) = m$ ist in der OR durch MM wiedergegeben.

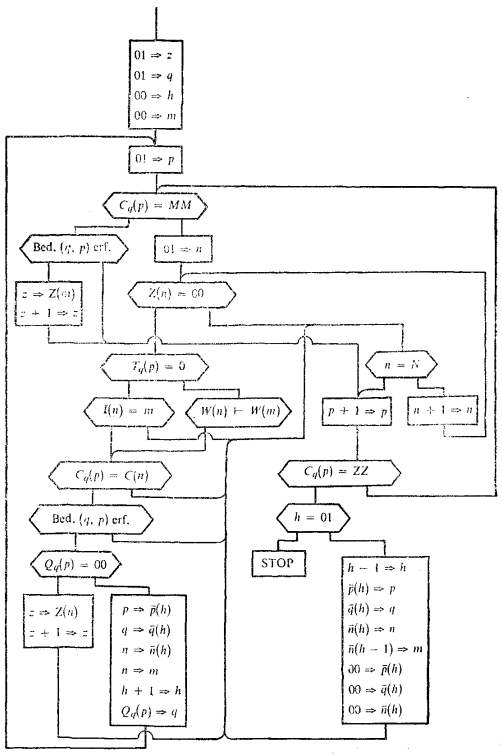

Abb. 2.

h stimmt, wenn stets $T_q(p) = 0$ ist, mit der Hierarchie $h(W)$ des gerade bearbeiteten Wortes überein. Sonst ist $h(W) \geqq h$.

$W(n) \vdash W(m)$ bedeutet: Das Wort mit der Nummer n hängt vom Wort mit der Nummer m ab, ohne daß auf dem entsprechenden Weg im Baum ein Wort mit einer Nebensatzcharakteristik (abgekürzt durch NSC) steht. In unserem Ausschnitt sind die folgenden NSC vorhanden: *DS*, *NR*, *NS* und *PS*. Wörter, die eine

dieser UC haben, sind die Spitze eines Nebensatzes im Baum. $W(n) \vdash W(m)$ bedeutet also: Es gibt ein $i > 0$ mit $W(m) = M_i(W(n))$, und für alle j mit $1 \leqq j < i$ ist $C(M_j(W(n))) \neq DS, NR, NS, PS$.

$\Rightarrow$ bedeutet, daß die rechte Seite den Wert bekommt, der links steht.

Viereckige Kästchen stellen Operationen dar, sechseckige enthalten logische Bedingungen, bei deren Erfülltsein der linke und bei deren Nicht-Erfülltsein der rechte Ausgang zu wählen ist. Der Speicher enthält für jedes $h \geqq 0$ drei Zellen $\bar{p}(h)$, $\bar{q}(h)$ und $\bar{n}(h)$, in die gewisse Werte von p, q und n abgespeichert werden.

Die im Flußdiagramm genannten Bedingungen (q, p) wollen wir hier nicht weiter schildern. Immer erfüllt ist Bed. (q, p) für alle Paare (q, p) außer (03, 05), (03, 06), (03, 07), (03, 08), (03, 09), (03, 10), (04, 07), (04, 08) und (06, 01). Bed. (06, 01) z. B. verhindert die ungrammatische Wortstellung *die abhängige von der Methode Lösung* (statt *die von der Methode abhängige Lösung*) und erlaubt aber die beiden richtigen Wortstellungen *die Lösung ist abhängig von der Methode* und *die Lösung ist von der Methode abhängig*. Sämtliche Bedingungen sind mit UC und den anderen Beziehungen im Baum formuliert, benutzen also keine äußeren Entscheidungen. Die strenge Formulierung von Bed. (06, 01) lautet z.B.:

$$\text{Ist } C(I(n)) = NA, \text{ so ist Bed. (06, 01) erfüllt.}$$
$$\text{Ist } C(I(n)) \neq NA, \text{ so ist Bed. (06, 01) frei,}$$

d. h., man kann sich aussuchen, ob man die Bedingung als erfüllt ansieht oder nicht. (Hier lassen sich stilistische Gesichtspunkte berücksichtigen.) Für $C(I(n)) = PA$ erhalten wir so die beiden oben genannten richtigen Wortstellungen, für $C(I(n)) = NA$ muß AF vor dem Adjektiv stehen, die ungrammatische Stellung wird verhindert.

Abschließend wollen wir zur Veranschaulichung eine Satz bearbeiten. Als Beispiel diene der Satz *Der Student gab eine von der Methode unabhängige Lösung an.*

Wir zählen die Wörter irgendwie ab, etwa rückwärts. Aus den Erläuterungen zu den UC folgt, daß der Satz folgende Struktur besitzt:

Der Student gab eine von der Methode unabhängige Lösung an.

11	10	09	08	07	06	05	04	03	02 01
10	09	01	03	04	05	07	03	09	09 00
ND	*SN*	*VV*	*ND*	*AF*	*ND*	*FE*	*NA*	*DO*	*PR pp*

(Die erste Zeile ist n, die zweite $I(n)$ und die dritte $C(n)$. $K(n)$ lassen wir fort!) Es ist $N = 11$.

Wir werden auf folgende Bedingungen stoßen, denen wir folgende Werte geben müssen:

Bed. (03, 08) : erfüllt, Bed. (03,10) : erfüllt,
Bed. (06, 01) : erfüllt.

Nach dem Flußdiagramm erhalten die Variablen z, q, h, m zunächst die Werte 01, 01, 00, 00, danach p den Wert 01. Die erste Entscheidung fällt negativ aus, da $C_{01}(01) = pp$ und nicht MM ist. Anschließend wird n 01, die ersten beiden Fragen werden bejaht, ebenso die dritte, da $I(01) = 00 = m$ ist. Die Bed. (01, 01) ist automatisch efüllt, und wegen $Q_{01}(01) = 02$ kommen wir zu

den Operationen $p \Rightarrow \bar{p}(h)$ usw. Der Speicher erhält als Inhalt

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
00	01	01	01

m wird 01, h wird 01 und q wird 02. p wird anschließend 01, die darauffolgende Frage wird verneint und n wird 01. Da nur für $n = 09$ $I(n) = 01 = m$ ist, wird n solange erhöht, bis wir $n = 09$ erreichen. Wegen $C_{02}(01) = VV = C(09)$ kommen wir dann zu $p \Rightarrow \bar{p}(h)$ usw.. Nach diesen Operationen steht im Speicher:

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
00	01	01	01
01	01	02	09

m wird 09, h wird 02 und q wird 03. Das bedeutet, daß wir nun zur OR $q = 03$ übergehen. Wir durchlaufen diese Reihe und kommen das erste Mal zu der Frage $C(p) = C(n)$ bei den Werten $q = 03$, $p = 01$, $n = 02$. Wegen $C_{03}(01) = \text{KS} \neq PR = C(02)$ wird diese Frage jedoch verneint. Das erste Mal ist $C(p) = C(n)$ bei der Kombination $q = 03$, $p = 08$, $n = '10$ gültig. Die Bed. (03,08) nehmen wir als erfüllt an, wie schon gesagt wurde. Wegen $Q_{03}(08) = 04 \neq 00$ kommen wir wieder zu $p \Rightarrow \bar{p}(h)$ usw.. Der Speicher enthält dann

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
00	01	01	01
01	01	02	09
02	08	03	10

m wird 10, h wird 03 und q wird 04. Die nächste Kombination, bei der wir den ja-Ausgang bei $C_q(p) = C(n)$ erreichen, ist $q = 04$, $p = 02$, $n = 11$. Wegen $Q_{04}(02) = 00$ wird $Z(11)$ zu 01, da z immer noch den Wert 01 hat. Anschließend wird z zu 02. Das Wort mit $n = 11$ (*Der*) wird also das erste Wort des Satzes. Danach durchlaufen wir die OR $q = 04$ bis zum Wert $p = 04$. Die nächste interessante Kombination ist $q = 04$, $p = 04$; dort ist $C_q(p) = MM$ gültig. Da m den Wert 10 hat, wird $Z(10)$ zu 02, *Student* also das zweite Wort. Schließlich erreichen wir bei $q = 04$, $p = 09$, $n = 11$ die Frage $C_q(p) = ZZ$, die mit ja beantwortet wird. Wegen $h = 03 \neq 01$ wird h jetzt 02, ferner wird p zu 08, q zu 03 und n zu 10, m wird 09, da $\bar{n}(01) = 09$ ist. Danach werden die Eintragungen bei $h = 02$ gelöscht:

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
00	01	01	01
01	01	02	09

Wir geben jetzt nur noch die wichtigsten Schritte an: Für $q = 03$, $p = 10$ gilt $C_q(p) = MM$. Da wir Bed. (03, 10) als erfüllt annehmen, wird $Z(09)$ zu 03, da $Q_{03}(10) = 00$ ist. z wird zu 04, und wir suchen in der Reihe $q = 03$ weiter. Für $q = 03$, $p = 15$ und $n = 03$ wird $C_q(p) = C(n)$ mit ja beantwortet. Der Speicherinhalt wird verändert:

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
00	01	01	01
01	01	02	09
02	15	03	03

Wegen $Q_{03}(15) = 04$ wird q zu 04, ferner h zu 03, und m wird zu 03. Den ja-Ausgang der Frage $C_q(p) = C(n)$ erreichen wir das erste Mal bei den Werten $q = 04$, $p = 02$ und $n = 08$. Wegen

$Q_{04}(02) = 00$ erhält $Z(08)$ den Wert 04, und z wird um 1 auf 05 erhöht. Bei $q = 04$, $p = 03$ und $n = 04$ gilt wieder $C_q(p) = C(n)$. Wegen $Q_{04}(03) = 06$ wird der Speicherinhalt verändert:

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
00	01	01	01
01	01	02	09
02	15	03	03
03	03	04	04

m wird zu 04, h wird 04, q wird 06. Da wir Bed. (06, 01) als erfüllt annehmen und $C_{06}(01) = C(07)$ ist, wird der Speicherinhalt bei deiser Kombination wieder verändert. Es wird eingetragen

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
.	.	.	.
.	.	.	.
.	.	.	.
04	01	06	07

m wird 07, h wird 05 und q wird 05. Für $q = 05$ und $p = 01$ gilt $C_q(p) = MM$. m ist 07, daher wird $Z(07)$ zu 05, z wird zu 06. (*von* wird also das fünfte Wort.) Für $q = 05$, $p = 02$ und $n = 05$ gilt $C_q(p) = C(n)$. Der Speicher erhält jetzt zusätzlich

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
.	.	.	.
.	.	.	.
.	.	.	.
05	02	05	05

Es wird m zu 05, h zu 06 und q zu 04.

Es dürfte klar geworden sein, wie der Algorithmus arbeitet. Daher fassen wir ganz kurz zusammen, wie die Bearbeitung weiterläuft: Der Speicher enthält jetzt

h	$\bar{p}$	$\bar{q}$	$\bar{n}$
00	01	01	01
01	01	02	09
02	15	03	03
03	03	04	04
04	01	06	07
05	02	05	05

Daher werden nacheinander die Reihen $q = 04, 05, 06, 04, 03$ und 02 bis zum Ende durchlaufen, und jedesmal, wenn wir bei ZZ in einer Reihe ankommen, wird h um 1 verringert. Schließlich kommen wir über die Frage $h = 01$ zum Stopbefehl. Der Satz ist dann vollständig bearbeitet.

Wir schließen nun noch einige Bemerkungen an:

Die inneren Satzzeichen (Kommata) werden bei diesem Algorithmus nicht erzeugt. (Sie müßten erzeugt werden, da sie im Baum nicht enthalten sind!) Man kann aber eine ganz einfache Regel angeben: Ein Komma muß immer dann gesetzt werden, wenn wir die Reihe $q = 03$ beginnen oder beenden und wenn $C(m) \neq VV$ ist.

Defektive Sätze können nach diesem Schema nicht behandelt werden. Hier müssen noch andere Prinzipien angewendet werden, die wir hier nicht erörtern.

Die Suchprozesse sind nicht optimal, d. h., es werden sehr viele überflüssige Suchoperationen durchgeführt. Durch Änderungen des Flußdiagramms lassen sich Verbesserungen in dieser Hinsicht erreichen.
Es gilt folgender Satz, den wir hier nicht beweisen:

Wird bei der Bearbeitung eines Baumes B niemals eine Kombination p, q, n, m erreicht, für die $Z(n) = 00$, $T_q(p) = 1$, $C_q(p) = C(n)$ und Bed. (q, p) erfüllt ist, so ist B bezüglich der Ordnung $Z(n)$ einfach projektiv geordnet. – In unseren Beispielsatz war dies der Fall.

Eine nicht-projektive Worstellung weist z. B. folgender Satz auf:

Der Student gab eine Lösung an, die von der Methode abhängt.

Projektiv, aber stilistisch schlecht ist

Der Student gab eine Lösung, die von der Methode abhängt, an.

Der erste Satz enthält einen verzögerten Attributsatz, der sich hinsichtlich der Reihenfolge $Z(n)$ nicht unmittelbar an das Beziehungswort anschließt.

Der geschilderte Algorithmus gibt selbstverständlich nicht alle möglichen grammatisch richtigen Wortstellungen an, die mit dem geschilderten Inventar an UC gebildet werden können. Für die automatische Übersetzung ist das auch nicht nötig, solange man die syntaktische Synthese bearbeitet. Durch Vervollständigung der OR und weitere Bedingungen (q, p) kann man jedoch alle Wortstellungen erhalten.

Der Algorithmus enthält auch nicht alle für das Deutsche notwendigen Ordnungsreihen.

Der im Übersetzungsexperiment unserer Arbeitsstelle aus dem Jahre 1963 angewendete Algorithmus arbeitete gewissermaßen nur mit zwei Ordnungsreihen: Die eine war praktisch die, die jetzt bei $q = 03$ steht. Alles, was im Baum unterhalb der Glieder aus dieser Reihe stand, wurde in die Reihenfolge gebracht, wie sie sich aus dem Englischen ergab. Der hier geschilderte Algorithmus ist einfacher und formal besser überschauber.

(Eingegangen am 16. September 1964.)

LITERATUR

[1] Meltschuk: Über einen Algorithmus der syntaktischen Analyse sprachlicher Texte. Automatische Übersetzung und Angewandte Linguistik (1962), Heft 7.

K syntaktické syntéze

JÜRGEN KUNZE

Na počátku autor studuje popis vět pomocí závislostních stromů. Jejich závislostní charakteristiky (Unterordnungscharakteristiken, zkr. UC) představují precisování a zobecnění takových pojmů jako je větný člen, rozšíření atd. Každé slovo (s výjimkou slov stojících na vrcholu stromu) je podřazeno nějakému jinému slovu pomocí určité UC. Jednotlivé UC popisují syntaktický vztah mezi příslušnými dvěma slovy.

Toto pojetí nachází své uplatnění ve strojovém překladu. Při syntaktické analýze vstupního jazyka je věta vstupního jazyka přeměna ve strom, tj. každé slovo je podřazeno nějakému jinému a je spojeno s určitou UC. Tento strom obsahuje gramatické vztahy vstupního jazyka. Následujícími změnami konstrukcí je pak přeměněn ve strom nový, obsahující vztahy jazyka výstupního.

Dále je třeba slova tohoto stromu uspořádat tak, aby jejich sled odpovídal slovosledným pravidlům výstupního jazyka (zde němčiny). Článek podává podrobný popis tohoto postupu. Algoritmus pro vytvoření slovosledu užívá UC a hierarchického uspořádání slov ve větě, daného závislostními charakteristikami. Sestává se s jediného jednoduchého logického schématu s jedním zásobníkem (pushdown store) a z několika uspořádaných posloupností UC. Touto metodou mohou být vytvářena také neprojektivní slovní uspořádání.

Navržený algoritmus představuje další rozšíření metody, které bylo pracovní skupinou, do níž autor náleží, použito při experimentu v květnu 1963.

Dipl.-Math. Jürgen Kunze, Arbeitsstelle für Mathematische und Angewandte Linguistik und Automatische Übersetzung der Deutschen Akademie der Wissenschaften, Mohrenstraße 39, Berlin W 8, DDR.