EUR 5053 d

KOMMISSION DER EUROPÄISCHEN GEMEINSCHAFTEN

AUTOMATISCHE SUFFIXANALYSE

von

M.PFEIFER

1974

Gemeinsame Kernforschungsstelle
Forschungsanstalt Ispra - Italien

Zentralstelle für die Verabeitung
wissenschaftlicher Informationen (CETIS)

HINWEIS

Das vorliegende Dokument ist im Rahmen des Forschungsprogramms der Kommission der Europäischen Gemeinschaften ausgearbeitet worden.

Es wird darauf hingewiesen, daß die Kommission der Europäischen Gemeinschaften, ihre Vertragspartner und die in deren Namen handelnden Personen :

keine Gewähr dafür übernehmen, daß die in diesem Dokument enthaltenen Informationen richtig und vollständig sind oder daß die Verwendung der in diesem Dokument enthaltenen Informationen oder der in diesem Dokument beschriebenen technischen Anordnungen, Methoden und Verfahren nicht gegen gewerbliche Schutzrechte verstößt;

keine Haftung für die Schäden übernehmen, die infolge der Verwendung der in diesem Dokument enthaltenen Informationen oder der in diesem Dokument beschriebenen technischen Anordnungen, Methoden oder Verfahren entstehen könnten.

Dieser Bericht wird in den auf der vierten Umschlagseite genannten Vertriebsstellen

zum Preise von BF 150.—

verkauft.

Kommission der
Europäischen Gemeinschaften
GD XIII - ZID
29, rue Aldringen
L u x e m b o u r g
Februar 1974

EUR 5053 d

AUTOMATISCHE SUFFIXANALYSE von M. PFEIFFER

Kommission der Europäischen Gemeinschaften
Gemeinsame Kernforschungsstelle Forschungsanstalt Ispra — Italien
Zentralstelle für die Verarbeitung wissenschaftlicher Informationen-CETIS
Luxemburg, Februar 1974 – 106 Seiten — BF 150. -

Die vorliegende Untersuchung beschreibt Methoden der "automatischen Suffixanalyse", und insbesondere den Ansatz, der bei dem früheren Indexing-Versuchsprojekt des CETIS gewählt wurde. Sie enthält eine allgemeine mathematische Darstellung des Problems unter Benutzung der in der Mengenlehre gebräuchlichen Notationen. Einige Beispiele aus der Literatur werden erörtert, und es wird eine Beurteilung unter linguistischen Gesichtspunkten gegeben. Schließlich wird ein Maschinenprogramm beschrieben, das alle bisherigen Ansätze handhaben kann.

EUR 5053 d

AUTOMATIC SUFFIX ANALYSIS, by M. PFEIFFER

Commission of the European Communities
Joint Nuclear Research Centre – Ispra Establishment (Italy)
Scientific Data Processing Centre CETIS
Luxembourg, February 1974 — 106 Pages — B.Fr. 150.—

The present study describes methods of "Automatic Suffix Analysis" and, in particular, reviews the approach chosen in the previous experimental indexing project of CETIS. It contains a general mathematical representation of the problem using set-theoretical notations. Some example found in the literature are discussed and an evaluation from the linguistical point of view is given. Finally, a machine program is described which is able to handle all different approaches found.

EUR 5053 d

AUTOMATIC SUFFIX ANALYSIS, by M. PFEIFFER

Commission of the European Communities
Joint Nuclear Research Centre – Ispra Establishment (Italy)
Scientific Data Processing Centre CETIS
Luxembourg, February 1974 — 106 Pages — B.Fr. 150.—

The present study describes methods of "Automatic Suffix Analysis" and, in particular, reviews the approach chosen in the previous experimental indexing project of CETIS. It contains a general mathematical representation of the problem using set-theoretical notations. Some example found in the literature are discussed and an evaluation from the linguistical point of view is given. Finally, a machine program is described which is able to handle all different approaches found.

EUR 5053 d

AUTOMATIC SUFFIX ANALYSIS, by M. PFEIFFER

Commission of the European Communities
Joint Nuclear Research Centre — Ispra Establishment (Italy)
Scientific Data Processing Centre CETIS
Luxembourg, February 1974 — 106 Pages — B.Fr. 150.—

The present study describes methods of "Automatic Suffix Analysis" and, in particular, reviews the approach chosen in the previous experimental indexing project of CETIS. It contains a general mathematical representation of the problem using set-theoretical notations. Some example found in the literature are discussed and an evaluation from the linguistical point of view is given. Finally, a machine program is described which is able to handle all different approaches found.

EUR 5053 d

KOMMISSION DER EUROPÄISCHEN GEMEINSCHAFTEN

AUTOMATISCHE SUFFIXANALYSE

von

M.PFEIFER

1974

Gemeinsame Kernforschungsstelle
Forschungsanstalt Ispra - Italien

Zentralstelle für die Verabeitung
wissenschaftlicher Informationen (CETIS)

Diplomarbeit vorgelegt
bei der Fakultät für Mathematik
der Universität Stuttgart

ZUSAMMENFASSUNG

Die vorliegende Untersuchung beschreibt Methoden der "automatischen Suffixanalyse", und insbesondere den Ansatz, der bei dem früheren Indexing-Versuchsprojekt des CETIS gewählt wurde. Sie enthält eine allgemeine mathematische Darstellung des Problems unter Benutzung der in der Mengenlehre gebräuchlichen Notationen. Einige Beispiele aus der Literatur werden erörtert, und es wird eine Beurteilung unter linguistischen Gesichtspunkten gegeben. Schließlich wird ein Maschinenprogramm beschrieben, das alle bisherigen Ansätze handhaben kann.

SCHLAGWORTER

UNIVERSITIES
MATHEMATICS
INDEXES
STANDARDIZED TERMINOLOGY
AUTOMATION
INFORMATION
PROGRAMMING
FORTRAN
IBM COMPUTERS

The present report is a first publication in the context of the cooperation between the

Information Science Research Unit of the
European Scientific Information Processing Centre (CETIS) of the
Joint Research Centre of the
Commission of the European Communities in Ispra,

and the

Institute for Informatics (IFI) of the
University of Stuttgart.

This diploma-thesis (Diplom-Arbeit) is presented to the

Faculty of Mathematics of the
University of Stuttgart.

Although such theses usually are not published (because of the limited time schedule in which they have to be compiled), it was deemed worthwhile to record this one in contrast to the practice for two reasons:

- A large amount of time was dedicated to discussions and consultations by the responsible persons, and
- the argument treated is of great interest in the field of information science.

This thesis describes methods of "Automatic Suffix Analysis", and, in particular, reviews the approach chosen in the previous experimental indexing project of CETIS. In the present automatic indexing system this approach has been abandoned in favour of a stem-suffix analysis method developed in the machine-translation context.

In automatic text processing, a suffix analysis is undertaken to standardize word incidences without a stem dictionary control. In many applications this relatively simple approach yields acceptable results such that more sophisticated methods need not be elaborated.

The representation of the problem is purely mathematical, and, as

such, has the advantage of allowing a unique description of the different approaches used in the literature. Therefore, the effort necessary for building a mathematical model seems to be justified.

Although any kind of evaluation in information science turns out to be somewhat dubious, the conclusions of this thesis seems to confirm the frequent observation, that, at this level, rudimentary from the linguistic point of view, the simplest solutions give the best results. Thus, for instance, the refined suffix analysis technique described by LOVINS is not as good as the less elaborate method used in the CETIS project.

The problems connected with automatic dictionary search and morphological analysis, in principle, can be considered to be resolved. However, only for a few languages a concrete analysis and the relative morphology and dictionaries, in the form of operational grammars and data bases, are available.

Therefore, the present work may very well become the first of a series, especially in consideration of the fact that automatic suffix analysis is extremely useful for semi-automatic stem-affix dictionary construction.

Dipl. Math. H. Fangmeyer

for the
Information Science Research Unit
CETIS

Dr. H. J. Schneider

for the
Institute for Informatics
University of Stuttgart
IFI

Herrn Professor Dr. W. Knödel danke ich dafür, dass ich diese Arbeit an dem Institut für Informatik der Universität Stuttgart durchführen konnte. Vor allem danke ich Herrn Dr. H. J. Schneider für die grosszügige Unterstützung und für die Vermittlung eines Praktikums bei EURATOM (Cetis), Ispra, Italien, wo diese Arbeit entstand.

Mein besonders herzlicher Dank gilt Herrn Dipl. Math. H. Fangmeyer (EURATOM, Ispra), der mich während des Praktikums betreute, in unermüdlicher Bereitschaft zu Gesprächen mir viele wertvolle Hinweise und Anregungen vermittelte und alle notwendigen Unterlagen zur Verfügung stellte.

Inhalt — Seite

0 Einleitung

Bei der automatischen Verarbeitung von Texten in natürlicher Sprache - zum Beispiel bei automatischer Sprachübersetzung, Dokumentation oder Wörterbucherstellung - ist es erforderlich, Texteinheiten zu erkennen, um ihnen Information zuordnen zu können. Dies geschieht durch einen automatischen Vergleich der Texteinheiten mit einem gespeicherten Wörterbuch, das für eine gegebene Anwendung die entsprechenden Informationen (Daten) enthält.

Zu den Texteinheiten zählen u. a. die Wörter der Sprache, in der ein Text abgefasst ist.

Wörter treten auf in flektierten und abgeleiteten Formen, d. h. ihre Zeichenketten stimmen, trotz ähnlichen Informationsgehaltes, nicht überein. Die Verschiedenheit äussert sich oft durch unterschiedliche Affixe (Präfixe, Suffixe) unter Beibehaltung eines gemeinsamen Stammes.

Die semantische Information eines Wortes ist im wesentlichen im Wortstamm enthalten, während die Affixe hauptsächlich eine grammatikalische Information beinhalten. (Dies gilt insbesondere für Flexions-Affixe.)

1 Ansätze zur Worterkennung

1.1 Vergleich mit einem Stammwörterbuch

Ein Stammwörterbuch enthält die Wortstämme einer Sprache. Jedem Wortstamm ist eine semantische Information, entsprechend der Anwendung, zugeordnet.

Ein Wort eines zu analysierenden Textes wird mit dem Stammwörterbuch verglichen. Stimmt der Stamm des Textwortes mit einem der Stämme des Wörterbuches überein, so wird ihm die mit dem Stamm gespeicherte Information zugeordnet.

Eine Anwendung dieser Methode ist nur sinnvoll, wenn dabei mehr Wert auf die semantische Information gelegt wird als auf die grammatikalische.

1.2 Suffixanalyse

Bei einer Suffixanalyse wird kein Stammwörterbuch vorausgesetzt, vielmehr werden Wortstämme erst erzeugt.

Vorgegeben wird eine Liste von Suffixen einer Sprache. Durch feste Regeln werden die Suffixe eines Wortes von diesem abgetrennt; die den Suffixen inhärente grammatikalische Information kann dem Wort zugeordnet werden.

Eine automatische Suffixanalyse ist eine wesentliche Hilfe bei der Erstellung von Stammwörterbüchern.

1.3 Stamm-Suffix-Methode

Mit einem kombinierten Ansatz, bei dem sowohl ein Stammwörterbuch als auch eine Suffixliste vorgegeben werden, erzielt man die besten Ergebnisse.

Zwei Beispiele einer solchen Methode seien hier erwähnt. Sie werden angewandt auf die englische Sprache.

1. SLC II (EURATOM) (15, 16)

   Es wird ein Stammwörterbuch vorgegeben, welches ausser den Wortstämmen selbst noch eine Liste der für Wortbildungen mit einem Stamm zulässigen Suffixe (und deren grammatikalische Information) enthält. Beispiel:

   ION (IZE, IZED, IZATION, ...)

   Ein zu analysierendes Wort wird genau dann erkannt, wenn sein Stamm mit einem der Stämme des Stammwörterbuches überein-

stimmt, und seine Endung eines der für diesen Wortstamm zulässigen Suffixe ist.

2. SMART System (SALTON) (20)

Ein Stammwörterbuch und eine Liste mit ca. 200 zulässigen Suffixen werden vorgegeben.

Ein Wort wird genau dann erkannt, wenn mindestens eine von fünf Bedingungen erfüllt ist. [1)]

Die vorliegende Arbeit befasst sich im folgenden mit einer automatischen Suffixanalyse.

[1)] Bedingungen (Persönliche Mitteilung von Prof. G. SALTON)

(1) Das Wort stimmt mit dem Wörterbucheintrag genau überein.

(2) Die ersten zwei oder mehr Zeichen des Wortes stimmen mit dem Wörterbucheintrag überein und der Rest des Wörterbucheintrages ist ein E und der Rest des Wortes ist ein zulässiges Suffix, welches mit A, E, I oder O beginnt.

(3) Die ersten drei oder mehr Zeichen des Wortes stimmen mit dem gesamten Wörterbucheintrag überein und der Rest des Wortes ist ein zulässiges Suffix.

(4) Die ersten drei oder mehr Zeichen des Wortes stimmen mit dem gesamten Wörterbucheintrag überein und das nächste Zeichen des Wortes stimmt mit dem vorangehenden überein und wird gefolgt von einem zulässigen Suffix.

(5) Die ersten zwei oder mehr Zeichen eines Wortes stimmen mit denen des Wörterbucheintrages überein und der Rest des Wörterbucheintrages ist ein Y und der Rest des Wortes besteht aus einem I, dem ein zulässiges Suffix folgt.

Beispiele:	Wort	Wörterbucheintrag	Suffix
zu (1)	LENGTH	LENGTH	
(2)	FILED	FILE	ED
(3)	REPORTS	REPORT	S
(4)	PUTTING	PUT	ING
(5)	COMPLIANCE	COMPLY	ANCE

Die vollständige Suffixliste ist aufgeführt in 3. 1. 2 (Menge E).

2. Mathematische Beschreibung einer Suffixanalyse

Definitionen der im folgenden verwendeten Symbole:

$\subset$ Teilmenge

$\cup$ Vereinigung von Mengen

$\cap$ Durchschnitt von Mengen

$\setminus$ Mengendifferenz

$\emptyset$ Leere Menge (Nullmenge)

$\in$ "ist als Element enthalten in der Menge"

$\prec$ Ordnung in einer Menge (strikte Totalordnung)

$\rightarrow$ Abbildung (f: A$\rightarrow$B)

$\sum$ Summe

$\mathbb{N}$ Menge der natürlichen Zahlen

$|A|$ Mächtigkeit der (endlichen) Menge A; Kardinalzahl

$:=$ "ist nach Definition gleich"

∞ unendlich

$\wedge$ Konjunktion

$\vee$ Disjunktion

$\neg$ Negation

$\Rightarrow$ Implikation

$\Leftrightarrow$ Äquivalenz

$\forall_x$ Allquantor

$\exists_x$ Existenzquantor

$=$ Gleichung

$\equiv$ Identität

$\|$ Verknüpfung von Zeichenketten

Durch eine Suffixanalyse wird zu jedem Wort einer Sprache ein Wortstamm gebildet.

Die Zuordnung Wort ⟿ Wortstamm kann durch eine Abbildungsfunktion mathematisch beschrieben werden.

2.1 Zeichenketten

Z sei die Menge aller Zeichen z^i einer Sprache.

$$Z = \{ \pm\ .\ ,\ )\ \&\ +\ \ldots\ abcde\ \ldots\ 0123\ \ldots\ 9 \}$$

Das Zeichen ± sei das Nil-Zeichen.

(Die Elemente der Menge Z sind nicht durch Kommata getrennt, da das Komma selbst Element der Menge ist.)

Definition von Zeichenketten:

(1) Jedes Zeichen $z^i \in Z$ ist eine Zeichenkette.
Die Zeichenkette ± ist die Null-Zeichenkette.

(2) Jede endliche "Kombination (Aneinanderreihung) von Zeichenketten" ist wieder eine Zeichenkette. Bei einer Kombination mit der Null-Zeichenkette bleibt eine Zeichenkette unverändert.

K sei die Menge aller Zeichenketten; Z ist eine Untermenge von K ($Z \subset K$).

Eine "Kombination (Aneinanderreihung) von Zeichenketten" ist eine assoziative, nicht kommutative Verknüpfung in K. Wir verwenden für diese Verknüpfung das Symbol ||.

Die algebraische Struktur (K, ||) ist eine Halbgruppe, da das Assoziativgesetz erfüllt ist:

$$a_1 \| (a_2 \| a_3) = (a_1 \| a_2) \| a_3$$

Es erübrigt sich daher, die Klammern zu schreiben, und wir können folgende abkürzende Schreibweise vereinbaren:

$$a_1 \| a_2 \| a_3 \| \dots \| a_n = \prod_{i=1}^{n} \| a_i \qquad (a_i \in K)$$

Diese Darstellung ist eineindeutig, da die Verknüpfung nicht kommutativ ist.

Weiter vereinbaren wir:

a) Alle Zeichenketten mit unteren Indizes sind beliebige Zeichenketten aus K.

b) Alle Zeichenketten mit oberen Indizes sind Zeichenketten aus Z. (In einigen Fällen, vor allem in Beispielen, werden diese speziellen Zeichenketten ohne das Verknüpfungssymbol $\|$ geschrieben; in diesen Fällen verwenden wir lateinische Grossbuchstaben. Beispiel:

$$\text{HAND} = h \| a \| n \| d = z^i \| z^j \| z^k \| z^l \quad)$$

Bemerkung: Die Gruppenaxiome sind für $(K, \|\ )$ nicht erfüllt. Es existiert zwar eindeutig das neutrale Element $\pm$,

$$a \| \pm = \pm \| a = a \ ,$$

es gibt jedoch zu einem $a \in K$ kein inverses Element $\bar{a} \in K$, für das

$$a \| \bar{a} = \bar{a} \| a = \pm$$

ist.

Definition der Länge einer Zeichenkette:

a sei eine Zeichenkette:

$$a = \prod_{i=1}^{p} \| a^i \qquad (a^i \in Z \ ; \ p \in \mathbb{N})$$

Dann ist $l(a)$ die Länge von a, und es gilt:

$$\boxed{\begin{array}{l} l(a) := p \\ l(\pm) := 0 \end{array}} \qquad (0 < p < \infty)$$

2.2 Zerlegung eines Wortes

Definition der Begriffe Wort, Wortendung, Wortkondition und Restwort:

Wort: Ein Wort ist eine Zeichenkette, die kein Nil-Zeichen enthält:

$$w := \mathop{||}_{i=1}^{n} w^i \qquad (n \in \mathbb{N} ; w^i \in Z ; w^i \neq \pm)$$

Die Länge eines Wortes ist:

$$l(w) = n$$

Beispiel: w = HAND $l(w) = n = 4$

Wortendung: Die Endung eines Wortes w ist eine Zeichenkette, die - wie folgt - definiert wird:

$$e_k(w) = \mathop{||}_{i=1}^{n-k} e^i := \mathop{||}_{i=k+1}^{n} w^i \qquad (0 \leq k \leq n-1 ; n=l(w))$$

Nach dieser Definition gibt es n verschiedene Wortendungen eines Wortes der Länge n.

Zusätzlich wird eine "Null-Wortendung" $e_n(w)$ definiert:

$$e_n(w) := \pm$$

Alle n+1 Wortendungen des Wortes w fassen wir zusammen in der Menge E_w:

$$E_w = \{ e_k(w) \;/\; k=0,\ldots,n \}$$

Für die Länge einer Wortendung gilt:

$$l(e_k(w)) = n-k$$

Beispiel: w = HAND (l(w) = n = 4)

k	$e_k(w)$	$l(e_k(w))$
0	HAND	4
1	AND	3
2	ND	2
3	D	1
4	$\pm$	0

Wortkondition: Die Kondition (condition code) eines Wortes w - mit der Wortendung $e_k(w)$ - ist eine Zeichenkette, die folgendermassen definiert wird:

$$c_{k,m}(w) = c_{k,m}(e_k(w)) = \prod_{i=1}^{k-m} c^i := \prod_{i=m+1}^{k} w^i \qquad (0 \leq m < k)$$

$$c_{k,m}(w) := \pm \quad \forall_{c_{k,m}} \; (m=k)$$

"Null-Wortkondition"

Zu der k-ten Wortendung $e_k(w)$ gibt es maximal k+1 verschiedene Wortkonditionen.

Ein Wort w mit n+1 Wortendungen hat damit maximal

$$1 + \sum_{q=1}^{n} q = 1 + \binom{n+1}{2} = 1 + \frac{n(n+1)}{2}$$

verschiedene Wortkonditionen.

Alle $c_{k,m}(w)$ bilden die Menge C_w:

$$C_w = \left\{ c_{k,m}(w) \;/\; k=0,\ldots,n \;;\; m=0,\ldots,k \right\}$$

Die Länge einer Wortkondition $c_{k,m}(w)$ ist:

$$l(c_{k,m}(w)) = k-m$$

Beispiel: $w = HAND$ $(l(w) = n = 4)$

k	$c_{k,0}(w)$	$c_{k,1}$	$c_{k,2}$	$c_{k,3}$	$c_{k,4}$	$e_k(w)$
0	±					HAND
1	H	±				AND
2	HA	A	±			ND
3	HAN	AN	N	±		D
4	HAND	AND	ND	D	±	±

Restwort: Der Rest eines Wortes w - mit der Wortendung $e_k(w)$ und der Wortkondition $c_{k,m}(w)$ - ist eine Zeichenkette, die auf folgende Art definiert wird:

$$r_{k,m}(w) = r_{k,m}(c_{k,m}(e_k(w))) = \prod_{i=1}^{m} r^i := \prod_{i=1}^{m} w^i \qquad (0 < m \leq k)$$

$$r_{k,m}(w) := \pm \quad \bigvee_{r_{k,m}}(m=0)$$

"Null-Restwort"

Beispiel: $w = HAND$ $(l(w) = n = 4)$

k	$r_{k,0}$ $c_{k,0}$	$r_{k,1}$ $c_{k,1}$	$r_{k,2}$ $c_{k,2}$	$r_{k,3}$ $c_{k,3}$	$r_{k,4}$ $c_{k,4}$	$e_k(w)$
0	± ±					HAND
1	± H	H ±				AND
2	± HA	H A	HA ±			ND
3	± HAN	H AN	HA N	HAN ±		D
4	± HAND	H AND	HA ND	HAN D	HAND ±	±

Es gibt also n+1 verschiedene Restwörter, sie sind nur abhängig von m $(0 \leq m \leq k)$:

$$r_m(w) = r_m(c_{k,m}(e_k(w))) = \prod_{i=1}^{m} r^i = \prod_{i=1}^{m} w^i \qquad (0 < m \leq k)$$

$$r_m(w) = \pm \quad \bigvee_{r_m}(m=0)$$

Alle n+1 Restwörter eines Wortes w bilden die Menge R_w:

$$R_w = \left\{ r_m(w) \;/\; k=0,\ldots,n \;;\; m=0,\ldots,k \right\}$$

Die Länge eines Restwortes ist:

$$\boxed{l(r_m(w)) = m}$$

Die so definierten Mengen E_w, C_w und R_w sowie die Menge $\{w\}$ sind Untermengen der Menge K aller Zeichenketten. Mit Hilfe der in K definierten Verknüpfung wird ein Wort - wie folgt - dargestellt:

$$\boxed{w = r_m(w) \parallel c_{k,m}(w) \parallel e_k(w)}$$

Diese Darstellung nennen wir Zerlegung des Wortes w. Eine solche Zerlegung ist nicht eindeutig bestimmt, vielmehr existieren für ein Wort w der Länge n genau

$$\sum_{i=1}^{n+1} i = \binom{n+2}{2} = \frac{(n+1)(n+2)}{2}$$

verschiedene Zerlegungen.

Beispiel: $w = \text{HAND}$ $(l(w) = n = 4)$

Nach obiger Formel gibt es 15 verschiedene Zerlegungen:

	Wort w		
k, m	Restwort $r_m(w)$	Wortkondition $c_{k,m}(w)$	Wortendung $e_k(w)$
0, 0	±	±	HAND
1, 0	±	H	AND
1, 1	H	±	AND
2, 0	±	HA	ND
2, 1	H	A	ND
2, 2	HA	±	ND
3, 0	±	HAN	D
3, 1	H	AN	D
3, 2	HA	N	D
3, 3	HAN	±	D
4, 0	±	HAND	±
4, 1	H	AND	±
4, 2	HA	ND	±
4, 3	HAN	D	±
4, 4	HAND	±	±

2.3 Zulässigkeit einer Zerlegung

An eine Wortzerlegung

$$w = r_m(w) \parallel c_{k,m}(w) \parallel e_k(w)$$

knüpfen wir folgende Bedingungen:

(1) Eine endliche Menge E von Suffixen e_i wird vorgegeben; die e_i sind Zeichenketten. Die Null-Zeichenkette $\pm$ sei in E enthalten.
Eine Zerlegung von w ist nur dann zulässig, wenn die ihr zugeordnete Wortendung $e_k(w) \in E_w$ auch Element der vorgegebenen Menge E ist, d.h. $e_k(w)$ muss enthalten sein im Durchschnitt von E und E_w:

$$B_1 := e_k(w) \in E \cap E_w$$

Diese Bedingung kann erfüllt sein für mehrere Zerlegungen, sie ist immer erfüllt für die Zerlegung mit $e_n(w) = \pm$.

(2) Für jede Zerlegung eines Wortes wird gefordert, dass die Zeichenkette

$$r_m(w) \parallel c_{k,m}(w)$$

(Restwort plus Wortkondition) eine festgelegte Mindestlänge hat.

Sei $l(w) = n$, dann ist $l(e_k(w)) = n-k$,
$l(c_{k,m}(w)) = k-m$ und
$l(r_m(w)) = m$.

Daraus ergibt sich als Länge von Restwort plus Wortkondition:

$$l(r_m(w)) + l(c_{k,m}(w)) = m + k - m = k \qquad (0 \leq k \leq n)$$

Eine Zerlegung eines Wortes w ist nur dann zulässig, wenn folgende Bedingungen erfüllt sind:

a) Falls die Länge n des Wortes kleiner ist als eine Konstante q, so ist die Länge k gleich n,

$$n < q \Rightarrow k = n \ ,$$

d.h. es sind nur Zerlegungen mit der Null-Wortendung zulässig ($e_n(w) = \pm$) .

b) Falls die Länge des Wortes grösser oder gleich der Konstanten q ist, so ist die Länge k mindestens gleich q:

$$n \geq q \Rightarrow k \geq q$$

Wir fassen diese beiden Bedingungen zusammen:

$$B_2 := \left[n < q \Rightarrow k = n\right] \wedge \left[n \geq q \Rightarrow k \geq q\right]$$

(3) Die Bedingung B_2 ist eine generelle Forderung an die Länge von Restwort plus Wortkondition für Zerlegungen mit beliebigen Wortendungen $e_k(w)$. Die folgende Bedingung sieht für Zerlegungen mit bestimmten Endungen $e_i \in E$ eine individuell festzulegende Längenbedingung vor. Wir bezeichnen diese - von Suffixen abhängige - Mindestlänge mit $q(e_i)$.

Es wird gefordert:

$$l(r_m(w)) + l(c_{k,m}(w)) = k \geq q(e_k(w))$$

Für alle $e_i \in E$, für die ein $q(e_i)$ definiert ist, bilden wir die Paare $(e_i, q(e_i))$ und fassen sie in der Menge Q zusammen.

Eine Zerlegung eines Wortes w ist nur dann zulässig, wenn die folgende Bedingung erfüllt ist:

$$B_3 := (e_k(w), q(e_k(w))) \in Q \Rightarrow k \geq q(e_k(w))$$

(4) Eine endliche Menge C von Konditionen (condition codes) wird vorgegeben.

Die Elemente C_i von C sind Mengen von Zeichenketten, die - wie folgt - gebildet werden können:

a) Mengen, die genau eine Zeichenkette aus K enthalten.

Beispiel: $\{IX\}$

Verkürzte Schreibweise: IX

b) Komplementmengen von Zeichenketten bezüglich K.

Beispiel: $K \setminus \{ER\}$

Verkürzte Schreibweise: $\neg$ ER

c) Mengen, die durch Vereinigung von Mengen der in a) und b) beschriebenen Arten gebildet werden sowie deren Komplementmengen bezüglich K.

Beispiel: 1) $\{IX\} \cup (K \setminus \{ER\})$

2) $K \setminus (\{IX\} \cup (K \setminus \{ER\})) = (K \setminus \{IX\}) \cap \{ER\}$

Verkürzte Schreibweise: 1) $IX \vee \neg ER$

2) $\neg(IX \vee \neg ER) = \neg IX \wedge ER$

Wir stellen diese Mengen also dar durch Verknüpfungen von Zeichenketten; die Verknüpfungsbasis ist $(\wedge, \vee, \neg)$.

Weiter vereinbaren wir, dass diese Mengen im folgenden repräsentiert werden durch eine natürliche Zahl.

Beispiel: $y := IX \vee \neg ER \quad (y \in \mathbb{N})$

$\neg\, y := \neg IX \wedge ER$

Die Menge $\{\pm\}$ sei in C enthalten.

Eine Zerlegung des Wortes w ist nur dann zulässig, wenn die Wortkondition $c_{k,m}(w)$ auch in einem Element der Menge C enthalten ist.

$$B_4 := c_{k,m}(w) \in C_i \wedge C_i \in C$$

Diese Bedingung ist mindestens erfüllt für $c_{k,m} = \pm$.

(5) Eine endliche Menge G von Paaren (C_i, e_j) mit $C_i \in C$ und $e_j \in E$ wird vorgegeben. Für jedes $e_j \in E$ sei mindestens ein Paar (C_i, e_j) definiert.

Das Paar $(\pm, \pm)$ sei in G enthalten.

Eine Zerlegung des Wortes w ist nur dann zulässig, wenn folgende Bedingung erfüllt ist:

$$B_5 := (C(e_k(w)), e_k(w)) \in G$$

Diese Bedingung ist mindestens erfüllt für das Paar $(\pm, \pm)$.

Die Bedingungen B_1 bis B_5 können angewandt werden in beliebiger Reihenfolge.

2.4 Eindeutigkeit einer Zerlegung

Die Bedingungen für die Zulässigkeit einer Zerlegung reduzieren die Anzahl der möglichen Zerlegungen eines Wortes w. Durch die folgende Bedingung B_6 wird die Eindeutigkeit garantiert.

(6) Wir definieren auf G eine zweistellige irreflexive, transitive und semikonnexe Relation und erhalten damit in G eine strikte Totalordnung $(G, \prec)$[1].

Als "grösstes" Element der geordneten Menge G wird das Paar $(\pm, \pm)$ definiert.

B_6 := Es seien

$$r_{m_i}(w) \parallel c_{k_i, m_i}(w) \parallel e_{k_i}(w) \qquad (i=2, 3, \ldots)$$

verschiedene zulässige Zerlegungen des Wortes w; dann wird diejenige Zerlegung ausgezeichnet, die dem kleinsten Paar $(C, e) \in G$ (im Sinne der Ordnung $\prec$) entspricht.

Ergebnis: Es sei $w = r_m(w) \parallel c_{k,m}(w) \parallel e_k(w)$ eine Zerlegung des Wortes w.

Die Zerlegung ist genau dann zulässig und umkehrbar eindeutig, wenn die Bedingungen B_1 bis B_6 erfüllt sind.

---

1) Die Elemente der Menge werden i. a. so geordnet, dass einer schärferen Bedingung stets die allgemeinere folgt. Beispiel: $(\pm$, UIDS)
$(\neg 2$, DS)
$(\pm$, S)

2.5 Endungsketten

Die Zeichenkette $r_m(w) \parallel c_{k,m}(w)$ kann nun auch als Wort interpretiert werden. Wir bezeichnen:

$$w_1 = w$$
$$w_2 = r_m(w_1) \parallel c_{k,m}(w_1) = r_{m_1}(w_1) \parallel c_{k_1,m_1}(w_1)$$

Dann gibt es für w_2 wieder genau eine zulässige Zerlegung, die die Bedingungen B_1 bis B_6 erfüllt:

$$w_2 = r_{m_2}(w_2) \parallel c_{k_2,m_2}(w_2) \parallel e_{k_2}(w_2)$$

Damit gilt:

$$w_1 = r_{m_2}(w_2) \parallel c_{k_2,m_2}(w_2) \parallel e_{k_2}(w_2) \parallel e_{k_1}(w_1)$$

Entsprechend wird $w_3 = r_{m_2}(w_2) \parallel c_{k_2,m_2}(w_2)$ nochmals zerlegt.

Dieser Prozess (Iteration) wird fortgesetzt, bis keine Zerlegung mehr entsteht, die sich von der vorhergehenden unterscheidet.

Bemerkung: Bei der Iteration wird zugelassen, dass die Mengen E, C, und G für jeden Iterationsschritt neu definiert werden. Wir kennzeichnen die Mengen dann mit oberen Indizes; so ist z. B. E^i die vorgegebene Menge E für den i-ten Iterationsschritt.

Man erhält nach r Iterationsschritten

$$w = w_1 = r_{m_r}(w_r) \parallel c_{k_r,m_r}(w_r) \parallel e_{k_r}(w_r) \parallel e_{k_{r-1}}(w_{r-1}) \parallel \ldots \parallel e_{k_1}(w_1)$$

oder

$$\boxed{w = w_1 = r_{m_r}(w_r) \parallel c_{k_r,m_r}(w_r) \parallel \Big\|_{i=1}^{r} e_{k_{r-i+1}}(w_{r-i+1})} \qquad (r \in \mathbb{N})$$

Die so definierte Endungskette $\Big\|_{i=1}^{r} e_{k_{r-i+1}}(w_{r-i+1})$ sei mit $e(w)$ bezeichnet, damit gilt:

$$\boxed{w = r_{m_r}(w) \parallel c_{k_r,m_r}(w) \parallel e(w)}$$

2.6 Der Wortstamm eines Wortes

Ein Wortstamm im morphologischen Sinn ist nicht notwendig identisch mit derjenigen Zeichenkette, die aus einem Wort durch Abtrennen von Suffixen entsteht. So erhält man z. B. aus dem Wort KNIVES den Wortstamm KNIF dadurch, dass man zunächst die Pluralendung ES von dem Wort abtrennt und dann den Buchstaben V in ein F umwandelt. Diese Umwandlung erfolgt durch Veränderung einer Wortkondition (Transformation), wobei die Wortkondition jedoch nicht notwendig identisch ist mit derjenigen Wortkondition, die zur Bestimmung der Endungskette definiert wurde.

Beispiel:

Wort:	w = KNIVES
Zerlegung:	w = KNIV ‖ ± ‖ ES
Wortendung:	ES
Wortkondition:	±
Zerlegung:	w = KNI ‖ V ‖ ES
Wortendung:	ES
Wortkondition:	V
Transformation:	V → F

Im Hinblick auf eine Transformation betrachten wir die Zerlegungen des Wortes w mit der festen Endungskette e(w):

$$w = r'_{m'}(w) \parallel c'_{k_r, m'}(w) \parallel e(w) \qquad (m' \text{ variabel})$$

(Die Länge von Restwort plus Wortkondition ist k_r.)

An diese Zerlegungen werden weitere Bedingungen geknüpft, die eine von ihnen eindeutig auszeichnen und es dadurch ermöglichen, den Wortstamm von w eindeutig zu definieren.

Analog zu B_4, B_5 und B_2 in 2.3 werden die Bedingungen B_7, B_8 und B_9 für die Zulässigkeit einer solchen Zerlegung definiert. (Die zu B_1 und B_3 analogen Bedingungen entfallen, da sich diese auf die Endungskette e(w)

beziehen; e(w) ist hier jedoch fest bestimmt.)

Eine endliche Menge H von Paaren $(C_j', t(C_j'))$ mit $C_j' \in C$ wird vorgegeben. Das Paar $(\pm, \pm)$ sei in H enthalten.

Bedeutung: Hat ein Wort eine Wortkondition aus der Menge C_j', so soll sie durch die Zeichenkette $t(C_j')$ ersetzt werden. (Bei Anwendung des Paares $(\pm, \pm)$ wird also keine Transformation durchgeführt).

Die $t(C_j')$ sind Zeichenketten:

$t_j = t(C_j') := \prod_{i=1}^{p_j} t^i$
$l(t_j) = p_j$

Alle t_j bilden die Menge T.

Unter folgenden Bedingungen ist die Wortzerlegung zulässig:

$B_7 := c_m'(w) \in C_j' \wedge C_j' \in C$
$B_8 := (C_j', t(C_j')) \in H$
$B_9 := [k \leq q \Rightarrow p_j \geq k-m'] \wedge [k > q \Rightarrow p_j \geq q-m']$

In der Menge H führen wir nun wieder eine strikte Totalordnung ein $(H, \prec)$. Das Paar $(\pm, \pm)$ wird als das "grösste" Element der Menge definiert.

Analog zu B_6 in 2.4 wird mit der folgenden Bedingung B_{10} eine der zulässigen Zerlegungen eindeutig ausgezeichnet:

B_{10} := Falls mehrere zulässige Zerlegungen

$$w = r'_{m_i'}(w) \parallel c'_{k_r, m_i'}(w) \parallel e(w) \qquad (i=2,3,..)$$

existieren,

so wird diejenige Zerlegung ausgezeichnet, die dem "kleinsten" Paar $(C,t) \in H$ (im Sinne der Ordnung $\prec$) entspricht.

Damit sind eine Wortzerlegung und eine Transformationsvorschrift eindeutig bestimmt.

Nach Ausführung der Transformation ergibt sich die Zeichenkette

$$w = r'_{m'}(w) \parallel t_{m'} \parallel e(w)$$

und wir können damit eindeutig den Wortstamm s(w) definieren.

Wortstamm: Der Stamm s(w) eines Wortes w ist eine Zeichenkette, die - wie folgt - definiert wird:

$$s(w) := r'_{m'}(w) \parallel t_{m'} \qquad r'_{m'}(w) = r'_{m'}(c'_{m'}(e(w)))$$

2.7 Bestimmung der Wortart

Die Art eines Wortes (Substantiv, Verb usw.) kann in vielen Fällen aufgrund einer typischen Wortendung bestimmt werden.

Beispiel: MEASUREMENT

Endung: MENT ; Stamm: MEASURE

Wortart: Substantiv

d.h.: MENT ist eine typische Endung englischer Substantive.

Die Art eines Wortes w sei mit a(w) bezeichnet.

Wortarten sind zum Beispiel:

Substantiv
Verb
Adjektiv
Pronomen
Zahlwort
usw.

Kombinationen von Wortarten bezeichnen wir wieder als Wortarten.

Beispiel:

Substantiv/Verb

Zusätzlich definieren wir eine "unbestimmte Wortart".

Alle Wortarten a_i werden zusammengefasst in der Menge A.

Wir definieren nun eine rechtseindeutige Relation R_1, deren Vorbereich mit G^1 übereinstimmt und deren Nachbereich die Menge A ist:

$$R_1 \subseteq G^1 \times A$$

D sei die Menge der geordneten Paare, welche die Relation R_1 erfüllen:

$$D := \left\{ ((C_i, e_j), a_k) \;/\; (C_i, e_j)\, R_1\, a_k \right\}$$

Aus der Rechtseindeutigkeit der Relation R_1 folgt die Gleichmächtigkeit der Mengen D und G^1:

$$|D| = |G^1|$$

Betrachten wir nun ein Wort mit seiner eindeutigen zulässigen Zerlegung

$$w = w_1 = r_{m_r}(w_r) \parallel c_{k_r, m_r}(w_r) \parallel e_{k_r}(w_r) \parallel \ldots \parallel e_{k_1}(w_1) ,$$

so wird die Wortart von w folgendermassen definiert:

$$a(w) = a_k \Longleftrightarrow ((C(e_{k_1}(w_1)), e_{k_1}(w_1)), a_k) \in D$$

2. 8 Der Wortstamm als Repräsentant einer Wortklasse

W sei die Menge aller Wörter w, S die Menge aller Stämme s(w). Die Abbildung

$$f : W \longrightarrow S$$

ist eindeutig.

Wir erweitern den Definitionsbereich der Funktion f um die Menge S. Die Elemente $s \in S$ werden durch die Funktion f auf sich selbst abgebildet.

Die Vereinigungsmenge von W und S (erweiterter Definitionsbereich der Funktion f) sei die Menge V:

$$V = W \cup S$$

Auf der Menge V wird die Relation R_2 definiert ($R_2 \subseteq V \times V$):

$$v_1 \; R_2 \; v_2 \Leftrightarrow s_1 = f(v_1) \wedge s_2 = f(v_2) \wedge s_1 = s_2$$

Diese Relation ist reflexiv, symmetrisch und transitiv, d.h. eine Äquivalenzrelation (RST-Relation).

Die Wörter $v_1, v_2 \in V$ sind also genau dann äquivalent, wenn ihre Stämme s_1 und s_2 gleich sind.

Durch R_2 wird die Menge V in disjunkte Klassen eingeteilt [1].

$$V = \bigcup_{i=1}^{n} [v_i]$$

$[v_i]$ ist die Klasse aller Elemente v, die zu v_i äquivalent sind. Der Stamm s_i gehört zur Klasse $[v_i]$. Wir wählen s_i als Repräsentanten der Klasse.

---

1) Vgl. z. B. bei HORNFECK (9), Satz über Äquivalenzrelationen.

Bemerkung: Bei WENZEL (22) werden die Stämme $s_i \in S$ als Morpheme bezeichnet. Er zerlegt die Menge S weiter in zwei disjunkte Untermengen:

1. Menge F der Formeme

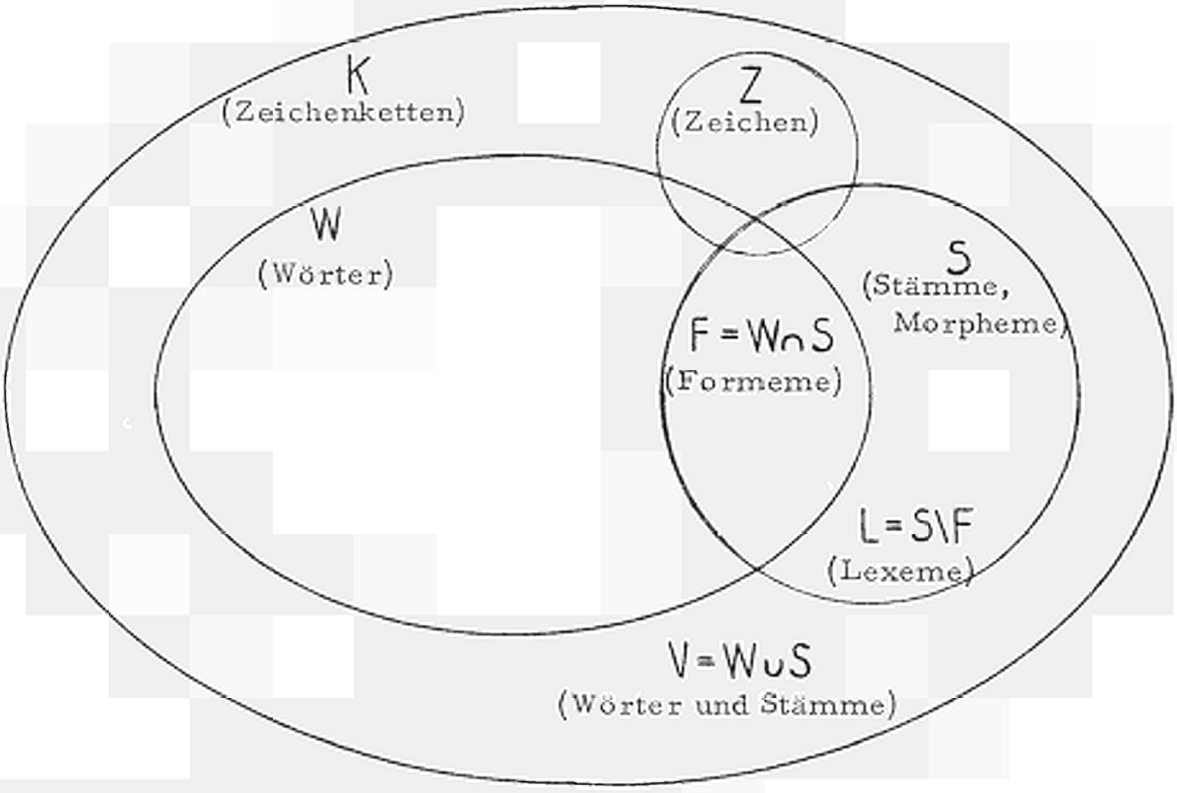
   Enthält eine Klasse $[s_i]$ nur das Element s_i, so bezeichnet er dieses mit Formem. Danach ist jedes Formem sein eigener Repräsentant.

2. Menge L der Lexeme

   Diejenigen Morpheme, die keine Formeme sind, nennt er Lexeme.

$$S = L \cup F \quad ; \quad L \cap S = \emptyset$$

Diese Bezeichnungen werden in der Linguistik jedoch nicht einheitlich verwendet.



K = Menge aller Zeichenketten
V = Menge aller Wörter und Stämme
W = Menge aller Wörter
S = Menge aller Stämme (Morpheme)
F = Menge aller Formeme
L = Menge aller Lexeme
Z = Menge aller Zeichen

3 Erstellung von Suffixlisten

In 2.3 wurde eine Menge E von Suffixen vorausgesetzt. Zur Erstellung einer solchen Suffixliste sind zwei Ansätze möglich:

1. Morphologischer Ansatz

   In vielen Grammatiken sind Zusammenstellungen von Suffixen einer Sprache enthalten, die aufgrund morphologischer Untersuchungen definiert sind (13 a) und die für eine automatische Suffixanalyse übernommen werden können.

2. Statistischer Ansatz

   In vorhandenen oder zu erstellenden rückläufigen Wörterbüchern (11, 14) sind alle Wörter, beginnend mit dem letzten Buchstaben eines Wortes, alphabetisch geordnet, sodass Wörter mit gleichen Endungen hintereinander aufgeführt sind. Falls eine grössere Zahl von Wörtern mit einer gleichen Endung existiert, kann man diese Wortendung als Suffix definieren. Eine solche Endung braucht jedoch kein Suffix im grammatikalischen Sinne zu sein.

   Falls das rückläufige Wörterbuch für jedes Wort auch noch dessen Wortart enthält, können mit diesem Ansatz speziell solche Suffixe gefunden werden, die charakteristisch sind für eine bestimmte Wortart.

In der Literatur über automatische Textverarbeitung finden sich mehr oder weniger umfangreiche Suffixlisten, in den meisten Fällen erstellt mit Hilfe einer Kombination der beiden obigen Methoden (1, 3, 4, 5, 12, 17, 18).

Ein automatischer Algorithmus zur Erstellung eines Regelsatzes, der sowohl eine Wortstammbildung als auch eine möglichst gute Wortartbestimmung ermöglichen soll sowie ein Anwendungsbeispiel, finden sich in Anhang I.

3.1 Spezielle Regelsätze

Im folgenden werden drei Regelsätze beschrieben, und zwar mit der in 2 eingeführten Notation. Dazu müssen jeweils folgende Mengen und Parameter definiert werden:

1. Menge E^i — Suffixe für den i-ten Iterationsschritt (siehe 2.3, Bedingung B_1)
2. Menge C^i — Konditionen für den i-ten Iterationsschritt
   a) Mengen, die genau eine Zeichenkette aus K enthalten
   b) Komplementmengen von Zeichenketten bezüglich K
   c) Mengen, die durch Vereinigung von Mengen der Arten a) und b) gebildet werden sowie deren Komplementmengen bezüglich K

   (siehe 2.3, Bedingung B_4 und 2.6, Bedingung B_7)
3. Menge T — Zeichenketten, in die eine Kondition transformiert werden kann (siehe 2.6)
4. Menge G^i — Suffixe mit zugeordneten Konditionen für den i-ten Iterationsschritt (siehe 2.3, Bedingungen B_5, B_6)
5. Menge H — Transformationen (siehe 2.6, Bedingungen B_8, B_{10})
6. Menge Q — Suffixe, für die eine spezielle Mindestlänge (des entstehenden Wortstammes) gefordert wird (siehe 2.3, Bedingung B_3)
7. Parameter q — Generelle Mindestlänge für Wortstämme (siehe 2.3, Bedingung B_2 und 2.6, Bedingung B_9)
8. Menge A — Wortarten (siehe 2.7)
9. Menge D — Elemente der Menge G^1 mit zugeordneten Wortarten (siehe 2.7)

3.1.1 EURATOM-Regeln

Von H. FANGMEYER und G. RAMBS (EURATOM (Cetis), Ispra) wurde 1967 ein Regelsatz erstellt, der in einem vollautomatischen experimentellen Indexierungssystem angewandt wurde (6, 7). Der dabei benutzte Algorithmus (Programm SUFFANAL, siehe Anhang V) ist rekursiv. Die Regeln ermöglichen Wortstammbildung und Wortartbestimmung.

1. Menge E^1 (159 Elemente)

AB	IZE	IER	ERES	AT
IB	E	LER	URES	IT
AC	IF	ER	NESSES	ANT
LIC	AG	IR	ES	MENT
TRIC	LING	OR	LINGS	ENT
RIC	ING	OUR	INGS	ERT
ISTIC	ISH	UR	IS	EST
IC	ARI	'S	ELS	IST
ANC	TRECAL	ISTICS	ITHMS	UT
ENC	TRICAL	ICS	ISMS	YT
IED	RICAL	OIDS	IUMS	ITU
LED	ICAL	UIDS	UMS	IV
ED	ISL	IDS	PTIONS	ROW
OID	RAL	DS	IONS	EX
UID	PTUAL	ANCES	ONS	IX
ID	UAL	ENCES	SHIPS	ACY
D	AL	AGES	LERS	ANCY
ANCE	EL	ACIES	ERS	ENCY
ENCE	ABIL	ANCIES	ORS	LABLY
AGE	FUL	ENCIES	OURS	ABLY
LIKE	ITHM	LIES	LESS	LY
LABLE	ISM	ARIES	NESS	ARY
ABLE	IUM	ERIES	ITS	ERY
IBLE	UM	ORIES	ANTS	ORY
ICLE	IAN	ITRIES	MENTS	ITRY
LE	PTION	TRIES	ENTS	TRY
ENE	ION	ITIES	PTS	ITY
ERE	ON	IES	ISTS	Y
URE	SHIP	ICLES	OUS	IZ
IQUE	IP	LES	S	S'
QUE	LAR	ENES	IAT	±
IVE	AR	ARES	LAT	

Menge E^i $(1 < i \leq 5)$ (79 Elemente)

AB	AGE	TRICAL	IR	ERT
IB	LIKE	RICAL	OUR	UT
AC	LABLE	RAL	UR	YT
LIC	ABLE	UAL	LIES	ITU
TRIC	IBLE	AL	IES	IV
RIC	LE	ABIL	ES	ROW
ISTIC	IQUE	FUL	IS	ACY
IC	QUE	ISM	LESS	ARY
ANC	E	IUM	OUS	ORY
ENC	IF	ION	S	ITRY
IED	AG	ON	IAT	TRY
LED	LING	IP	LAT	ITY
ED	ING	LAR	AT	Y
ID	ISH	AR	IT	IZ
ANCE	ARI	IER	ANT	±
ENCE	TRECAL	ER	ENT	

2. Menge C^1 (15 Elemente)

a)
U
L
XIONS
±

b)
¬A
¬E
¬XI
¬V
¬X

c)
1 := BvCvDvFvGvHvKvLvMvNvPvRvSvTvVvWvXvZ
2 := ANvOR
3 := ACvICvADvEDvUD
4 := MvN
5 := SvZ
6 := SvT

Menge C^i $(1 < i \leqslant 5)$ (5 Elemente)

a)
XIONS
±

b)
¬E

c)
1 := Bv Cv Dv Fv Gv Hv Kv Lv Mv Nv Pv Rv Sv Tv Vv Wv Xv Z
6 := Sv T

3. Menge T (2 Elemente)

CT
±

4. Menge G^1 (164 Elemente)

In der Menge G^1 ist eine strikte Totalordnung definiert, die der Reihenfolge der Elemente in der folgenden Liste entspricht (zuerst Spalten, dann Zeilen).

Beispiel: (±, ISH) ≺ (±, ARI) ≺ (±, TRECAL)

(±, AB)	(±, ARI)	(±, ANCES)	(±, ITS)
(±, IB)	(±, TRECAL)	(±, ENCES)	(±, ANTS)
(±, AC)	(±, TRICAL)	(3, ES)	(±, MENTS)
(1, LIC)	(±, RICAL)	(±, AGES)	(±, ENTS)
(±, TRIC)	(±, ICAL)	(±, ACIES)	(±, PTS)
(±, RIC)	(±, IAL)	(±, ANCIES)	(L, ISTS)
(±, ISTIC)	(4, AL)	(±, ENCIES)	(±, OUS)
(±, IC)	(±, RAL)	(±, LIES)	(±, S)
(±, ANC)	(±, PTUAL)	(±, ARIES)	(±, IAT)
(±, ENC)	(±, UAL)	(±, ERIES)	(1, LAT)
(±, IED)	(±, AL)	(±, ORIES)	(±, AT)
(1, LED)	(±, EL)	(±, ITRIES)	(±, IT)
(¬E, ED)	(±, ABIL)	(±, TRIES)	(±, ANT)
(¬V, OID)	(±, FUL)	(±, ITIES)	(±, MENT)
(±, UID)	(±, ITHM)	(±, IES)	(±, ENT)
(U, ID)	(±, ISM)	(±, ICLES)	(±, ERT)
(±, ID)	(±, IUM)	(±, LES)	(±, EST)
(2, D)	(±, UM)	(±, ENES)	(L, IST)
(±, ANCE)	(±, IAN)	(±, ARES)	(±, UT)
(±, ENCE)	(±, PTION)	(±, ERES)	(±, YT)
(3, E)	(±, ION)	(±, URES)	(±, ITU)
(±, AGE)	(±, ON)	(±, NESSES)	(6, IV)
(±, LIKE)	(±, SHIP)	(±, ES)	(±, ROW)
(1, LABLE)	(±, IP)	(1, LINGS)	(±, EX)
(±, ABLE)	(1, LAR)	(±, INGS)	(±, IX)
(±, IBLE)	(±, AR)	(±, IS)	(±, ACY)
(±, ICLE)	(±, IER)	(±, ELS)	(±, ANCY)
(1, LE)	(1, LER)	(±, ITHMS)	(±, ENCY)
(±, ENE)	(5, ER)	(±, ISMS)	(1, LABLY)
(±, ERE)	(±, ER)	(±, IUMS)	(±, ABLY)
(±, URE)	(±, IR)	(±, UMS)	(¬A, LY)
(±, IQUE)	(±, OR)	(±, PTIONS)	(±, ARY)
(±, QUE)	(±, OUR)	(¬X, IONS)	(±, ERY)
(±, IVE)	(±, UR)	(¬XI, ONS)	(±, ORY)
(±, IZE)	(±, 'S)	(±, SHIPS)	(±, ITRY)
(±, E)	(±, ISTICS)	(1, LERS)	(±, TRY)
(±, IF)	(±, ICS)	(±, ERS)	(±, ITY)
(±, AG)	(¬V, OIDS)	(±, ORS)	(±, Y)
(1, LING)	(±, UIDS)	(±, OURS)	(±, IZ)
(±, ING)	(U, IDS)	(±, LESS)	(±, S')
(±, ISH)	(2, DS)	(±, NESS)	(±, ±)

Menge G^i $(1 < i \leq 5)$ (79 Elemente)

In der Menge G^i ist eine strikte Totalordnung definiert, die der Reihenfolge der Elemente in der folgenden Liste entspricht (zuerst Spalten, dann Zeilen).

(±, AB)	(±, IBLE)	(±, IUM)	(±, AT)
(±, IB)	(1, LE)	(±, ION)	(±, IT)
(±, AC)	(±, IQUE)	(±, ON)	(±, ANT)
(1, LIC)	(±, QUE)	(±, IP)	(±, ENT)
(±, TRIC)	(±, E)	(1, LAR)	(±, ERT)
(±, RIC)	(±, IF)	(±, AR)	(±, UT)
(±, ISTIC)	(±, AG)	(±, IER)	(±, YT)
(±, IC)	(1, LING)	(±, ER)	(±, ITU)
(±, ANC)	(±, ING)	(±, IR)	(6, IV)
(±, ENC)	(±, ISH)	(±, OUR)	(±, ROW)
(±, IED)	(±, ARI)	(±, UR)	(±, ACY)
(1, LED)	(±, TRECAL)	(±, LIES)	(±, ARY)
(¬E, ED)	(±, TRICAL)	(±, IES)	(±, ORY)
(±, ID)	(±, RICAL)	(±, ES)	(±, ITRY)
(±, ANCE)	(±, RAL)	(±, IS)	(±, TRY)
(±, ENCE)	(±, UAL)	(±, LESS)	(±, ITY)
(±, AGE)	(±, AL)	(±, OUS)	(±, Y)
(±, LIKE)	(±, ABIL)	(±, S)	(±, IZ)
(1, LABLE)	(±, FUL)	(±, IAT)	(±, ±)
(±, ABLE)	(±, ISM)	(1, LAT)	

5. Menge H (2 Elemente)

(XIONS, CT)
(± , ±)

Ordnung: (XIONS, CT) $\prec$ (± , ±)

6. Menge Q (0 Elemente)

$Q = \emptyset$

7. Parameter q

$q = 4$

8. Menge A (4 Elemente)

N (= Substantiv)
V (= Verb)
A (= Adjektiv oder Adverb)
0 (= unbestimmt)

9. Menge D (164 Elemente)

```
(( ¬V,   OID), N) ((*  ,ITRIES), N) ((*  ,  ITRY), N) ((±  ,  AB), O)
((±  ,   UID), N) ((±  , TRIES), N) (( ±,   ITY), N) ((±  ,  IB), O)
((  U,    ID), N) ((±  , ITIES), N) (( ±,___ S'), N) ((±  ,  AC), O)
((  2,     D), N) ((±  , ICLES), N)                  ((±  , ANC), O)
((±  ,  ANCE), N) ((±  ,   LES), N) (( 1,   LED), V) ((±  , ENC), O)
((±  ,  ENCE), N) ((±  ,  ENES), N) ((¬E,    ED), V) ((±  , IED), O)
((  3,     E), N) ((±  ,  ARES), N) (( ±,   IZE), V) ((±  ,  ID), O)
((±  ,   AGE), N) ((±  ,  ERES), N) (( 1,  LING), V) (( 1,  LE), O)
((±  ,  ICLE), N) ((±  ,  URES), N) (( ±,   ING), V) ((±  , QUE), O)
((±  ,   ENE), N) ((±  ,NESSES), N)                  ((±  ,   E), O)
(( ±,   ERE), N) (( 1, LINGS), N)                    ((±  ,  IF), O)
(( ±,   URE), N) ((±  ,  INGS), N) (( 1,   LIC), A) ((±  ,  AG), O)
(( ±,    EL), N) ((±  ,   ELS), N) (( ±,  TRIC), A) ((±  , ISH), O)
(( ±,  ITHM), N) ((±  , ITHMS), N) (( ±,   RIC), A) ((±  , ARI), O)
(( ±,   ISM), N) ((±  ,  ISMS), N) (( ±, ISTIC), A) ((±  , RAL), O)
(( ±,   IUM), N) ((±  ,  IUMS), N) (( ±,    IC), A) ((±  ,  AL), O)
(( ±,    UM), N) ((±  ,   UMS), N) (( ±,  LIKE), A) ((±  ,ABIL), O)
(( ±, PTION), N) ((±  ,PTIONS), N) (( 1, LABLE), A) ((±  ,  IP), O)
(( ±,   ION), N) ((¬X,  IONS), N) (( ±,  ABLE), A) ((±  ,  AR), O)
(( ±,    ON), N) ((¬XI,   ONS), N) (( ±,  IBLE), A) ((±  , IER), O)
(( ±,  SHIP), N) ((±  , SHIPS), N) (( ±,  IQUE), A) ((±  ,  ER), O)
((  1,  LER), N) (( 1,  LERS), N) (( ±,   IVE), A) ((±  ,  IR), O)
((  5,   ER), N) ((±  ,   ERS), N) (( ±,TRECAL), A) ((±  ,  OR), O)
(( ±,   OUR), N) ((±  ,   ORS), N) (( ±,TRICAL), A) ((±  ,  UR), O)
(( ±,    'S), N) ((±  ,  OURS), N) (( ±, RICAL), A) ((±  ,LIES), O)
(( ±,ISTICS), N) ((±  ,  NESS), N) (( ±,  ICAL), A) ((±  , IES), O)
(( ±,   ICS), N) ((±  ,   ITS), N) (( ±,   IAL), A) ((±  ,  ES), O)
(( ¬V,  OIDS), N) ((±  ,  ANTS), N) (( 4,    AL), A) ((±  ,  IS), O)
(( ±,   UIDS), N) ((±  , MENTS), N) (( ±, PTUAL), A) ((±  ,   S), O)
(( *U,   IDS), N) ((*  ,  ENTS), N) (( ±,   UAL), A) ((±  , IAT), O)
((  2,    DS), N) ((±  ,   PTS), N) (( ±,   FUL), A) (( 1, LAT), O)
(( ±, ANCES), N) (( L,  ISTS), N) (( ±,   IAN), A) ((±  ,  AT), O)
(( ±, ENCES), N) ((±  ,  MENT), N) (( 1,   LAR), A) ((±  ,  IT), O)
((  3,    ES), N) (( L,   IST), N) (( ±,  LESS), A) ((±  , ANT), O)
(( ±,   AGES), N) ((±  ,    EX), N) (( ±,   OUS), A) ((±  , ENT), O)
(( ±, ACIES), N) ((±  ,    IX), N) (( ±,   EST), A) ((±  , ERT), O)
(( ±,ANCIES), N) ((±  ,   ACY), N) (( 1, LABLY), A) ((±  ,  UT), O)
(( ±,ENCIES), N) ((±  ,  ANCY), N) (( ±,  ABLY), A) ((±  ,  YT), O)
(( ±, ARIES), N) ((±  ,  ENCY), N) ((¬A,    LY), A) ((±  , ITU), O)
(( ±, ERIES), N) ((±  ,   ERY), N) (( ±,   ARY), A) (( 6,  IV), O)
(( ±, ORIES), N) ((±  ,   ORY), N)                   ((±  , ROW), O)
                                                     ((±  , TRY), O)
                                                     ((±  ,   Y), O)
                                                     ((±  ,  IZ), O)
                                                     ((*  ,   ±), O)
```

3.1.2 SM-Regeln

Die folgende Suffixliste [1] wird zusammen mit einem vorgegebenen Stammwörterbuch im SMART Retrieval System (19, 20) angewandt. Sie dient also nicht zur Wortstammbildung selbst.

Es wird hier dennoch versucht, auch mit dieser Suffixliste eine automatische Stammbildung durchzuführen, d.h. die Suffixe werden als Regeln für einen Stammbildungsalgorithmus interpretiert.

Es wurden Versuche mit und ohne Iteration und mit den Mindestlängen 0 und 2 (für den entstehenden Stamm) durchgeführt. Die besten Ergebnisse ergaben sich bei einer Mindestlänge von 2 und iterativer Anwendung der Regeln.

Die so definierten Regeln werden hier mit SM-Regeln bezeichnet.

---

[1] Persönliche Mitteilung von Prof. G. SALTON

1. Menge E^i (201 Elemente) (i = 1, 2, 3)

IA	**IZING**	AGES	IZERS	ORILY
A	YZING	ACIES	YZERS	ISTICALLY
FIC	ING	ENCIES	ERS	YTICALLY
ISTIC	ISH	FIES	ATORS	ICALLY
YTIC	TH	OLOGIES	ORS	ALLY
IC	YTICAL	LOGIES	LESS	FULLY
IFIED	ICAL	ARIES	IVENESS	ARLY
FIED	ENTIAL	ERIES	NESS	LESSLY
ENED	MENTAL	ORIES	ANTS	EOUSLY
YSED	AL	EITIES	MENTS	**ITIOUSLY**
ATED	FUL	ABILITIES	ENTS	IOUSLY
IZED	ISM	IBILITIES	ISTS	OUSLY
YZED	UM	IVITIES	YSTS	ENTLY
ED	IAN	ITIES	EOUS	LY
FOLD	AN	TIES	ACIOUS	ARY
HOOD	EN	IES	ITIOUS	ERY
AE	IFICATION	AIRES	IOUS	ORY
ICE	FICATION	URES	OUS	RY
ANCE	IZATION	YSES	S	EITY
ENCE	ATION	ATES	ANT	ABILITY
TUDE	ITION	IZES	MENT	IBILITY
AGE	ION	YZES	ENT	IVITY
LIKE	ISON	ES	EST	ITY
ABLE	ON	INGS	IST	TY
IBLE	SHIP	THS	YST	Y
SOME	AR	YSIS	ACY	S'
INE	FIER	ITIS	ENCY	0
URE	IZER	ALS	FY	1
WISE	YZER	ISMS	OLOGY	2
YSE	ER	IANS	LOGY	3
ATE	ATOR	ANS	ABLY	4
ITE	OR	ENS	IBLY	5
ATIVE	'S	FICATIONS	FIEDLY	6
ITIVE	AS	IZATIONS	EDLY	7
IVE	ISTICS	ATIONS	SOMELY	8
IZE	ZTICS	ITIONS	ATELY	9
YZE	ICS	IONS	ITELY	±
ENING	HOODS	ISONS	ITIVELY	
YSING	ANCES	ONS	IVELY	
ATING	ENCES	ARS	ISHLY	
FYING	TUDES	FIERS	ARILY	

2. Menge C^i (1 Element)

±

3. Menge T (1 Element)

±

4. Menge G^i (201 Elemente)

$$G^i = \left\{ (\pm, e_j) \,/\, \pm \in C^i \wedge e_j \in E^i \wedge \bigvee_{e_j} \exists_{(\pm, e_j)} \right\}$$

Da die Menge C^i nur das Element $\pm$ enthält, ist die Menge G^i gleichmächtig mit der Menge E^i ($|G^i| = |E^i|$).
In der Menge G^i ist eine strikte Totalordnung definiert.
Die Ordnung entspricht der Reihenfolge der Elemente $e_i \in E^i$ (zuerst Spalten, dann Zeilen).

5. Menge H (1 Element)

$(\pm, \pm)$

6. Menge Q (0 Elemente)

$Q = \emptyset$

7. Parameter q

$q = 2$

8. Menge A (1 Element)

0 (= unbestimmt)

9. Menge D (201 Elemente)

$$D = \left\{ ((\pm, e_j), 0) \,/\, (\pm, e_j) \in G^1 \wedge 0 \in A \wedge \bigvee_{(\pm, e_j)} \exists_{((\pm, e_j), 0)} \right\}$$

Die Menge A enthält nur das Element 0, daraus folgt die Gleichmächtigkeit der Mengen G^1 und D:

$|G^1| = |D|$

3.1.3 LOVINS-Regeln (12)

1. Menge E[1] (295 Elemente)

IA	ATIVE	ARISATION	LESS	ATELY
ATA	IVE	ISATION	EABLENESS	ATIVELY
A	ICALIZE	ENTATION	ABLENESS	IVELY
AIC	ENTIALIZE	ARIZATION	IBLENESS	ELY
ALLIC	IALIZE	IZATION	ATENESS	ATINGLY
ARIC	IONALIZE	ATION	ITENESS	INGLY
ATIC	ALIZE	ACTION	ATIVENESS	LILY
ITIC	ARIZE	ION	IVENESS	ARILY
ANTIC	IZE	ON	ENESS	ILY
ALISTIC	E	O	INGNESS	AICALLY
ARISTIC	ANCING	EAR	ISHNESS	ALLICALLY
IVISTIC	ENCING	AR	ARINESS	ALISTICALLY
ISTIC	AGING	IER	INESS	ISTICALLY
IC	ENING	ARISER	ICALNESS	ICALLY
ANCED	IONING	ARIZER	ANTIALNESS	OIDALLY
ENCED	ATING	IZER	ENTIALNESS	ENTIALLY
ISHED	ENTING	ATOR	IONALNESS	IALLY
IED	YING	OR	ALNESS	IZATIONALLY
ENED	ARIZING	AS	FULNESS	ATIONALLY
IONED	IZING	ISTICS	LESSNESS	IONALLY
ATED	ING	ICS	EOUSNESS	ENTALLY
ENTED	YISH	ANCES	IOUSNESS	ALLY
ARIZED	ISH	ENCES	ITOUSNESS	EFULLY
IZED	I	OIDES	OUSNESS	IFULLY
ED	AICAL	IDES	ENTNESS	FULLY
ARIOD	ISTICAL	AGES	NESS	ENLY
IOD	ICAL	ACIES	ANTS	EARLY
EHOOD	OIDAL	ANCIES	ICISTS	ARLY
ELIHOOD	EAL	ENCIES	ISTS	LESSLY
IHOOD	ANCIAL	ARIES	ACEOUS	EOUSLY
HOOD	ARIAL	ALITIES	ANTANEOUS	IOUSLY
WARD	ENTIAL	IVITIES	EOUS	OUSLY
AE	IAL	ITIES	ACIOUS	ENTLY
ICANCE	IZATIONAL	IES	IOUS	LY
ANCE	ATIONAL	INES	ITOUS	ARY
ENCE	IONAL	NESSES	OUS	ERY
ICIDE	ENTAL	ATES	US	ICIANRY
OTIDE	AL	ATIVES	'S	ATORY
IDE	EFUL	ES	S	ACITY
AGE	IFUL	INGS	ICANT	ICITY
ATABLE	FUL	IS	ANT	EITY
ARIZABLE	YL	ENTIALS	IZEMENT	ICALITY
IZABLE	ICISM	IALS	EMENT	ANTIALITY
ABLE	OIDISM	IONALS	ENT	ENTIALITY
ENCIBLE	ICALISM	ALS	ICIST	IALITY
IBLE	IONALISM	ISMS	ICALIST	IONALITY
ENE	ALISM	ICIANS	IALIST	ALITY
IDINE	INISM	IANS	ALIST	ELITY
INE	ATIVISM	ARISATIONS	IONIST	ARIZABILITY
ONE	ISM	ENTATIONS	ENTIST	IZABILITY
EATURE	IUM	ARIZATIONS	IST	ABILITY
ATURE	UM	IZATIONS	ACY	IBILITY
ESE	ICIAN	ATIONS	ANCY	INITY
WISE	IAN	IONS	ENCY	ARITY
ENTIATE	OGEN	ARS	EALY	IVITY
INATE	EN	IERS	ABLY	ITY
IONATE	ICATION	IZERS	IBLY	Y
ATE	ENTIATION	ATORS	IEDLY	S'
ITE	INATION	ELESS	EDLY	±

Menge E^2 (1 Element)

$\pm$

Menge E^i $(i > 2)$ (0 Elemente)

$E^i = \emptyset$

2. Menge C^1 (32 Elemente)

a) C
F
L
IN
$\pm$

b) $\neg$C
$\neg$E
$\neg$F

c) 1 := T $\vee$ LL
2 := 0 $\vee$ E
3 := A $\vee$ E
4 := L $\vee$ I $\vee$ UδE 1)
5 := U $\vee$ X $\vee$ S $\wedge \neg$OS
6 := A $\vee$ C $\vee$ E $\vee$ M
7 := δS$\delta\delta$ $\vee$ $\neg$S$\delta\delta$ 1)
8 := L $\vee$ I
9 := L $\vee$ N
10 := N $\vee$ R
11 := DR $\vee$ T $\wedge \neg$TT
12 := S $\vee$ T $\wedge \neg$OT
13 := L $\vee$ M $\vee$ N $\vee$ R
14 := S $\vee$ U
15 := L $\vee$ I $\vee$ UδE 1)
16 := D $\vee$ F $\vee$ PH $\vee$ TH $\vee$ L $\vee$ ER $\vee$ OR $\vee$ ES $\vee$ T
17 := MET $\vee$ RYST

$\neg$2 = $\neg$0 $\wedge \neg$E
$\neg$3 = $\neg$A $\wedge \neg$E
$\neg$5 = $\neg$U $\wedge \neg$X $\wedge \neg$S $\vee$ OS
$\neg$6 = $\neg$A $\wedge \neg$C $\wedge \neg$E $\wedge \neg$M
$\neg$9 = $\neg$L $\wedge \neg$N
$\neg$14 = $\neg$S $\wedge \neg$U
$\neg$17 = $\neg$MET $\wedge \neg$RYST

---

1) δ steht für ein beliebiges Zeichen aus Z.

Menge C^2 (45 Elemente)

a)

BB	OND	RR	RPT	DEX
UAD	LUD	METR	ERT	PEX
VAD	RUD	ISTR	TT	TEX
DD	GG	URS	YT	EX
CID	LL	SS	IEV	IX
LID	MM	UCT	OLV	LUX
ERID	NN	MIT	AX	YZ
PAND	PP	UMPT	BEX	±

c)

18 := UL ∧¬(AUL ∨ IUL ∨ OUL)
19 := END ∧¬SEND
20 := HER ∧¬(PHER ∨ THER)
21 := ENT ∧¬MENT
22 := ET ∧¬NET

Mengen C^i (i > 2) (0 Elemente)

$C^i = \emptyset$

3. Menge T (42 Elemente)

RB	LUC	P	CIS	RUS
B	UC	METER	LIS	YS
AC	D	ISTER	MIS	S
EC	IEF	UR	ERIS	OLUT
BIC	G	R	PANS	T
DIC	L	UAS	ENS	±
PIC	UM	VAS	ONS	
TIC	M	HES	ERS	
IC	N	ES	LUS	

4. Menge G^1 (195 Elemente)

In der Menge G^1 ist eine strikte Totalordnung definiert, die der Anordnung der Elemente in der folgenden Liste entspricht. (Die Elemente e_i der Paare (c_i, e_i) sind rückläufig alphabetisch angeordnet.)

(±,	IA)	(±,	INATE)	(13,	UM)
(±,	ATA)	(±,	IONATE)	(±,	ICIAN)
(±,	A)	(±,	ATE)	(±,	IAN)
(±,	AIC)	(16,	ITE)	(±,	OGEN)
(¬17,	ALLIC)	(±,	ATIVE)	(¬E,	EN)
(±,	ARIC)	(±,	IVE)	(F,	ICATION)
(±,	ATIC)	(±,	ICALIZE)	(±,	ENTIATION)
(1,	ITIC)	(±,	ENTIALIZE)	(±,	INATION)
(±,	ANTIC)	(±,	IALIZE)	(±,	ARISATION)
(±,	ALISTIC)	(±,	IONALIZE)	(±,	ISATION)
(±,	ARISTIC)	(±,	ALIZE)	(±,	ENTATION)
(±,	IVISTIC)	(±,	ARIZE)	(±,	ARIZATION)
(±,	ISTIC)	(¬E,	IZE)	(¬E,	IZATION)
(±,	IC)	(±,	E)	(±,	ATION)
(±,	ANCED)	(±,	ANCING)	(F,	ACTION)
(±,	ENCED)	(±,	ENCING)	(¬9,	ION)
(±,	ISHED)	(±,	AGING)	(11,	ON)
(±,	IED)	(¬E,	ENING)	(±,	O)
(¬E,	ENED)	(±,	IONING)	(IN,	EAR)
(±,	IONED)	(¬2,	ATING)	(15,	AR)
(¬2,	ATED)	(±,	ENTING)	(±,	IER)
(±,	ENTED)	(±,	YING)	(±,	ARISER)
(±,	ARIZED)	(±,	ARIZING)	(±,	ARIZER)
(¬E,	IZED)	(¬E,	IZING)	(¬E,	IZER)
(¬E,	ED)	(7,	ING)	(±,	ATOR)
(±,	AROID)	(±,	YISH)	(12,	OR)
(±,	OID)	(±,	ISH)	(±,	AS)
(±,	EHOOD)	(±,	I)	(±,	ISTICS)
(¬E,	ELIHOOD)	(±,	AICAL)	(±,	ICS)
(±,	IHOOD)	(±,	ISTICAL)	(±,	ANCES)
(±,	HOOD)	(±,	ICAL)	(±,	ENCES)
(±,	WARD)	(±,	OIDAL)	(±,	OIDES)
(±,	AE)	(IN,	EAL)	(¬5,	IDES)
(±,	ICANCE)	(±,	ANCIAL)	(±,	AGES)
(±,	ANCE)	(±,	ARIAL)	(±,	ACIES)
(±,	ENCE)	(±,	ENTIAL)	(±,	ANCIES)
(±,	ICIDE)	(±,	IAL)	(±,	ENCIES)
(±,	OTIDE)	(±,	IZATIONAL(	(±,	ARIES)
(¬5,	IDE)	(±,	ATIONAL)	(±,	ALITIES)
(±,	AGE)	(±,	IONAL)	(±,	IVITIES)
(±,	ATABLE)	(±,	ENTAL)	(±,	ITIES)
(±,	ARIZABLE)	(¬17,	AL)	(¬C,	IES)
(¬E,	IZABLE)	(±,	EFUL)	(¬6,	INES)
(±,	ABLE)	(±,	IFUL)	(±,	NESSES)
(±,	ENCIBLE)	(±,	FUL)	(±,	ATES)
(±,	IBLE)	(10,	YL)	(±,	ATIVES)
(¬E,	ENE)	(±,	ICISM)	(¬E,	ES)
(¬2,	IDINE)	(±,	OIDISM)	(7,	INGS)
(¬6,	INE)	(±,	ICALISM)	(±,	IS)
(10,	ONE)	(±,	IONALISM)	(±,	ENTIALS)
(¬F,	EATURE)	(±,	ALISM)	(±,	IALS)
(¬E,	ATURE)	(¬3,	INISM)	(±,	IONALS)
(±,	ESE)	(±,	ATIVISM)	(¬17,	ALS)
(±,	WISE)	(±,	ISM)	(±,	ISMS)
(±,	ENTIATE)	(±,	IUM)	(±,	ICIANS)

(±, IANS)	(±, ACIOUS)	(±, IZATIONALLY)
(±, ARISATIONS)	(±, IOUS)	(±, ATIONALLY)
(±, ENTATIONS)	(±, ITOUS)	(±, IONALLY)
(±, ARIZATIONS)	(±, OUS)	(±, ENTALLY)
(±, IZATIONS)	(C, US)	(±, ALLY)
(±, ATIONS)	(±, 'S)	(±, EFULLY)
(±, IONS)	(¬14, S)	(±, IFULLY)
(± 8, ARS)	(±, ICANT)	(±, FULLY)
(±, IERS)	(±, ANT)	(¬E, ENLY)
(¬E, IZERS)	(±, IZEMENT)	(IN, EARLY)
(±, ATORS)	(±, EMENT)	(4, ARLY)
(±, ELESS)	(±, ENT)	(±, LESSLY)
(±, LESS)	(±, ICIST)	(±, EOUSLY)
(¬E, EABLENESS)	(±, ICALIST)	(±, IOUSLY)
(±, ABLENESS)	(±, IALIST)	(±, OUSLY)
(±, IBLENESS)	(±, ALIST)	(±, ENTLY)
(±, ATENESS)	(±, IONIST)	(±, LY)
(±, ITENESS)	(±, ENTIST)	(¬E, ARY)
(±, ATIVENESS)	(±, IST)	(¬E, ERY)
(±, IVENESS)	(±, ACY)	(±, ICIANRY)
(¬E, ENESS)	(±, ANCY)	(±, ATORY)
(±, INGNESS)	(±, ENCY)	(±, ACITY)
(±, ISHNESS)	(IN, EALY)	(±, ICITY)
(¬E, ARINESS)	(±, ABLY)	(±, EITY)
(±, INESS)	(±, IBLY)	(±, ICALITY)
(±, ICALNESS)	(±, IEDLY)	(±, ANTIALITY)
(±, ANTIALNESS)	(¬E, EDLY)	(±, ENTIALITY)
(±, ENTIALNESS)	(±, ATELY)	(±, IALITY)
(±, IONALNESS)	(±, ATIVELY)	(±, IONALITY)
(±, ALNESS)	(±, IVELY)	(±, ALITY)
(±, FULNESS)	(¬E, ELY)	(±, ELITY)
(±, LESSNESS)	(±, ATINGLY)	(±, ARIZABILITY)
(±, EOUSNESS)	(±, INGLY)	(±, IZABILITY)
(±, IOUSNESS)	(±, LILY)	(±, ABILITY)
(±, ITOUSNESS)	(±, ARILY)	(±, IBILITY)
(±, OUSNESS)	(±, ILY)	(L, INITY)
(±, ENTNESS)	(±, AICALLY)	(±, ARITY)
(±, NESS)	(±, ALLICALLY)	(±, IVITY)
(±, ANTS)	(±, ALISTICALLY)	(±, ITY)
(±, ICISTS)	(±, ISTICALLY)	(±, Y)
(±, ISTS)	(±, ICALLY)	(±, S')
(±, ACEOUS)	(±, OIDALLY)	(±, ±)
(±, ANTANEOUS)	(±, ENTIALLY)	
(±, EOUS)	(±, IALLY)	

Menge G^2 (1 Element)

(±, ±)

Menge G^i $(i > 2)$ (0 Elemente)

$G^i = \emptyset$

5. Menge H (45 Elemente)

In der Menge H besteht eine strikte Totalordnung, die der Anordnung der Elemente in der folgenden Liste entspricht (zuerst Spalten, dann Zeilen).

(BB, B)	(UMPT, UM)	(EX, EC)	(RUD, RUS)
(DD, D)	(RPT, RB)	(IX, IC)	(20, HES)
(GG, G)	(URS, UR)	(LUX, LUC)	(MIT, MIS)
(LL, L)	(ISTR,ISTER)	(UAD, UAS)	(21, ENS)
(MM, M)	(METR,METER)	(VAD, VAS)	(ERT, ERS)
(NN, N)	(OLV, OLUT)	(CID, CIS)	(22, ES)
(PP, P)	(18, L)	(LID, LIS)	(YT, YS)
(RR, R)	(BEX, BIC)	(ERID, ERIS)	(YZ, YS)
(SS, S)	(DEX, DIC)	(PAND, PANS)	(±, ±)
(TT, T)	(PEX, PIC)	(19, ENS)	
(IEV, IEF)	(TEX, TIC)	(OND, ONS)	
(UCT, UC)	(AX, AC)	(LUD, LUS)	

6. Menge Q (51 Elemente)

(ALLIC, 3)	(AL, 3)	(IZERS, 3)
(ANTIC, 4)	(ALISM, 3)	(ANTS, 3)
(ALISTIC, 3)	(ISM, 3)	(ACIOUS, 3)
(ANCED, 3)	(EN, 3)	(ANT, 3)
(ENTED, 4)	(ICATION, 3)	(ENT, 4)
(IZED, 3)	(IZATION, 3)	(ANCY, 3)
(ANCE, 3)	(ATION, 3)	(INGLY, 3)
(AGE, 3)	(ACTION, 3)	(ALLICALLY, 4)
(IONATE, 5)	(ION, 3)	(ALISTICALLY, 3)
(IZE, 3)	(IZER, 3)	(IZATIONALLY, 3)
(ANCING, 3)	(AS, 3)	(ATIONALLY, 3)
(AGING, 3)	(ANCES, 3)	(ALLY, 3)
(ENTING, 4)	(AGES, 3)	(ARLY, 3)
(YING, 3)	(ALS, 3)	(LY, 3)
(IZING, 3)	(ISMS, 3)	(ARY, 3)
(ISH, 4)	(ATIONS, 3)	(ARITY, 3)
(ATIONAL, 3)	(IONS, 3)	(Y, 3)

7. Parameter q

q = 2

8. Menge A (1 Element)

0 (= unbestimmt)

9. Menge D (195 Elemente)

$$D = \left\{ ((C_i, e_j), 0) \,/\, (C_i, e_j) \in G^1 \wedge 0 \in A \wedge \bigvee_{(C_i, e_j)} \overline{\exists ((C_i, e_j), 0)} \right\}$$

Die Menge A enthält nur das Element 0, daraus folgt die Gleichmächtigkeit der Mengen G^1 und D:

$$|G^1| = |D|$$

3.1.4 Weitere Regeln

J. B. LOVINS erwähnt in (12) weitere Algorithmen von

a) J. W. TUKEY, Princeton University,

b) M. LESK unter Leitung von G. SALTON und

c) J. L. DOLBY, R and D Consultants, Los Altos, California,

alle aufgrund persönlicher Mitteilungen.

D. S. COLOMBO beschreibt in (1) einen Regelsatz für eine Wortstammbildung ("fragmentation algorithm"). Er benutzt dabei einen rekursiven Algorithmus, der in Form eines Flussdiagramms im Anhang II dargestellt ist.[1] Für die erzeugten Stämme wird eine Mindestlänge von drei Zeichen gefordert. Eine Darstellung des Regelsatzes in der in 2 abgeleiteten Form - durch Mengen - ist zwar möglich, es wird jedoch darauf verzichtet, da die Angaben über die verwendeten Suffixlisten in der Arbeit unvollständig sind.

---

[1] Übernommen aus (1)

4 Anwendung spezieller Regelsätze auf eine Wortkollektion

4.1 Ergebnisse - Wortstämme und Wortarten

Die in 3.1 beschriebenen Regelsätze wurden auf 648 zufällig ausgewählte Wörter angewandt.

(Die Auswahl der Wörter erfolgte aus G. SALTON, "Automatic Information Organisation and Retrieval" (19). Es wurden auf jeder Seite das 3. Wort der 3. Textzeile, das 5. Wort der 5. Textzeile und das 3. Wort der drittletzten Textzeile ausgewählt, sofern es weder "common words" (ARE, AT, A, THESE usw.) noch Ziffern oder Zahlen waren. Von diesen Wörtern wurden ausserdem für jedes 40. Wort

SUFFIX	NEGATIVE	PURPOSE
DICTIONARY	SIMPLE	LARGER
NUMBER	FALLS	COME
DOCUMENTS	MEASURE	SHOWS
REQUIRED	REJECTED	

anhand des Shorter Oxford Dictionary (21) "maximale Wortklassen" gebildet, die in die Wortkollektion einbezogen wurden. Schliesslich wurden die Wörter alphabetisch geordnet.)

Die Ergebnisse - Wortstämme und Wortarten - sind in den Tafeln III, 1 bis III, 18 im Anhang III dargestellt, und zwar in

Spalte 1	Wort,
2	Wortart nach LEHNERT (11),
3	" " EURATOM-Regeln,
4	Wortstamm, gebildet durch EURATOM-Regeln,
5	" " " SM-Regeln,
6	" " " LOVINS-Regeln.

Die für die Bewertung definierten Wortklassen (siehe auch 2.8 und 4.2) sind durch Striche voneinander getrennt.

Die technische Durchführung der folgenden Untersuchungen erfolgte mit einem von H. FANGMEYER (EURATOM (Cetis), Ispra) erstellten Programm, dessen Möglichkeiten der in 2 beschriebenen Abbildungs-

funktion entsprechen. Eine Beschreibung dieses Programms sowie eine Benutzeranleitung finden sich im Anhang V (Programm SUFFANAL).

4.2 Ein Kriterium zur Bewertung der Wortstammbildung

Methoden zur Bewertung von Stammbildungsalgorithmen finden sich bei SALTON (19) und LOVINS (13). Diese Methoden sind jedoch anwendungsorientiert, und zwar in Hinblick auf Dokument-Retrieval-Systeme.

An dieser Stelle soll jedoch nur versucht werden, zu beurteilen, ob die Stammbildung selbst befriedigend gelöst wird, d.h. bis zu welchem Grad eine automatische Suffixanalyse für morphologisch verwandte Wörter gleiche Stämme erzeugen kann.

Für eine derartige Beurteilung wurden Klassen morphologisch verwandter Wörter der Testkollektion intellektuell definiert.

Die Bestimmung solcher Wortklassen ist jedoch subjektiv. Beispiel: Die Wörter FACT und FACTOR haben ihrer Herkunft nach zwar den gleichen Stamm, die Wortbedeutungen sind jedoch so verschieden, dass die Wörter nicht auf einen gemeinsamen Stamm reduziert werden sollten. In solchen Zweifelsfällen wurden die Wörter verschiedenen Klassen zugeteilt.

In dem folgenden Kriterium werden bewertet:

(1) Gleiche Stammbildung innerhalb einer Wortklasse

(2) Ungleiche Stammbildung verschiedener Wortklassen

Worthäufigkeiten blieben unberücksichtigt.

Abkürzungen:

a = Anzahl aller Wörter der Wortkollektion

k = Anzahl der in der Wortkollektion definierten Wortklassen $(1 \leq k \leq a)$

s_i = Anzahl der durch die Suffixanalyse gebildeten verschiedenen Stämme in der i-ten Wortklasse $(s_i \geq 1)$

$$s = \sum_{i=1}^{k} s_i \quad (k \leq s \leq a)$$

u = Anzahl der eindeutig repräsentierten Wortklassen. (Eine Wortklasse ist genau dann durch die gebildeten Stämme eindeutig repräsentiert, wenn keiner dieser Stämme auch von Wörtern anderer Klassen gebildet wurde.)
$(0 \leqslant u \leqslant k \;;\; u \leqslant s)$

zu (1): Masszahl für die Stammbildung innerhalb der Wortklassen:

$$M_1 = \frac{a - s}{a - k}$$

$M_1 = 1$ für $a = k$

zu (2): Masszahl für die Verschiedenheit der Stämme aus verschiedenen Wortklassen:

$$M_2 = \frac{u}{k}$$

Die Masszahl M wird gebildet aus dem Produkt von M_1 und M_2:

$$M = M_1 . M_2 = \frac{a - s}{a - k} \cdot \frac{u}{k}$$

$M = \frac{u}{a}$ für $k = a$

Behauptung: $0 \leqslant M_1, M_2, M \leqslant 1$

Beweis: Es gelten die Ungleichungen:

$1 \leqslant k \leqslant s \leqslant a$ (I)
$0 \leqslant u \leqslant k$ (II)

(I) $\Rightarrow$ $a, s, k \geqslant 0 \wedge a \geqslant s \wedge a \geqslant k \Rightarrow M_1 \geqslant 0$

(I) $\Rightarrow$ $a \geqslant s \wedge a \geqslant k \wedge s \geqslant k \Rightarrow M_1 \leqslant 1$

(II) $\Rightarrow$ $u, k \geqslant 0 \Rightarrow M_2 \geqslant 0$

(II) $\Rightarrow$ $u \leqslant k \Rightarrow M_2 \leqslant 1$

$\Rightarrow 0 \leqslant M_1 \leqslant 1 \wedge 0 \leqslant M_2 \leqslant 1$

$\Rightarrow 0 \leqslant M = M_1 . M_2 \leqslant 1$

Extremfälle: 1. Idealfall

$M = 1 \Rightarrow M_1 = 1 \wedge M_2 = 1$

$\Rightarrow (a = k \vee k = s) \wedge u = k$

$\Rightarrow (u = k = s) \vee (u = k = a)$

$\Rightarrow (u = k = s) \vee (u = k = s = a)$

$\Rightarrow u = k = s$

(I)

Es wird für jede Wortklasse genau ein Stamm gebildet (k=s) und alle diese Stämme sind verschieden (u=s).

2. Ungünstigster Fall

$$M = 0 \Rightarrow M_1 = 0 \vee M_2 = 0 \vee M_1 = M_2 = 0$$
$$\Rightarrow (a = s \wedge a \neq k) \vee (u = 0)$$
$$\vee (a = s \wedge a \neq k \wedge u = 0)$$

Entweder
sind alle Stämme innerhalb jeder Wortklasse verschieden ($a = s$), wobei mindestens eine Wortklasse mehr als ein Wort enthält ($a \neq k$),

oder/und
keine Wortklasse ist eindeutig repräsentiert ($u = 0$).

4.3 Bewertung der Regelsätze

a) Wortstammbildung

Für die drei zur Verfügung stehenden Regelsätze ergaben sich nach dem in 4.2 definierten Kriterium folgende Masszahlen für die Wortstammbildung:

EURATOM-Regeln	M = 0,83
SM-Regeln	M = 0,81
LOVINS-Regeln	M = 0,69

Tabellen zur Berechnung dieser Masszahlen finden sich im Anhang IV, Tafeln IV,1 bis IV,3.

b) Bewertung der Wortartbestimmung

Eine Bestimmung der Wortarten wird nur mit den EURATOM-Regeln durchgeführt.

Für die 648 Wörter der Wortkollektion wurden anhand des "Rückläufigen Wörterbuches der englischen Gegenwartssprache" von M. LEHNERT (11) und des "Shorter Oxfort Dictionary" (21) die Wortarten bestimmt. Dabei bedeuten:

N = Substantiv
V = Verb
A = Adjektiv oder Adverb
O = Sonstige Wortarten

Die durch O gekennzeichneten Wörter sind im wesentlichen Homographen.

Man erhielt folgende Ergebnisse:

Wortart nach LEHNERT und Sh. Oxf. Dictionary	Wortart nach EURATOM-Regeln N	V	A	O	Gesamtzahl der Wörter		Anteil der Wörter, deren Wortart richtig bestimmt wurde	
N	[139]	0	3	71	213		65,3 %	
V	6	[88]	0	48	142	444	62,0 %	68,2 %
A	1	1	[76]	11	89		85,4 %	
O	32	10	12	[150]	204		73,5 %	
	178	99	91	280	648			

Von den 648 Wörtern wurde für 453 die Wortart richtig bestimmt, das sind

69,9 % .

Von den Wörtern, deren Wortart richtig bestimmt wurde, entfallen auf:

Substantive	30,7 % (139)
Verben	19,4 % (88)
Adjektive	16,8 % (76)
Sonstige	33,1 % (150)

Von den Wörtern, deren Wortart falsch bestimmt wurde, entfallen auf:

Substantive	37,9 % (74)
Verben	27,7 % (54)
Adjektive	6,7 % (13)
Sonstige	27,7 % (54)

5 Literatur

(1) D.S. COLOMBO; "Automatic Retrieval Systems and Associated Retrieval Languages".

Computer and Information Science Research Center, The Ohio State University, Columbus, Ohio 43210, 1969 Technical Report Series (OSU-CISRC-TR-69-17)

(siehe Kapitel 3 - 3.1.4 - AII)

(2) J.L. DOLBY, H.L. RESNIKOFF and E. MACMURRAY; "A Tape Dictionary for Linguistic Experiments".

Proceedings - Fall Joint Computer Conference, 1963 (AI, 2)

(3) G.M. DYSON; "Computer Input and the Semantic Organisation of Scientific Terms - I".

Information Storage and Retrieval, Volume 3, 1966/1967 (3)

(4) LOIS L. EARL; "Structural Definition of Affixes from Multisyllable Words".

Mechanical Translation, Volume 9, Number 2, June, 1966 (3)

(5) LOIS L. EARL; "Part-of-Speech Implications of Affixes".

Mechanical Translation, Volume 9, Number 2, June, 1966 (3)

(6) H. FANGMEYER and G. LUSTIG; "The EURATOM Automatic Indexing Project".

Paper presented at the IFIP Congress, Edinburgh, 1968 (3.1.1)

(7) H. FANGMEYER and G. LUSTIG; "Experiments with the CETIS Automatic Indexing System".

Handling of Nuclear Information, International Atomic Energy Agency, Vienna, 1970 (3.1.1)

(8) DONNA HOLMBERG-DICKSON; "Wortstammretrieval: Eine Studie zur gegenwärtigen Lage in Praxis und Forschung".

Berichte zur Anwendungsentwicklung 1, Dezember 1972

(9) B. HORNFECK; "Algebra".

De Gruyter Lehrbuch, Walter de Gruyter & Co, Berlin 1969 • (2.8)

(10) E. M. KEEN; "Suffix Dictionaries".

Information Storage and Retrieval, Scientific Report No. IRS-13, Dezember 1967

(11) M. LEHNERT; "Rückläufiges Wörterbuch der englischen Gegenwartssprache".

VEB Verlag Enzyklopädie, Leipzig 1971
(3 - 4.1 - 4.3 - AI, 2)

(12) J. B. LOVINS; "Development of a Stemming Algorithm".

Mechanical Translation, Volume 11, Numbers 1 and 2, March and June 1968
(3 - 3.1.3 - 3.1.4)

(13) J. B. LOVINS; "Error Evaluation for Stemming Algorithms as Clustering Algorithms".

Journal of the American Society for Information Science, January-February, 1971
(4.2)

(13a) H. MARCHAND; "The Categories and Types of Present-Day English Word-Formation".

C. H. BECK'sche Verlagsbuchhandlung, München, 1969
(3)

(14) E. MATER; "Rückläufiges Wörterbuch der deutschen Gegenwartssprache".

VEB Verlag Enzyklopädie, Leipzig 1965
(3)

(15) S. PERSCHKE, H. FANGMEYER; "Methodology and Software Development for Application in Information Science".

Commission of the European Communities, Joint Research Centre, Ispra Establishment - Italy, Annual Report 1971
(1.3) EUR 4842 e

(16) S. PERSCHKE, H. FANGMEYER; "Automatic Thesaurus Construction for Information Retrieval".

Commission of the European Communities, Joint Research Centre, Ispra Establishment - Italy, Annual Report 1972
(1.3) EUR 5060e

(17) H. L. RESNIKOFF and J. L. DOLBY; "The Nature of Affixing in Written English".

Mechanical Translation, Volume 8, Numbers 3 and 4, June and October, 1965
(3)

(18) H. L. RESNIKOFF and J. L. DOLBY; "The Nature of Affixing in Written English, Part II".

Mechanical Translation, Volume 9, Number 2, June, 1966
(3)

(19) G. SALTON; "Automatic Information Organization and Retrieval".

McGraw-Hill Book Company, New York 1968
(3.1.2 - 4.1 - 4.2)

(20) G. SALTON; "The SMART Retrieval System".

Prentice-Hall, Inc., Englewood Cliffs, N. J., 1971
(1.3 - 3.1.2)

(21) "Shorter Oxford English Dictionary".

Oxford University Press, London, 1967
(2.4.3 - 4.1)

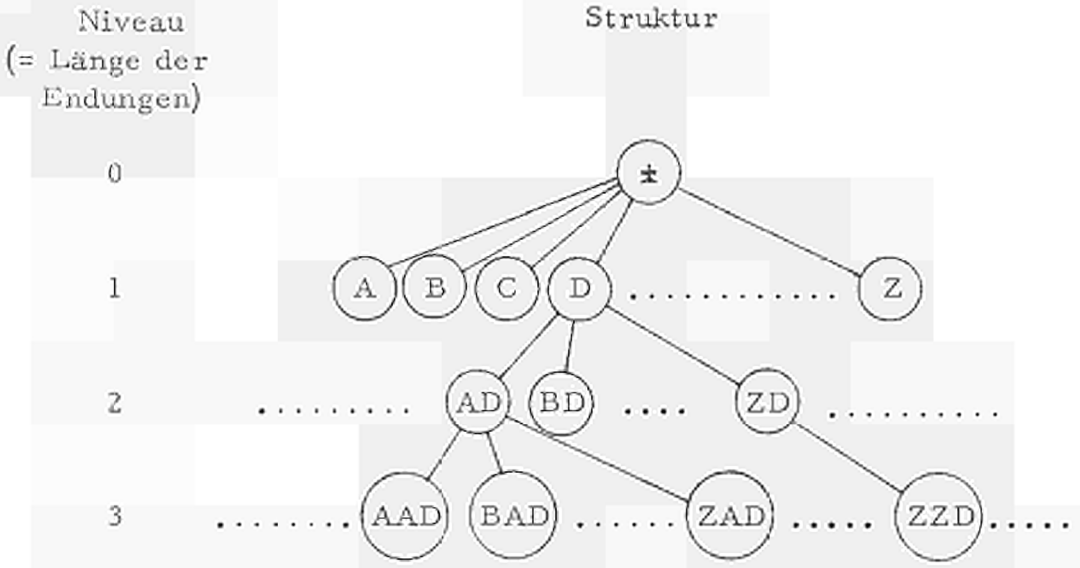
(22) G. WENZEL; "Structure of a Lexicon in Natural Language Processing".

IBM Germany, Wissenschaftliches Zentrum Heidelberg, Report 69.05.002
(2.8)

Anhang I Automatische Erstellung von Suffixlisten
(siehe 3)

1 Algorithmus

Der hier beschriebene automatische Algorithmus setzt ein rückläufiges Wörterbuch in maschinenlesbarer Form voraus.
Jedes Wort des Wörterbuches hat als Grundeigenschaft eine bestimmte Endung; dadurch kann das Wörterbuch nach folgendem Schema strukturiert werden:



Den Wörtern können weitere Eigenschaften zugeordnet sein, etwa die Wortarten.
Beispiele: Die Eigenschaft, eindeutig Substantiv zu sein.
Die Eigenschaft, Substantiv oder Verb zu sein.
Die Eigenschaft, eindeutig Adjektiv zu sein.

Aus allen in dem Wörterbuch definierten Eigenschaften werden m spezielle Eigenschaften e_k ($k = 1, \ldots, m$) ausgewählt.
Alle anderen werden zusammengefasst in der Eigenschaft e_0.
Jedem Wort kann damit genau eine der Eigenschaften e_k mit $k=0, \ldots, m$ zugeordnet werden.

Ein Knoten der obigen Struktur repräsentiert eine Menge M von Wörtern mit gleicher Endung. Die Mengen (Knoten) werden mit den In-

dizes i und j gekennzeichnet; i bestimmt das Niveau und j ist ein laufender Index für die Mengen eines Niveaus ($j=1,\ldots,j_{max}$).

Eigenschaften einer Menge $M_{i,j}$:

$E_{k;i,j}$ = Anzahl der Elemente von $M_{i,j}$ mit der Eigenschaft e_k

$Z_{i,j}$ = Anzahl der Elemente der Menge $M_{i,j}$ (Mächtigkeit)

$$Z_{i,j} = \sum_{k=0}^{m} E_{k;i,j}$$

$P_{l;i,j}$ = Verhältnis von $E_{l;i,j}$ zu einer Funktion aller $E_{k;i,j}$

$$P_{l;i,j} = \frac{E_{l;i,j}}{f(E_{0;i,j}, E_{1;i,j}, \ldots, E_{m;i,j})}$$

$E_{h;i,j}$ = Maximum aller $E_{k;i,j}$ mit $k \geq 1$

$$E_{h;i,j} = \text{Max}\left\{E_{k;i,j} \,/\, k = 1, \ldots, m\right\}$$

Daraus folgt:

e_h ist die in der Menge $M_{i,j}$ am häufigsten auftretende spezielle Eigenschaft und $P_{h;i,j}$ die entsprechende Verhältniszahl.

Damit sind jeder Menge $M_{i,j}$ zwei Grössen zugeordnet,

$$Z_{i,j}(M_{i,j}) \text{ und } P_{h;i,j}(M_{i,j})$$

mit deren Hilfe die folgenden Kriterien zur Bestimmung von Suffixen definiert werden können.

Kriterien zur Bestimmung von Suffixen:

I Die Anzahl $Z_{i,j}$ der Elemente (Wörter) einer Menge $M_{i,j}$ muss mindestens gleich einer Konstanten U_1 sein.

$$Z_{i,j} \geqslant U_1$$

II Das Verhältnis der Anzahl der Elemente mit der Eigenschaft e_h zu einer Funktion aller $E_{k;i,j}$ $(k=0,\ldots,m)$ in der Menge $M_{i,j}$ muss mindestens gleich einer Konstanten U_2 sein.

$$P_{h;i,j} \geqslant U_2$$

III Die längste Wortendung, die die Kriterien I und II erfüllt, wird ausgewählt.

Bemerkung: Das Kriterium III ist automatisch dadurch erfüllt, dass der in Tafel I, 1 beschriebene Algorithmus die Struktur des Wörterbuches von dem Niveau n her abarbeitet (agglomeratives oder bottom-up Verfahren).

Spezielle Anwendungen:

a) Bestimmung von Suffixen unter Berücksichtigung einer speziellen Eigenschaft.

Als Eigenschaften e_k werden Wortarten definiert, z. B.:

k = 1 Substantiv
k = 2 Verb
k = 3 Adjektiv

Alle übrigen Wortarten werden durch k = 0 gekennzeichnet.

b) Bestimmung von Suffixen ohne Berücksichtigung von Eigenschaften.

Es wird nur die Eigenschaft e_0 definiert.

Die Funktion $f(E_{0;i,j})$ wird so festgelegt, dass das Kriterium II immer erfüllt ist:

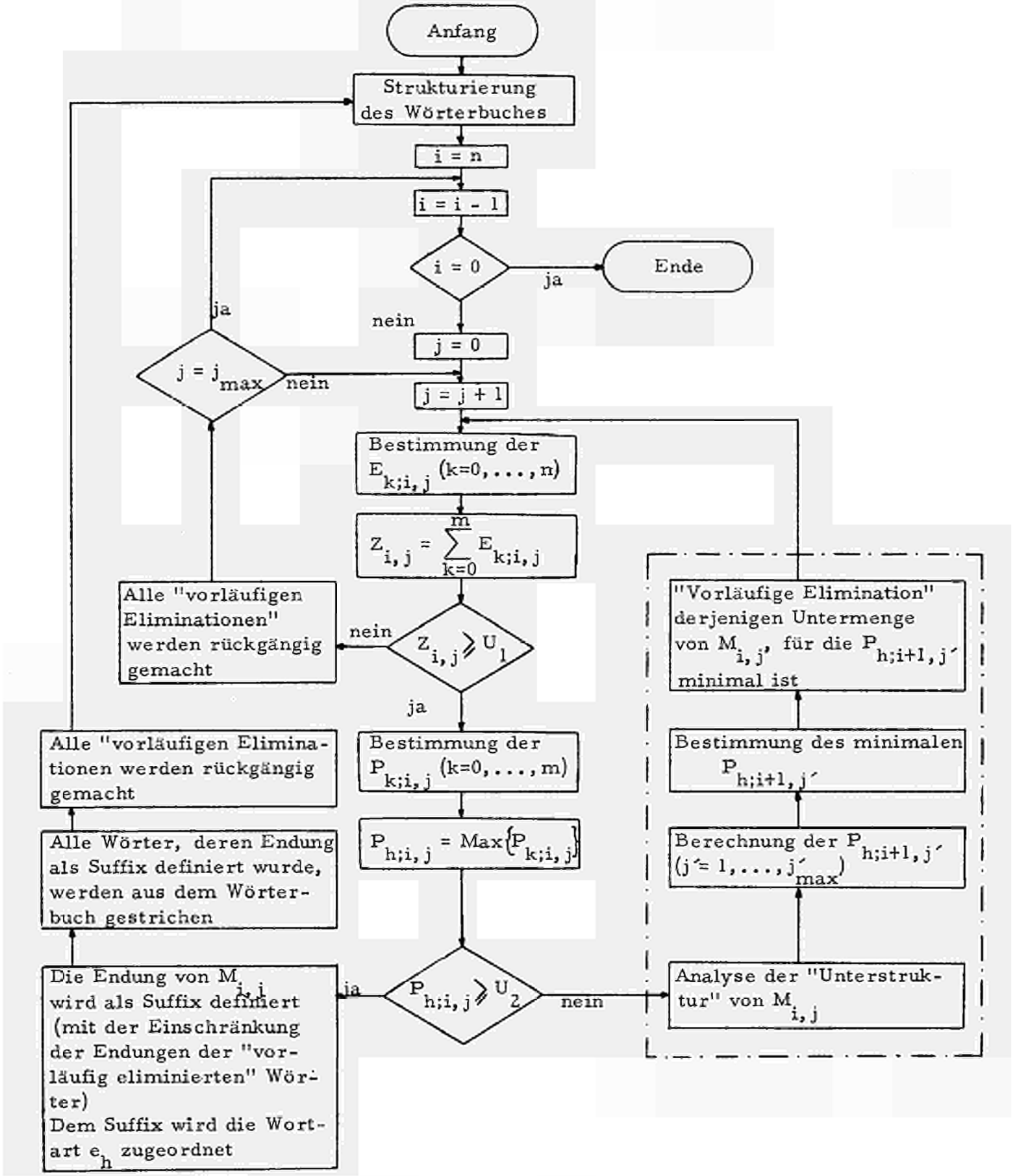
$$f(E_{0;i,j}) = \frac{E_{0;i,j}}{U_2}$$

Daraus folgt:

$$P_{h;i,j} = P_{0;i,j} = \frac{E_{0;i,j} \cdot U_2}{E_{0;i,j}} = U_2$$

Der eingerahmte Teil des Algorithmus' in Tafel I, 1 wird nicht durchlaufen, d.h. praktisch wird nur das Kriterium I angewandt.

Algorithmus zur automatischen Erstellung von Suffixlisten



Tafel I, 1

2 Beispiel zur automatischen Erstellung einer Suffixliste

Ausgangsmaterial bildet das "Rückläufige Wörterbuch der englischen Gegenwartssprache" von M. LEHNERT (11), welches ca. 110000 Stichwörter und deren Wortart umfasst.

Ausdrücke, die aus zwei oder mehr Wörtern bestehen (z. B. PEACE PARADE), bleiben unberücksichtigt.

Da das Wörterbuch

1. von Verben nur den Infinitiv,
2. von Substantiven nur den Nominativ Singular und
3. von Adjektiven keine adverbialen Formen und Steigerungsstufen

enthält, kann für folgende Wortendungen kein statistischer Ansatz gemacht werden:

ED	Verb (Perfekt und Partizip)
ING	Verb (Partizip)
ER	Adjektiv (Komparativ)
S ES	Verb (3. Pers. Singular), Substantiv (Plural)
'S	Substantiv (Genitiv)
EST	Adjektiv (Superlativ)
LY	Adverb eines Adjektives
S'	Substantiv (Genitiv)

Alle Wörter mit diesen Endungen werden aus dem Wörterbuch eliminiert. Für diese Endungen werden Suffixregeln nach grammatikalischen Gesichtspunkten definiert.

Für alle anderen Endungen (ca. 70000 Wörter) kann der in Tafel I, 1 beschriebene Algorithmus angewandt werden, falls das Wörterbuch in maschinenlesbarer Form vorliegt.

(Ein von J. L. DOLBY, H. L. RESNIKOFF und E. MACMURRAY erstelltes Wörterbuch auf Magnetband, das ca. 75000 Wörter und deren Wortarten enhält, ist in (2) beschrieben.)

In diesem Beispiel sollen speziell für den Endbuchstaben D Suffixe gefunden werden, und zwar im

1. Schritt alle Suffixe, die charakteristisch sind für eine der Wortarten Substantiv, Verb oder Adjektiv mit einer Genauigkeit (ohne Berücksichtigung der Häufigkeiten) von 95%, und im
2. Schritt Suffixe für Wörter, für die keine Regel im 1. Schritt gefunden wurde. Diese Suffixe eignen sich also nicht für eine Wortartbestimmung.

Für jedes Suffix wird gefordert, dass es Endung ist von mindestens 100 Wörtern des Wörterbuches.

Parameter für den 1. Schritt:

Eigenschaften: e_1 = Wortart ist eindeutig Substantiv (N)
e_2 = Wortart ist eindeutig Verb (V)
e_3 = Wortart ist eindeutig Adjektiv (A)
e_0 = sonstige Wortarten (O)[1)]

Funktion f: $f(E_{0;i,j}, \ldots, E_{3;i,j}) = \dfrac{\sum_{k=1}^{3} E_{k;i,j}}{100}$

Daraus folgt:

$P_{1;i,j} = \dfrac{E_{1;i,j} \cdot 100}{\sum_{k=1}^{3} E_{k;i,j}}$ = Prozentuales Verhältnis der Anzahl von Wörtern mit einer bestimmten Wortart zu der Gesamtzahl von Wörtern mit eindeutigen Wortarten in der Menge $M_{i,j}$.

$P_{1;i,j} = c \leqslant U_2$ für $\sum_{k=1}^{3} E_{k;i,j} = 0$

U_1 = 100 (Wörter)
U_2 = 95 (%)

[1)] Die Wortart O haben also alle Artikel, Pronomen usw., ebenso alle Homographen, die mehreren Wortarten angehören. Beispiel:
SPARE N (Ersatzteil)
V (sparen)
A (spärlich)

Parameter für den 2. Schritt:

Eigenschaften: e_0 = beliebige Wortart (O)

Funktion f: $f(E_{0;i,j}) = \frac{E_{0;i,j}}{U_2}$

Daraus folgt:

$$P_{h;i,j} = P_{0;i,j} = \frac{E_{0;i,j} \cdot U_2}{E_{0;i,j}} = U_2$$

Damit ist das Kriterium II immer erfüllt.

U_1 = 100 (Wörter)

U_2 = beliebig

Mit diesem Algorithmus ergeben sich folgende Regeln (für den Endbuchstaben D ohne ED):

	Wortart	Anzahl der Wörter mit dieser Endung
(¬1, EAD)	N (97,8 %)	105
(¬2, AD)	N (96,5 %)	114
(± , OID)	O	342
(± , ID)	O	349
(± , LD)	O	167
(± , AND)	O	181
(± , END)	O	100
(± , OUND)	O	121
(± , ND)	O	129
(± , OOD)	N (98,4 %)	200
(± , OD)	O	102
(¬3, ARD)	N (97,3 %)	265
(± , RD)	O	230
(± , D)	O	145

1 := DvLvNvR
2 := EvLvS
3 := DvW

Zum Vergleich die in 3.1 beschriebenen Regelsätze:

EURATOM-Regeln

(¬V, OID)	N (75,8%)	334
(± , UID) (U, ID)	N (40,0%)	11
(± , ID)	O	346
(AN, D)	N (85,1%)	181
(OR, D)	N (88,9%)	65

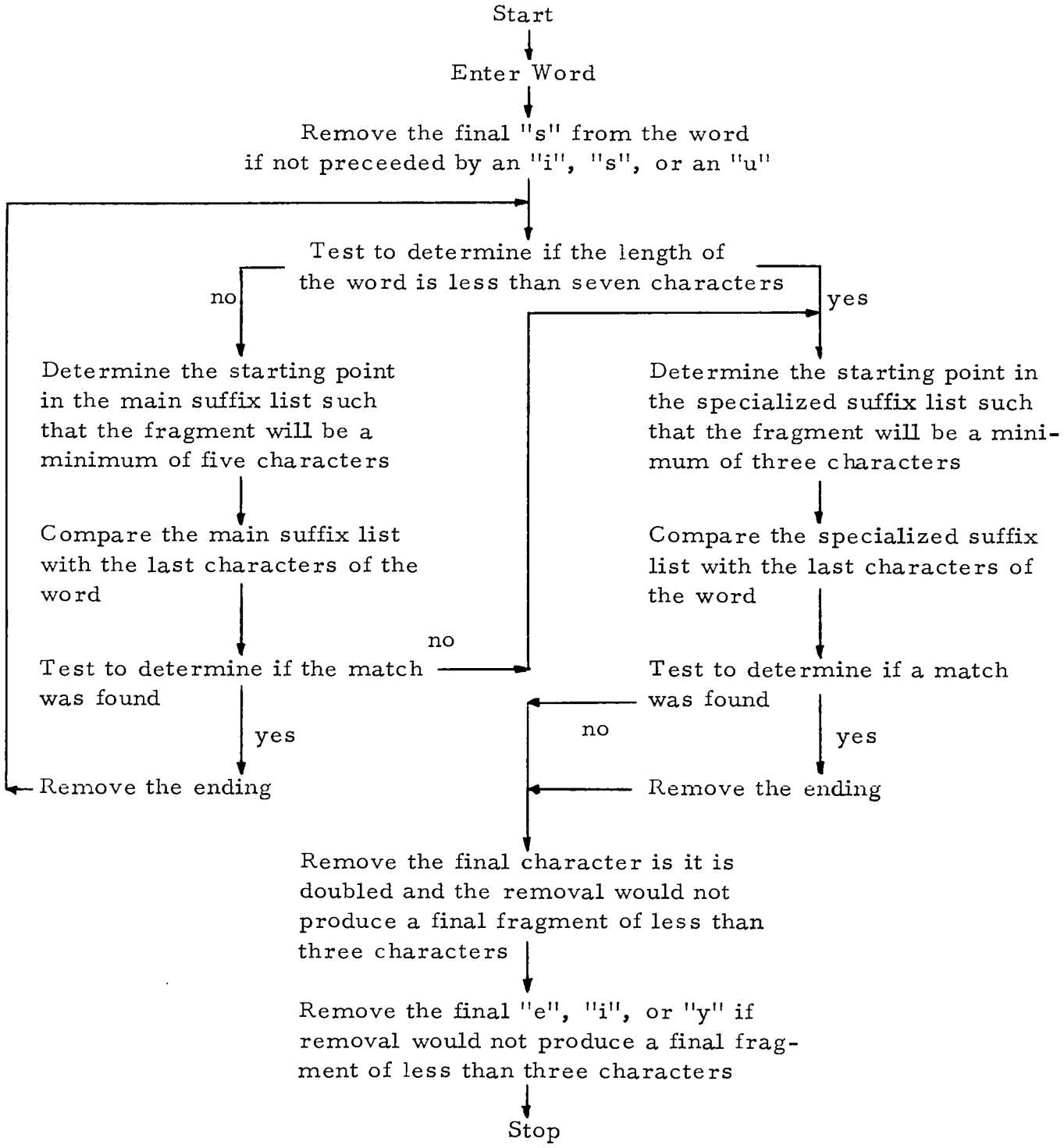
SM-Regeln

(± , FOLD)	32
(± , HOOD)	92

LOVINS-Regeln

(± , AROID)	7
(± , OID)	335
(± , EHOOD)	12
(¬E,ELIHOOD)	3
(± , IHOOD)	2
(± , HOOD)	75
(± , WARD)	66

Anhang II Flussdiagramm eines "fragmentation algorithm"
von D. S. COLOMBO
(übernommen aus (1), siehe 3.1.4)



Outline in flowchart form of the fragmentation algorithm used in the batch and on-line system

Anhang III Ergebnisse bei Anwendung spezieller Regelsätze auf eine Wortkollektion - Listen der Wortstämme und Wortarten - (siehe 4.1)

ABSTRACT	N, V, A	ABSTRACT	ABSTRACT	ABSTRACT
ACCESSIBILITY	N	ACCESSIBIL	ACC	ACCESS
ADD	V	ADD	ADD	AD
ADDITION	N, V	ADDIT	ADD	ADDIT
ADVANTAGEOUS	A	ADVANT	ADVANTAG	ADVANTAG
ADVERSELY	A	ADVER	ADVERSE	ADVERS
ALGORITHM	N	ALGOR	ALGORITHM	ALGORITHM
ALL	N, A, O	ALL	ALL	AL
AMBIGUOUS	A	AMBIGU	AMBIGU	AMBIGU
ANALYSIS	N	ANAL	AN	ANALYS
ANSWER	N, V	ANSW	ANSW	ANSWER
ANSWERING	N	ANSW	ANSW	ANSWER
ANSWERS	N, V	ANSW	ANSW	ANSWER
APPARENT	A	APPAR	APP	APPAR
APPENDIX	N, V	APPEND	APPENDIX	APPENDIC
APPLICABLE	A	APPL	APPL	APPLIC
APPLIED	A, V	APPL	APPLI	APPL
AREAS	N	AREA	ARE	ARE
ARRANGED	V	ARRANG	ARRANG	ARRANG
ASCERTAINED	V	ASCERTAIN	ASCERTAIN	ASCERTAIN
ASSIGNED	V	ASSIGN	ASSIGN	ASSIGN
ASSIGNMENT	N	ASSIGN	ASSIGN	ASSIGNM
ASSOCIATION	N	ASSOC	ASSOCI	ASSOCI
ASSUME	V	ASSUM	ASSUME	ASSUM
ATOMIC	A	ATOM	ATOM	ATOM
AUTOMATIC	N, A	AUTOM	AUTOMAT	AUTOM
AVAILABLE	A	AVAIL	AVAIL	AVAIL
AVERAGE	N, V, A	AVER	AV	AVER
AVERAGED	V	AVER	AVERAG	AVERAG
BASE	N, V, A	BASE	BASE	BAS

BASEBALL	N	0	BASEBALL	BASEBALL	BASEBAL
BEING	N, V	0	BEING	BE	BEING
BENEFIT	N, V	0	BENEF	BENEFIT	BENEFIT
BLUEPRINTS	N, V	0	BLUEPRINT	BLUEPRINT	BLUEPRINT
BOTTOM	N, V, A	0	BOTTOM	BOTTOM	BOTTOM
BOUND	N, V, A	0	BOUND	BOUND	BOUND
BROTHERS	N, V	N	BROTH	BRO	BROTHER
BUCKET	N, V	0	BUCKET	BUCKET	BUCKES
BUILT	V	0	BUILT	BUILT	BUILT
CALCULATIONS	N	N	CALCUL	CALCUL	CALCUL
CALLED	V	V	CALL	CALL	CALL
CARDS	N, V	0	CARD	CARD	CARD
CASE	N, V	0	CASE	CASE	CAS
CATALOGS	N, V	0	CATALOG	CATALOG	CATALOG
CATEGORY	N	N	CATEG	CATEG	CATEGOR
CENTROIDS	N	N	CENTR	CENTROID	CENTROID
CERTAIN	N, A	0	CERTAIN	CERTAIN	CERTAIN
CHANCES	N, V	0	CHANC	CH	CHANC
CHANGE	N, V	0	CHANG	CHANGE	CHANG
CHAPTER	N, V	0	CHAPT	CHAPT	CHAPTER
CHARACTERS	N, V	N	CHARACT	CHARACT	CHARACTER
CHOOSING	V	V	CHOO	CHOO	CHOOS
CIRCUMSTANCES	N, V	N	CIRCUMST	CIRCUMST	CIRCUMST
CITE	V	0	CITE	CITE	CIT
CLASSES	N, V	0	CLAS	CL	CLASS
CLASSIFICATION	N	N	CLAS	CL	CLASSIFIC
CLOSED	V	V	CLOS	CLO	CLOS
CLUSTER	N, V	0	CLUST	CLUST	CLUSTER
CLUSTERING	V	V	CLUST	CLUST	CLUSTER

Tafel III, 2

CLUSTERS	N, V	NO	CLUST	CLUST	CLUSTER
CO-OCCURRENCES	N	NO	CO-OCCURR	CO-OCCURR	CO-OCCURR
CODES	N	OO	CODE	COD	COD
CODING	N	OO	CODING	COD	COD
COLLECTION	N	ON	COLLECT	COLLECT	COLLECT
COLUMNS	N	OO	COLUMN	COLUMN	COLUMN
CAME	N, V	OO	CAME	CAME	CAM
COME	V, O	OO	COME	COME	COM
COMER	N	OO	COMER	COM	COMER
COMERS	N	OO	COMER	COM	COMER
COMES	V	OO	COME	COM	COM
COMING	N, V, A	OO	COMING	COM	COM
COMINGS	N	OO	COMING	COM	COM
COMPARE	N, V	OO	COMP	COMPARE	COMPAR
COMPARING	V	VO	COMP	COMP	COMPAR
COMPLETE	V, A	OA	COMPLET	COMPLETE	COMPLET
COMPLETELY	A	AO	COMPLET	COMPLETE	COMPLET
COMPLEX	N, V, A	NO	COMPL	COMPLEX	COMPLEC
COMPUTER	N	OO	COMP	COMPUT	COMPUTER
CONCEPTS	N	ON	CONC	CONCEPT	CONCEPT
CONDITION	N, V	NO	COND	COND	CONDIT
CONNECTION	N	NO	CONNECT	CONNECT	CONNECT
CONNECTIONS	N	NO	CONNECT	CONNECT	CONNECT
CONSIDER	V	OO	CONS	CONSID	CONSIDER
CONSISTS	N, V	OO	CONSIST	CON	CONS
CONSULTED	V	OV	CONSULT	CONSULT	CONSULT
CONTEND	V	OO	CONTEND	CONTEND	CONTENS
CONTRARY	N, A	OA	CONTR	CONTR	CONTR
CONTROL	N, V	OO	CONTROL	CONTROL	CONTROL
COORDINATES	N, V	OO	COORDIN	COORDIN	COORDIN
COORDINATION	N	ON	COORDIN	COORDIN	COORD
CORRECT	V, A	OO	CORRECT	CORRECT	CORRECT
CORRELATION	N	NN	CORREL	CORREL	CORREL
CORRELATIONS	N	NN	CORREL	CORREL	CORREL

COSINE	N	COSIN	CO	COS
COVERED	V	COVER	COV	COVER
CRITERION	N	CRIT	CRIT	CRITER
DATA	N	DATA	CAT	DAT
DEALT	V	DEALT	DEALT	DEALT
DECK	N, V	DECK	DECK	DECK
DEEPER	A	DEEP	DEEP	DEEPER
DEFICIENCY	N	DEFICI	DEFICI	DEFICI
DEFINED	V	DEFIN	DEFIN	DEFIN
DEFINITIONS	N	DEFIN	DEFIN	DEFINIT
DENSITY	N	DENS	DEN	DENS
DEPENDING	V	DEPEND	DEPEND	DEPEND
DERIVED	V	DERIV	DERIV	DERIV
DESIGN	N, V	DESIGN	DESIGN	DESIGN
DESIGNED	V, A	DESIGN	DESIGN	DESIGN
DESK	N	DESK	DESK	DESK
DETAILED	V	DETAIL	DETAIL	DETAIL
DETERMINATION	N	DETERMIN	DETERMIN	DETERM
DETERMINE	V	DETERMIN	DETERM	DETERMIN
DEVELOPMENTS	N	DEVELOP	DEVELOP	DEVELOPMENT
DICTION	N	DICT	DICT	DICT
DICTIONARIAN	N	DICT	DICT	DICTIONAR
DICTIONARIANS	N	DICTIONARIAN	DICT	DICTIONAR
DICTIONARIES	N	DICT	DICT	DICTION
DICTIONARY	N	DICT	DICT	DICTION
DICTIONS	N	DICT	DICT	DICT
DIFFERENTIAL	N, A	DIFF	DIFF	DIFFER
DIRECTED	V	DIRECT	DIRECT	DIRECT
DIRECTION	N	DIRECT	DIRECT	DIRECT
DISPLAY	N, V	DISPLA	DISPL	DISPLA
DISTINCT	A	DISTINCT	DISTINCT	DISTINCT
DISTINGUISH	V	DISTINGU	DISTINGU	DISTINGU

DOCUMENT	N, V	N	DOCU	DOCU	DOCUM
DOCUMENTAL	A	O	DOCUM	DOCU	DOCUM
DOCUMENTALLY	A	A	DOCUM	DOCU	DOCUM
DOCUMENTARILY	A	A	DOCUM	DOCU	DOCUMENT
DOCUMENTARY	N, A	A	DOCUM	DOCU	DOCUMENT
DOCUMENTATION	N	N	DOCUM	DOCU	DOCUM
DOCUMENTATIONS	N	N	DOCUM	DOCU	DOCUM
DOCUMENTED	V	V	DOCUM	DOCU	DOCUM
DOCUMENTING	V	V	DOCUM	DOCU	DOCUM
DOCUMENTIZE	V	V	DOCUM	DOCU	DOCUMENT
DOCUMENTIZED	V	V	DOCUM	DOCU	DOCUMENT
DOCUMENTIZES	V	O	DOCUM	DOCU	DOCUMENTIZ
DOCUMENTIZING	V	V	DOCUM	DOCU	DOCUMENT
DOCUMENTS	N, V	N	DOCU	DOCU	DOCUMENT
DONE	V, A	O	DONE	DONE	DON
EDITING	V	V	EDIT	EDIT	EDIT
EFFECTIVELY	A	A	EFFECT	EFFECT	EFFECT
EFFECTIVENESS	N	N	EFFECT	EFFECT	EFFECT
EFFICIENT	N, A	O	EFFICI	EFFICI	EFFICI
ELEMENTS	N	N	ELEM	ELE	ELEMENT
ENABLES	V	N	ENAB	ENABL	ENABL
ENTER	V	O	ENTER	ENT	ENTER
ENTRY	N	O	ENTR	ENT	ENTR
EQUALITY	N	N	EQUAL	EQU	EQU
EQUIVALENCE	N	N	EQUIV	EQUIV	EQUIVAL
EXAMINED	V	V	EXAMIN	EXAMIN	EXAMIN
EXAMPLE	N, V	O	EXAMP	EXAMPLE	EXAMPL
EXCEED	V	O	EXCEED	EXCE	EXCEED
EXCEPT	V, O	O	EXCEPT	EXCEPT	EXCEPT
EXCERPT	N, V	O	EXCERPT	EXCERPT	EXCERB
EXHIBITED	V	V	EXHIB	EXHIBIT	EXHIBIT
EXHIBITS	N, V	N	EXHIB	EXHIBIT	EXHIBIT
EXPAND	V	N	EXPAN	EXPAND	EXPANS
EXPANSION	N	N	EXPAN	EXP	EXPANS
EXPECTED	V	V	EXPECT	EXPECT	EXPECT
EXPRESSED	V	V	EXPR	EXPR	EXPRESS
EXTENT	N	O	EXTENT	EXT	EXTENS

FACT	N	0	FACT	FACT	FACT
FACTOR	N, V	0	FACT	FACT	FACT
FAILED	V	0	FAIL	FAIL	FAIL
FALL	N, V	0	FALL	FALL	FAL
FALLEN	V	0	FALLEN	FALL	FALL
FALLER	N	0	FALL	FALL	FALLER
FALLERS	N	N	FALL	FALL	FALLER
FALLING	N, V	V	FALL	FALL	FALL
FALLINGS	N	N	FALL	FALL	FALL
FALLS	N, V	0	FALL	FALL	FALL
FELL	N, V, A	0	FELL	FELL	FEL
FELLED	V	V	FELL	FELL	FELL
FELLING	V, N	V	FELL	FELL	FELL
FELLINGS	N	N	FELL	FELL	FELL
FELLS	N, V	0	FELL	FELL	FELL
FAR	V, A	0	FAR	FAR	FAR
FEELING	N, A, V	0	FEEL	FEEL	FEEL
FIELDS	N, V	0	FIELD	FIELD	FIELD
FILE	N, V	0	FILE	FILE	FIL
FILL	N, V	0	FILL	FILL	FIL
FINAL	N, A	0	FINAL	FIN	FIN
FIRST-ORDER	O	0	FIRST-ORD	FIRST-ORD	FIRST-ORDER
FIT	N, V, A	0	FIT	FIT	FIT
FLAG	N, V	0	FLAG	FLAG	FLAG
FLIGHT	N, V	0	FLIGHT	FLIGHT	FLIGHT
FLOWCHART	N	0	FLOWCHART	FLOWCHART	FLOWCHART
FOUND	V	0	FOUND	FOUND	FOUND
FREQUENCY	N	N	FREQU	FREQU	FREQU
FUNCTION	N, V	N	FUNCT	FUNCT	FUNCT
GENERAL	N, A	0	GENE	GEN	GENER
GENERATING	V	V	GENER	GEN	GENER
GIVEN	V	0	GIVEN	GIV	GIV
GRAMMARS	N	0	GRAMM	GRAMM	GRAMMAR

GRAPH	N, V	GRAPH	GRAPH	GRAPH
GROUPINGS	N	GROUP	GROUP	GROUP
GROUPS	N, V	GROUP	GROUP	GROUP
HAND	N, V	HAND	HAND	HAND
HARDWARE	N	HARDW	HARDWARE	HARDWAR
HAVING	N, V	HAVING	HAV	HAV
HIERARCHICAL	A	HIERARCH	HIERARCH	HIERARCH
HIERARCHIES	N	HIERARCH	HIERARCH	HIERARCH
HIGHEST	A	HIGH	HIGH	HIGHEST
HIGHLY	A	HIGH	HIGH	HIGH
HOLD	N, V	HOLD	HOLD	HOLD
HOUSE	N, V	HOUS	HOUSE	HOUS
IDENTICAL	A	IDENT	ID	IDENT
IDENTIFICATION	N	IDENT	ID	IDENTIFIC
IDENTIFIED	V	IDENT	ID	IDENTIF
IDENTIFIERS	N	IDENTIFI	IDENTI	IDENTIF
IDENTIFY	V	IDENT	IDENTI	IDENTIF
IDENTIFYING	V	IDENT	IDENTI	IDENTIF
IMPORTANCE	N	IMPORT	IMPORT	IMPORT
IMPURTANT	A	IMPORT	IMPORT	IMPORT
INCLUDE	V	INCLUD	INCLUDE	INCLUD
INCLUDED	V	INCLUD	INCLUD	INCLUD
INCLUDES	V	INCLUD	INCLUD	INCLUD
INCLUDING	V, O	INCLUD	INCLUD	INCLUD
INCORPORATED	V	INCORPOR	INCORP	INCORPOR
INCORPORATION	N	INCORPOR	INCORP	INCORPOR
INCREASE	N, V	INCREA	INCREASE	INCREAS
INDEXED	V	INDEX	INDEX	INDEX
INFERENCE	N	INFER	INF	INFER
INFORMATION	N	INFORM	INFORM	INFORM
INITIAL	N, V, A	INIT	INITI	INIT
INPUT	N	INPUT	INPUT	INPUT
INSTRUCTIONS	N	INSTRUCT	INSTRUCT	INSTRUCT
INTERMEDIATE	N, A, V	INTERM	INTERMEDI	INTERMEDI

INTERPRETATION	N	N	INTERPRET	INTERPRET	INTERPRET
INTERSTING	V, A	V	INTERST	INTERST	INTERST
INTRODUCED	V	V	INTRODUC	INTRODUC	INTRODUC
INTRODUCING	V	V	INTRODUC	INTRODUC	INTRODUC
ISOLATES	V	0	ISOL	ISOL	ISOL
ITEMS	N, V	0	ITEM	ITEM	ITEM
ITERATIVE	A	A	ITER	IT	ITER
KEEP	N, V	0	KEEP	KEEP	KEEP
KEY	N, V	0	KEY	KE	KEY
KINDS	N	0	KIND	KIND	KIND
LABELED	V	V	LABEL	LABEL	LABEL
LABELS	N, V	0	LABEL	LABEL	LABEL
LANGUAGE	N	N	LANGU	LANGU	LANGU
LANGUAGES	N	N	LANGU	LANGU	LANGU
LARGE	N, A, V	0	LARG	LARGE	LARG
LARGED	V	V	LARG	LARG	LARG
LARGELY	A	A	LARG	LARGE	LARG
LARGEN	V	0	LARGEN	LARG	LARG
LARGENED	V	V	LARGEN	LARG	LARG
LARGENESS	N	N	LARG	LARGE	LARG
LARGENESSES	N	N	LARG	LARGE	LARGE
LARGENING	V	V	LARGEN	LARG	LARG
LARGENS	V	0	LARGEN	LARG	LARGEN
LARGER	A	0	LARG	LARG	LARGER
LARGES	N	0	LARG	LARG	LARG
LARGESSE	N	0	LARG	LARGESSE	LARGESS
LARGESSES	N	0	LARG	LARG	LARGESS
LARGEST	A	A	LARG	LARG	LARGEST
LARGET	N	0	LARGET	LARGET	LARGES
LARGETS	N	0	LARGET	LARGET	LARGET
LARGING	V	V	LARG	LARG	LARG
LARGISH	A	0	LARG	LARG	LARG
LARGISHLY	A	A	LARG	LARG	LARGISH
LARGITION	N	N	LARG	LARG	LARGIT
LARGITIONS	N	N	LARG	LARG	LARGIT
LEADING	N, V, A	V	LEAD	LEAD	LEAD
LESS	N, A, O	0	LESS	LE	LES
LET	N, V	0	LET	LET	LES
LEVELS	N, V	0	LEVEL	LEVEL	LEVEL

Tafel III, 8

LINK	N, V	LINK	LINK	LINK	
LISTED	V	LIST	LIST	LIST	
LISTS	N, V	LIST	LIST	LIST	
LOCATIONS	N	LOCAT	LOC	LOC	
LOOKED	V	LOOK	LOOK	LOOK	
MAIN	N, A	MAIN	MAIN	MAIN	
MAINTAINED	V	MAINTAIN	MAINTAIN	MAINTAIN	
MAINTAINING	V	MAINTAIN	MAINTAIN	MAINTAIN	
MAKING	N, V	MAKING	MAK	MAK	
MATCH	N, V	MATCH	MATCH	MATCH	
MATCHES	N, V	MATCH	MATCH	MATCH	
MATHEMATICS-LOGICS	N	MATHEMATICS-LOG	MATHEMATICS-LOG	MATHEMATICS-LOG	
MATRIX	N	MATR	MATRIX	MATRIC	
MEASURABLE	A	MEAS	MEASUR	MEASUR	
MEASURABLY	A	MEAS	MEASUR	MEASUR	
MEASURE	N, V	MEAS	ME	MEASUR	
MEASURED	V	MEAS	MEASUR	MEASUR	
MEASURELESS	A	MEAS	ME	MEASUR	
MEASURELESSLY	A	MEAS	ME	MEASURE	
MEASURELESSNESS	N	MEAS	MEAS	MEASURE	
MEASURELESSNESSES	N	MEAS	MEASURE	MEASURELESS	
MEASUREMENT	N	MEAS	ME	MEASUR	
MEASUREMENTS	N	MEAS	ME	MEASUREMENT	
MEASURER	N	MEAS	MEASUR	MEASURER	
MEASURERS	N	MEAS	MEASUR	MEASURER	
MEASURES	N, V	MEAS	ME	MEASUR	
MEASURING	N, V	MEAS	MEASUR	MEASUR	
MEASURINGS	N	MEAS	MEASUR	MEASUR	
METHODS	N	METHOD	METHOD	METHOD	
MODEL	N, V	MODEL	MODEL	MODEL	
MODELS	N, V	MODEL	MODEL	MODEL	
MODIFICATION	N	MODIF	MOD	MODIFIC	
MULTILIST	N	MULTIL	MULTIL	MULTIL	
NEED	N, V	NEED	NE	NEED	
NEEDED	V	NEED	NE	NEED	
NEGATE	V	NEGAT	NEG	NEG	
NEGATED	V	NEGAT	NEG	NEG	
NEGATES	V	NEGAT	NEG	NEG	
NEGATING	V	NEGAT	NEG	NEG	

NEGATION	N	N	NEGAT	NEG	NEG
NEGATIONAL	A	A	NEGAT	NEG	NEG
NEGATIONALLY	A	A	NEGAT	NEG	NEG
NEGATIONIST	N	O	NEGATIONIST	NEG	NEGAT
NEGATIONISTS	N	O	NEGATIONIST	NEG	NEGATION
NEGATIONS	N	N	NEGAT	NEG	NEG
NEGATIVE	N, V, A	A	NEGAT	NEG	NEG
NEGATIVED	V	V	NEGAT	NEGATIV	NEGATIV
NEGATIVELY	A	A	NEGAT	NEGAT	NEG
NEGATIVENESS	N	N	NEGAT	NEGAT	NEG
NEGATIVENESSES	N	N	NEGAT	NEGAT	NEGATIVE
NEGATIVES	N, V	O	NEGAT	NEGATIV	NEG
NEGATIVING	V	V	NEGAT	NEGATIV	NEGATIV
NEGATIVITIES	N	N	NEGAT	NEGAT	NEGAT
NEGATIVITY	N	N	NEGAT	NEGAT	NEGAT
NEGATORILY	A	A	NEGATORI	NEGAT	NEGATOR
NEGATORY	A	N	NEGAT	NEGAT	NEG
-		O	-	-	-
NEW	N, A	O	NEW	NEW	NEW
-		O	-	-	-
NODES	N	O	NODE	NOD	NOD
-		O	-	-	-
NORMAL	N, A	A	NORM	NORM	NORM
NORMALIZED	V	V	NORM	NORM	NORMAL
-		O	-	-	-
NOTED	V	O	NOTED	NOT	NOT
-		O	-	-	-
NOUN	N	O	NOUN	NOUN	NOUN
-		O	-	-	-
NUMBER	N, V, A	O	NUMB	NUMB	NUMBER
NUMBERABLE	A	A	NUMB	NUMB	NUMBER
NUMBERABLY	A	A	NUMB	NUMB	NUMBER
NUMBERED	V	V	NUMB	NUMB	NUMBER
NUMBERER	N	O	NUMB	NUMB	NUMBERER
NUMBERERS	N	N	NUMB	NUMB	NUMBERER
NUMBERING	V	V	NUMB	NUMB	NUMBER
NUMBERLESS	A	A	NUMB	NUMB	NUMBER
NUMBERLESSLY	A	A	NUMB	NUMB	NUMBER
NUMBERS	N, V	N	NUMB	NUMB	NUMBER
NUMERAL	N, A	O	NUME	NUM	NUMER
NUMERALLY	A	A	NUME	NUM	NUMER
NUMERALS	N	O	NUME	NUM	NUMER
NUMERARILY	A	A	NUMER	NUM	NUMER
NUMERARY	A	A	NUMER	NUM	NUMER
NUMERATE	V, A	O	NUMER	NUM	NUMER
NUMERATED	V	V	NUMER	NUM	NUMER
NUMERATES	V	O	NUMER	NUM	NUMER
NUMERATING	V	V	NUMER	NUM	NUMER
NUMERATION	N	N	NUMER	NUM	NUMER
NUMERATIONS	N	N	NUMER	NUM	NUMER
NUMERATIVE	A	A	NUMER	NUM	NUMER
NUMERATIVELY	A	A	NUMER	NUMERAT	NUMER
NUMERATOR	N	O	NUMER	NUM	NUMER
NUMERATORS	N	N	NUMER	NUM	NUMER
NUMERIC	N	A	NUME	NUM	NUMER

NUMERICAL	A	A	NUME	NUM	NUMER
NUMERICALLY	A	A	NUME	NUM	NUMER
NUMERICS	N	N	NUMER	NUM	NUMER
NUMERIST	N	O	NUMERIST	NUM	NUMER
NUMERISTS	N	O	NUMERIST	NUM	NUMER
NUMEROSITIES	N	N	NUMERO	NUMERO	NUMEROS
NUMEROSITY	N	N	NUMERO	NUMERO	NUMEROS
NUMEROUS	A	A	NUMER	NUM	NUMER
NUMEROUSLY	A	A	NUMER	NUM	NUMER
NUMEROUSNESS	N	N	NUMER	NUM	NUMER
NUMEROUSNESSES	N	N	NUMER	NUMER	NUMEROUS
-		O	-	-	-
OBTAINED	V	V	OBTAIN	OBTAIN	OBTAIN
-		O	-	-	-
OBVIOUSLY	A	A	OBVI	OBV	OBV
-		O	-	-	-
ONES	N	O	ONES	ON	ON
-		O	-	-	-
OPERATIONS	N	N	OPER	OP	OPER
-		O	-	-	-
ORDER	N, V	O	ORDER	ORD	ORDER
-		O	-	-	-
ORIGINAL	N, A	A	ORIGIN	ORIGIN	ORIGIN
ORIGINALLY	A	A	ORIGIN	ORIGIN	ORIGIN
-		O	-	-	-
PART	N, V, A	O	PART	PART	PART
PARTIAL	N, A	A	PART	PARTI	PART
PARTIALLY	A	A	PARTI	PARTI	PART
PARTITIONING	V	V	PART	PART	PARTIT
-		O	-	-	-
PARTICULAR	N, A	O	PARTICUL	PARTICUL	PARTICUL
PARTICULARLY	A	A	PARTICUL	PARTICUL	PARTICUL
-		O	-	-	-
PERFORMED	V	V	PERFORM	PERFORM	PERFORM
-		O	-	-	-
PHENOMENON	N	N	PHENOMEN	PHENOM	PHENOMENON
-		O	-	-	-
PHRASE	N, V	O	PHRA	PHRASE	PHRAS
PHRASES	N, V	O	PHRA	PHR	PHRAS
-		O	-	-	-
PLEASE	V	O	PLEA	PLEASE	PLEAS
-		O	-	-	-
PLOT	N, V	O	PLOT	PLOT	PLOT
-		O	-	-	-
POINT	N, V	O	POINT	POINT	POINT
POINTER	N	O	POINT	POINT	POINTER
POINTS	N, V	O	POINT	POINT	POINT
-		O	-	-	-
POSITIVE	N, A	A	POSIT	PO	POSIT
-		O	-	-	-
POSSIBLE	N, A	A	POSS	PO	POSS
-		O	-	-	-
PRECEDING	V	V	PREC	PREC	PRECED
-		O	-	-	-
PRECISION	N	N	PREC	PRECI	PRECIS

PREDICTION	N	ONUOONAOOUOONNOVVOOOVOOOOOOOOOOOOOOOOOOOOOOOOOO	PREDICT	PREDICT	PREDICT
PRESENT	N, V, A		PRES	PR	PRES
PRESENTATION	N		PRES	PR	PRES
PRESERVES	N, V		PRESERV	PRESERV	PRESERV
PREVIOUSLY	A		PREVI	PREV	PREV
PRINCIPAL	N, A		PRINC	PRINCIP	PRINCIP
PRINTOUT	N		PRINTO	PRINTOUT	PRINTOUT
PROCEDURE	N		PROC	PROC	PROCEDUR
PROCEDURES	N		PROC	PROC	PROCEDUR
PROCESS	N, V		PROC	PROC	PROCES
PROCESSED	V		PROC	PROC	PROCESS
PROCESSING	V		PROC	PROC	PROCESS
PROGRAMS	N, V		PROGRAM	PROGRAM	PROGRAM
PROPER	N, A		PROP	PROP	PROPER
PROPERTY	N		PROP	PROP	PROPERT
PROPOSED	V		PROPO	PROPO	PROPOS
PROVIDE	V		PROV	PROVIDE	PROV
PUNCH	N, V		PUNCH	PUNCH	PUNCH
PURPORT	N, V		PURPORT	PURPORT	PURPORT
PURPORTS	N, V		PURPORT	PURPORT	PURPORT
PURPOSE	N, V		PURPO	PURPOSE	PURPOS
PURPOSED	V		PURPO	PURPO	PURPOS
PURPOSEFUL	A		PURPO	PURPOSE	PURPOS
PURPOSEFULLY	A		PURPO	PURPOSE	PURPOS
PURPOSEFULNESS	N		PURPO	PURPOSE	PURPOSE
PURPOSEFULNESSES	N		PURPO	PURPOSE	PURPOSEFUL
PURPOSELESS	A		PURPO	PURPOSE	PURPOS
PURPOSELESSLY	A		PURPO	PURPOSE	PURPOSE
PURPOSELESSNESS	N		PURPO	PURPOSE	PURPOSE
PURPOSELESSNESSES	N		PURPO	PURPOSE	PURPOSELESS
PURPOSELY	A		PURPO	PURPOSE	PURPOS
PURPOSER	N		PURPO	PURPO	PURPOSER
PURPOSERS	N		PURPO	PURPO	PURPOSER
PURPOSES	N, V		PURPO	PURPO	PURPOS
PURPOSING	V		PURPO	PURPO	PURPOS
PURPOSIVE	A		PURPO	PURPO	PURPOS
PURPOSIVELY	A		PURPO	PURPO	PURPOS
PURPOSIVENESS	N		PURPO	PURPOS	PURPOS
PURPOSIVENESSES	N		PURPO	PURPOS	PURPOSIVE
PUSH	N, V		PUSH	PUSH	PUSH

QUALIFY	V	QUAL	QUALI	QUALIF
QUERIES	N, V	QUER	QU	QUER
QUERY	N, V	QUER	QU	QU
RANKING	V	RANK	RANK	RANK
REAL-TIME	N	REAL-TIM	REAL-TIME	REAL-TIM
REASONABLY	A	REAS	RE	REASON
REASONING	N, V	REAS	RE	REASON
RECALL	N, V	RECALL	RECALL	RECAL
RECOGNITION	N	RECOGN	RECOGN	RECOGNIT
RECORD	N, V	RECOR	RECORD	RECORD
REDUCIBILITY	N	REDUCIBIL	REDUC	REDUC
REFERENCE	N, V	REFER	REF	REFER
REFERENCES	N, V	REFER	REF	REFER
REGIONAL	A	REGI	REG	REG
REJECT	N, V	REJECT	REJECT	REJECT
REJECTABLE	A	REJECT	REJECT	REJECT
REJECTABLY	A	REJECT	REJECT	REJECT
REJECTAMENTA	N	REJECTAMENTA	REJECT	REJECTAMENT
REJECTANEOUS	A	REJECTAN	REJECT	REJECTAN
REJECTANEOUSLY	A	REJECTAN	REJECT	REJECTAN
REJECTED	V	REJECT	REJECT	REJECT
REJECTER	N	REJECT	REJECT	REJECTER
REJECTERS	N	REJECT	REJECT	REJECTER
REJECTING	V	REJECT	REJECT	REJECT
REJECTION	N	REJECT	REJECT	REJECT
REJECTIONS	N	REJECT	REJECT	REJECT
REJECTMENT	N	REJECT	REJECT	REJECTM
REJECTMENTS	N	REJECT	REJECT	REJECTMENT
REJECTS	N, V	REJECT	REJECT	REJECT
RELATION	N	RELAT	REL	REL
RELATIONS	N	RELAT	REL	REL
RELATIVELY	A	RELAT	RELAT	REL
RELEVANT	A	RELEV	RELEV	RELEV
REMAINDER	N, V	REMAIND	REMAIND	REMAINDER
REPLACE	V	REPL	REPLACE	REPLAC
REPRESENTED	V	REPR	REPR	REPRES
REPRESENTS	V	REPR	REPR	REPRESENT

REQUEST	N, V	A	REQU	REQU	REQUEST
REQUESTED	V	V	REQUEST	REQU	REQUEST
REQUESTER	N	O	REQUEST	REQU	REQUESTER
REQUESTERS	N	N	REQUEST	REQU	REQUESTER
REQUESTING	V	V	REQUEST	REQU	REQUEST
REQUESTS	N, V	O	REQUEST	REQU	REQUEST
REQUIRABLE	A	A	REQU	REQUIR	REQUIR
REQUIRABLY	A	A	REQU	REQUIR	REQUIR
REQUIRE	V	O	REQU	REQUIRE	REQUIR
REQUIRED	V	V	REQU	REQUIR	REQUIR
REQUIREMENT	N	N	REQU	REQUIRE	REQUIR
REQUIREMENTS	N	N	REQU	REQUIRE	REQUIREMENT
REQUIRER	N	O	REQU	REQUIR	REQUIRER
REQUIRERS	N	N	REQU	REQUIR	REQUIRER
REQUIRES	V	O	REQU	REQUIR	REQUIR
REQUIRING	V	V	REQU	REQUIR	REQUIR
REQUISITE	N, A	O	REQU	REQUI	REQUISIT
REQUISITELY	A	A	REQU	REQUI	REQUISIT
REQUISITENESS	N	N	REQU	REQUI	REQUIS
REQUISITENESSES	N	N	REQU	REQUIS	REQUISITE
REQUISITES	N	O	REQU	REQUISIT	REQUISIT
REQUISITION	N, V	N	REQU	REQUI	REQUISIT
REQUISITIONED	V	V	REQU	REQUI	REQUISIT
REQUISITIONING	V	V	REQU	REQUI	REQUISIT
REQUISITIONIST	N	U	REQUISITIONIST	REQUI	REQUISIT
REQUISITIONISTS	N	O	REQUISITIONIST	REQUI	REQUISITION
REQUISITIONS	N, V	N	REQU	REQUI	REQUISIT
REQUISITOR	N	O	REQU	REQUISIT	REQUISIT
REQUISITORIES	N	N	REQU	REQUISIT	REQUISITOR
REQUISITORILY	A	A	REQUISITORI	REQUISIT	REQUISITOR
REQUISITORS	N	N	REQU	REQUISIT	REQUISITOR
REQUISITORY	A, N	N	REQU	REQUISIT	REQUISITOR
-		O	-	-	-
RESULT	N, V	O	RESULT	RESULT	RESULT
RESULTED	V	V	RESULT	RESULT	RESULT
RESULTING	V	V	RESULT	RESULT	RESULT
RESULTS	N, V	O	RESULT	RESULT	RESULT
-		O	-	-	-
RETRIEVED	V	V	RETRIEV	RETRIEV	RETRIEV
-		O	-	-	-
REVISED	V	V	REVI	REVI	REVIS
-		O	-	-	-
RIGHT	N, V, A, O	O	RIGHT	RIGHT	RIGHT
-		O	-	-	-
ROOM	N, V	O	ROOM	ROOM	ROOM
-		O	-	-	-
ROOT	N, V	O	ROOT	ROOT	ROOT
-		O	-	-	-
RULES	N, V	O	RULE	RUL	RUL
-		O	-	-	-
SCHEDULE	N, V	O	SCHEDUL	SCHEDULE	SCHEDUL
-		O	-	-	-
SEARCH	N, V	O	SEARCH	SEARCH	SEARCH
-		O	-	-	-
SECOND	N, V, A	O	SECOND	SECOND	SECONS

SECTION	N, V	ONO	SECT	SECT	SECT
SEEN	V	OOO	SEEN	SE	SEEN
SENTENCE	N, V	ONO	SENT	SENT	SENT
SET	N, V	OOO	SET	SET	SES
SETS	N, V	OOO	SETS	SET	SET
SHEW	V	OOO	SHEW	SHEW	SHEW
SHEWING	V	OVO	SHEW	SHEW	SHEW
SHEWN	V	OOO	SHEWN	SHEWN	SHEWN
SHEWS	V	OOO	SHEW	SHEW	SHEW
SHOW	N, V	OOO	SHOW	SHOW	SHOW
SHOWED	V	OVO	SHOW	SHOW	SHOW
SHOWER	N, V	ONO	SHOW	SHOW	SHOWER
SHOWERS	N, V	ONA	SHOW	SHOW	SHOWER
SHOWILY	A	OAA	SHOWI	SHOWI	SHOW
SHOWINESS	N	ONN	SHOWI	SHOWI	SHOW
SHOWINESSES	N	ONN	SHOWI	SHOWI	SHOWI
SHOWING	N, V	OVO	SHOW	SHOW	SHOW
SHOWINGS	N	ONO	SHOW	SHOW	SHOW
SHOWISH	A	OOA	SHOW	SHOW	SHOW
SHOWISHLY	A	OAA	SHOW	SHOW	SHOWISH
SHOWN	V	OOO	SHOWN	SHOWN	SHOWN
SHOWS	V	OOO	SHOW	SHOW	SHOW
SHOWY	A	OOO	SHOW	SHOW	SHOW
SIGNIFICANCE	N	ONO	SIGN	SIGNI	SIGNIF
SIGNS	N, V	OOO	SIGN	SIGN	SIGN
SIMILARITY	N	ONO	SIMIL	SIMIL	SIMIL
SIMPLE	N, A, V	OVA	SIMP	SIMPLE	SIMPL
SIMPLED	V	OVN	SIMP	SIMPL	SIMPL
SIMPLENESS	N	NNN	SIMP	SIMPLE	SIMPL
SIMPLENESSES	N	NNN	SIMP	SIMPLE	SIMPLE
SIMPLER	A, N	NNN	SIMP	SIMPL	SIMPLER
SIMPLERS	N	NNN	SIMP	SIMPL	SIMPLER
SIMPLES	N, V	NVO	SIMP	SIMPL	SIMPL
SIMPLESSE	N	OOO	SIMP	SIMPLESSE	SIMPLESS
SIMPLESSES	N	OOA	SIMP	SIMP	SIMPLESS
SIMPLEST	A	OAA	SIMPL	SIMPL	SIMPLEST
SIMPLETON	N	ONN	SIMPLET	SIMPLET	SIMPLET
SIMPLETONS	N	NNN	SIMPLET	SIMPLET	SIMPLETON
SIMPLEX	A, N	NNO	SIMPL	SIMPLEX	SIMPLEC
SIMPLEXES	N	OOO	SIMPLEX	SIMPLEX	SIMPLEX
SIMPLICITER	A	OON	SIMP	SIMPLICIT	SIMPLICITER
SIMPLICITIES	N	NNN	SIMP	SIMPL	SIMPLIC
SIMPLICITY	N	NNN	SIMP	SIMPL	SIMPL
SIMPLIFICATION	N	NNN	SIMPL	SIMPL	SIMPLIFIC
SIMPLIFICATIONS	N	NNO	SIMPL	SIMPLI	SIMPLIFIC
SIMPLIFIED	V	OVO	SIMPL	SIMPL	SIMPLIF
SIMPLIFIES	V	OVO	SIMPL	SIMPLI	SIMPLIF

SIMPLIFY	V	SIMPL	SIMPLI	SIMPLIF
SIMPLIFYING	V	SIMPL	SIMPLI	SIMPLIF
SIMPLING	V	SIMP	SIMPL	SIMPL
SIMPLIST	N	SIMPL	SIMPL	SIMPL
SIMPLISTIC	A	SIMPL	SIMPL	SIMPL
SIMPLISTICALLY	A	SIMPL	SIMPL	SIMPL
SIMPLISTS	N	SIMPL	SIMPL	SIMPL
SIMPLY	A	SIMP	SIMP	SIMP
SITUATION	N	SITU	SITU	SITU
SO-CALLED	A	SO-C	SO-CALL	SO-CALL
SORT	N, V	SORT	SORT	SORT
SPECIFIC	A	SPEC	SPECI	SPECIF
SPECIFICALLY	A	SPEC	SPECIF	SPECIF
SPECIFICATIONS	N	SPEC	SPECI	SPECIFIC
SPECIFIED	V	SPEC	SPEC	SPECIF
SPECIFY	V	SPEC	SPECI	SPECIF
STATEMENT	N	STAT	ST	STAT
STATES	N, V	STAT	ST	ST
STATISTICAL	A	STATIST	STAT	STAT
STATISTICS	N	STAT	STAT	STAT
STEM	N, V	STEM	STEM	STEM
STEPS	N, V	STEP	STEP	STEP
STOP	N, V	STOP	STOP	STOP
STRATEGY	N	STRATEG	STRATEG	STRATEG
STRUCTURE	N	STRUCT	STRUCT	STRUCTUR
STRUCTURES	N	STRUCT	STRUCT	STRUCTUR
SUBJECT	N, V, A	SUBJECT	SUBJECT	SUBJECT
SUBSET	N	SUBSET	SUBSET	SUBSES
SUBSETS	N	SUBSET	SUBSET	SUBSET
SUFFER	V	SUFF	SUFF	SUFFER
SUFFIX	N, V	SUFF	SUFFIX	SUFFIC
SUFFIXED	V	SUFFIX	SUFFIX	SUFFIX
SUFFIXES	N, V	SUFFIX	SUFFIX	SUFFIX
SUFFIXING	V	SUFFIX	SUFFIX	SUFFIX
SUFFIXION	N	SUFFIX	SUFFIX	SUFFIX
SUFFIXIONS	N	SUFFICT	SUFFIX	SUFFIX
SUFFIXMENT	N	SUFFIX	SUFFIX	SUFFIXM
SUFFIXMENTS	N	SUFFIX	SUFFIX	SUFFIXMENT
SUITABLE	A	SUIT	SUIT	SUIT

SUM	N, V	SUM	SUM	SUM	SUM
SUPER-SCRIPTS	N	SUPER-SCRI	SUPER-SCRIPT	SUPER-SCRIPT	SUPER-SCRIPT
SURFACE	N, V	SURF	SURFACE	SURFAC	SURFAC
SYNTACTIC	A	SYNTACT	SYNTACT	SYNTACT	SYNTACT
SYNTOL	O	SYNTOL	SYNTOL	SYNTOL	SYNTOL
SYSTEM	N	SYSTEM	SYSTEM	SYSTEM	SYSTEM
SYSTEMS	N	SYSTEM	SYSTEM	SYSTEM	SYSTEM
TABLE	N, V	TABL	TABLE	TABL	TABL
TAKEN	V	TAKEN	TAK	TAK	TAK
TAPE	N, V	TAPE	TAPE	TAP	TAP
TASK	N, V	TASK	TASK	TASK	TASK
TENACITY	N	TENAC	TENAC	TEN	TEN
TEND	V	TEND	TEND	TENS	TENS
TERM-DOCUMENT	N	TERM-DOCU	TERM-DOCU	TERM-DOCUM	TERM-DOCUM
TERM	N, V	TERM	TERM	TERM	TERM
TERMINATE	V, A	TERMIN	TERMIN	TERM	TERM
TERMS	N, V	TERM	TERM	TERM	TERM
TEXT	N	TEXT	TEXT	TEXT	TEXT
TEXTS	N	TEXT	TEXT	TEXT	TEXT
THEOREM	N	THEOREM	THEOREM	THEOREM	THEOREM
THEOREMS	N	THEOREM	THEOREM	THEOREM	THEOREM
THESAURUS	N	THESAURU	THESAURU	THESAURUS	THESAURUS
THESAURUSES	N	THESAURU	THESAURU	THESAURUS	THESAURUS
TIME	N, V	TIME	TIME	TIM	TIM
TRANSFER	N, V	TRANSF	TRANSF	TRANSFER	TRANSFER
TRANSFORMED	V	TRANSFORM	TRANSFORM	TRANSFORM	TRANSFORM
TREE	N, V	TREE	TREE	TRE	TRE
TREELIKE	A	TREE	TREE	TREELIK	TREELIK
TYPE	N, V	TYPE	TYPE	TYP	TYP
TYPES	N, V	TYPE	TYP	TYP	TYP
UPDATE	V	UPDAT	UPD	UPD	UPD

USE	N, V	0	USE	USE	US
USED	V	0	USED	US	US
USER	N	0	USER	US	USER
USING	V	0	USING	US	USING
USUALLY	A	A	USUAL	USU	USU
-		0	-	-	-
VALUE	N, V	0	VALU	VALUE	VALU
-		0	-	-	-
VARIETY	N	0	VARIET	VARIE	VARIET
VARIOUS	A	A	VARI	VAR	VAR
-		0	-	-	-
VECTORS	N, V	N	VECT	VECT	VECTOR
-		0	-	-	-
VISIT	N, V	0	VISIT	VISIT	VISIT
-		0	-	-	-
WANTED	V	V	WANT	WANT	WANT
-		0	-	-	-
WAYS	N, V	0	WAYS	WA	WAY
-		0	-	-	-
WEIGHTS	N, V	0	WEIGHT	WEIGHT	WEIGHT
-		0	-	-	-
WORD	N, V	0	WORD	WORD	WORD
WORDS	N, V	0	WORD	WORD	WORD
-		0	-	-	-
WRITE	V	0	WRIT	WR	WRIT
-		0	-	-	-

Tafel III, 18

Anhang IV Tabellen zur Berechnung von Masszahlen für die Wortstammbildung für spezielle Regelsätze (siehe 4.3)

Ergebnisse und Masszahlen bei Anwendung der Regeln von EURATOM:

Mächtigkeit der Wortklassen	Anzahl der Wortklassen	Anzahl der Wörter	Anzahl der Klassen mit i Stämmen i = 1	2	3	4	5	Anzahl der Stämme	Anzahl der nicht eindeutig repräsentierten Klassen
1	227	227	227					227	6
2	58	116	50	8				66	4
3	7	21	5	2				9	1
4	3	12	2	1				4	
5	2	10	1				1	6	
6	2	12		2				4	
7	1	7				1		4	
8	1	8			1			3	1
12	1	12			1			3	
14	1	14		1				2	
15	2	30	1		1			4	
18	1	18					1	5	
21	3	63		1	2			8	
29	1	29				1		4	
32	1	32				1		4	
37	1	37					1	5	
	312	648	286	15	5	3	3	358	12
	= K	= A						= S	= K-U

$$M_1 = \frac{648 - 358}{648 - 312} = \frac{290}{336} = 0,86$$

$$M_2 = \frac{312 - 12}{312} = \frac{300}{312} = 0,96$$

$$M = M_1 \cdot M_2 = \frac{87000}{104832} = 0,83$$

Tafel IV, 1

Ergebnisse und Masszahlen bei Anwendung der SM-Regeln:

Mächtigkeit der Wortklassen	Anzahl der Wortklassen	Anzahl der Wörter	Anzahl der Klassen mit i Stämmen i = 1	2	3	4	5	6	7	Anzahl der Stämme	Anzahl der nicht eindeutig repräsentierten Klassen
1	227	227	227							227	6
2	58	116	46	12						70	1
3	7	21	4	3						10	1
4	3	12	1	2						5	
5	2	10			2					6	
6	2	12	1	1						3	
7	1	7			1					3	
8	1	8	1							1	
12	1	12		1						2	
14	1	14	1							1	
15	2	30	1			1				5	
18	1	18					1			5	
21	3	63			1	2				11	
29	1	29							1	7	
32	1	32						1		6	
37	1	37					1			5	
	312 = K	648 = A	282	19	4	3	2	1	1	367 = S	8 = K-U

$$M_1 = \frac{648 - 367}{648 - 312} = \frac{281}{336} = 0,84$$

$$M_2 = \frac{312 - 8}{312} = \frac{304}{312} = 0,97$$

$$M = M_1 \cdot M_2 = \frac{85424}{104832} = 0,81$$

Tafel IV, 2

Ergebnisse und Masszahlen bei Anwendung der Regeln von LOVINS:

Mächtigkeit der Wortklassen	Anzahl der Wortklassen	Anzahl der Wörter	Anzahl der Klassen mit i Stämmen i = 1	2	3	4	5	6	7	8	10	12	Anzahl der Stämme	Anzahl der nicht eindeutig repräsentierten Klassen
1	227	227	227										227	8
2	58	116	38	20									78	2
3	7	21	5	2									9	
4	3	12	2	1									4	
5	2	10		1		1							6	
6	2	12			2								6	
7	1	7			1								3	
8	1	8			1								3	
12	1	12			1								3	
14	1	14			1								3	
15	2	30					1	1					11	
18	1	18							1				7	
21	3	63						1	1	1			21	
29	1	29										1	12	
32	1	32									1		10	
37	1	37					1						5	
	312 = K	648 = A	272	24	6	1	2	2	2	1	1	1	408 = S	10 = K-U

$$M_1 = \frac{648 - 408}{648 - 312} = \frac{240}{336} = 0,71 \quad ====$$

$$M_2 = \frac{312 - 10}{312} = \frac{302}{312} = 0,97 \quad ====$$

$$M = M_1 \cdot M_2 = \frac{72480}{104832} = 0,69 \quad ====$$

Tafel IV, 3

Anhang V Das Programm SUFFANAL

1 Möglichkeiten des Programms

Das Programm SUFFANAL wurde erstellt im Jahre 1967 von H. FANGMEYER, EURATOM (Cetis), Ispra.

Es ermöglicht eine automatische Herstellung von Stammwörterbüchern und die Bestimmung von Wortarten.

An der ursprünglichen Version des Programms wurden verschiedene Änderungen und Korrekturen vorgenommen, sodass alle in 3.1 beschriebenen Regelsätze mit dem Programm kompatibel sind.

Die Möglichkeiten des Programms entsprechen der in 2 hergeleiteten Abbildungsfunktion.

2 Benutzeranleitung

2.1 Externe Parameter

a) Beschreibung

Der Ablauf des Programms wird von Steuerkarten gelenkt, die acht Parameter enthalten:

1. INP
2. OUT
3. REP
4. LEN
5. TRU
6. ORD
7. PRI
8. SUP

1. Der Parameter INP (INPut) bestimmt die Eingabeeinheit für die zu analysierenden Wörter.
2. Der Parameter OUT (OUTput) bestimmt die Einheit für eine Ausgabe in maschinenlesbarer Form.
   (Bei OUT=0 erfolgt keine Ausgabe.)

3. Die maximale Anzahl der Iterationen wird durch den Parameter REP (REPetition) festgelegt.

4. Mit dem Parameter LEN (LENgth) kann die minimale Anzahl von Zeichen der Wortstämme festgesetzt werden.
   (Würde diese Mindestlänge bei der Anwendung einer Regel unterschritten, so wird die entsprechende Regel nicht angewandt.)

5. Der Parameter TRU (TRUncation) erlaubt es, von jedem nicht analysierten Wort eine feste, wählbare Anzahl von Zeichen zu unterdrücken. (Dabei wird jedoch die durch den Parameter LEN festgesetzte Mindestlänge eingehalten.)

6. Der Parameter ORD (ORDer) ermöglicht eine Wahl des Suchalgorithmus' in der Liste der Regeln.
   (Bei ORD≠1 wird in jedem Iterationsschritt unter allen Regeln nach passenden gesucht.
   ORD=1 bestimmt einen speziellen Suchalgorithmus: Nach jedem Iterationsschritt werden nur noch solche Regeln berücksichtigt, die in der Liste der Regeln nach der zuvor angewandten Regel aufgeführt sind. Dies bedeutet u.a., dass eine Regel höchstens einmal angewandt werden kann.)

7. Der Parameter PRI (PRInt) kontrolliert die Ausgabe der analysierten Wörter.
   (PRI=0 bedeutet Unterdrückung, PRI≠0 bewirkt eine Ausgabe durch den Drucker.)

8. Der Parameter SUP (SUPpression) kontrolliert die Ausgabe der nicht analysierten Wörter.
   (SUP=1 bedeutet Unterdrückung, SUP≠1 bewirkt eine Ausgabe durch den Drucker.)

b) Codierung

Die Codierung der acht externen Parameter geschieht wie in folgendem Beispiel:

```
$/ TRU=      0    ,ORD=      0    ,PRI=      4    ,SUP=      4
$/ INP=      9    ,OUT=      0    ,REP=      3    ,LEN=      2    ,C
```

Die Reihenfolge der Parameter ist beliebig. Es müssen jedoch alle Parameter definiert und das Format eingehalten werden. Die Symbole $/ kennzeichnen die Parameterkarten.

Format:

	1. Parameterkarte Spalte			2. Parameterkarte Spalte	
	1	$		1	$
	2	/		2	/
	3	blank		3	blank
1. Param.	4-6	Parametername	5. Param.	4-19	analog zum 1. Parameter
	7	=			
	8-14	blank			
	15	Zahlenwert des Parameters			
	16-19	blank			
	20	,		20	,
2. Param.	21-36	analog zum 1. Parameter	6. Param.	21-36	-"-
	37	,		37	,
3. Param.	38-53	-"-	7. Param.	38-53	-"-
	54	,		54	,
4. Param.	55-70	-"-	8. Param.	55-70	-"-
	71	,		71-72	blank
	72	C			

c) Fehlermeldungen [1]

$ THE IDENTIFICATION FOR DATACARDS IS MISSED.
THE CARD WILL NOT BE INTERPRETED.

D. h.: Entweder ist die Kennzeichnung der Parameterkarten für die externen Parameter nicht korrekt (siehe b)
oder die Anordnung des Kartenstapels ist nicht vorschriftsgemäss (siehe AV2. 5).

Verhalten des Programms:
Das Programm läuft zunächst weiter, kommt jedoch dann zum Halt, da die vorgeschriebene Anzahl von acht externen Parametern oder andere Daten nicht gefunden werden.

$ .... IS NOT A PARAMETER FOR THIS PROGRAMM. IT WILL BE IGNORED.

D. h.: Die richtig gekennzeichnete Parameterkarte enthält einen ungültigen Parameternamen.

Verhalten des Programms:
Das Programm läuft zunächst weiter, kommt jedoch dann zum Halt, da die vorgeschriebene Anzahl von acht externen Parametern nicht gefunden wird.

$ AN ERROR HAS BEEN DETECTED READING THE PARAMETER CARD.

D. h.: In einem numerischen Feld der Parameterkarte - Zahlenwert des Parameters - steht ein Alpha-Zeichen.

Verhalten des Programms:
Stop

$.... PARAMETERS ARE MISSED.

D. h.: Entweder wurden nicht alle acht externen Parameter definiert
oder ein oder mehrere Parameternamen wurden falsch definiert
oder es fehlt auf der 1. Parameterkarte das Zeichen für "continue" (C).

Verhalten des Programms:
Stop

---

[1] Alle Meldungen des Programms sind gekennzeichnet durch ein vorangehendes $-Zeichen.

2.2 Regeln

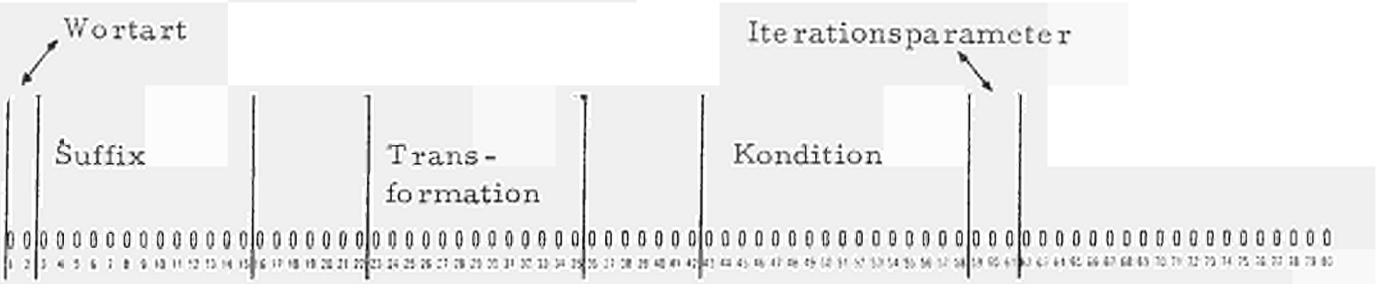
a) Anzahl der Regeln

Maximal 500 Regeln können definiert werden. Die Regeln werden als Datenkarten eingelesen. Jede Karte enthält eine Regel.

b) Aufbau und Codierung einer Regel [1)]

Eine Regel enthält folgende Parameter:

Wortart
Suffix
Transformation
Kondition
Iterationsparameter



Wortart: (Spalten 1 und 2 auf einer Regelkarte)

Das Feld kann jede Kombination von zwei alphanumerischen Zeichen enthalten, einschliesslich Leerzeichen.

Wird für ein Textwort eine passende Regel gefunden, so wird die Regel angewandt und dem Wort die entsprechende Wortart zugewiesen.
Die Bestimmung der Wortart erfolgt im 1. Iterationsschritt. Falls in weiteren Iterationsschritten noch passende Regeln gefunden werden, so haben diese Regeln auf die Wortartbestimmung keinen Einfluss mehr.

---

1) Die Codierung der EURATOM-Regeln findet sich im Anhang V (Programmprotokoll).

Suffix: (Spalten 3 bis 15 auf einer Regelkarte)

Das Feld kann jede Kombination von maximal 13 alphanumerischen Zeichen enthalten.

Ein Textwort wird mit dieser Zeichenkette - Suffix - verglichen, und zwar von rechts nach links. Stimmt ein Suffix vollständig mit der Wortendung überein, so wird das Suffix von dem Wort abgetrennt, falls die Regel keine Einschränkungen enthält. (Einschränkungen sind Konditionen und/oder der Parameter in den Spalten 59 bis 61, siehe unten.)

Transformation: (Spalten 23 bis 35 auf einer Regelkarte)

Das Feld kann folgende Zeichenketten enthalten:

1. Beliebige Zeichenkette, jedoch kein Minus-Zeichen am Anfang.
2. Minus-Zeichen, danach eine Ziffer von 1 bis 5.

Im ersten Fall wird die entsprechende Zeichenfolge an das um das Suffix gekürzte Wort angehängt, jedoch nur bis zu einer maximalen Länge von 29 Zeichen.
Im zweiten Fall werden von dem um das Suffix gekürzten Wort noch die entsprechende Anzahl von Zeichen abgetrennt.

Kondition: (Spalten 43 bis 58 auf einer Regelkarte)

In den Spalten 43, 45, 47, ..., 57 des Feldes stehen Plus-, Minus- oder Leerzeichen. Die anderen Spalten können alle sonstigen Zeichen enthalten.

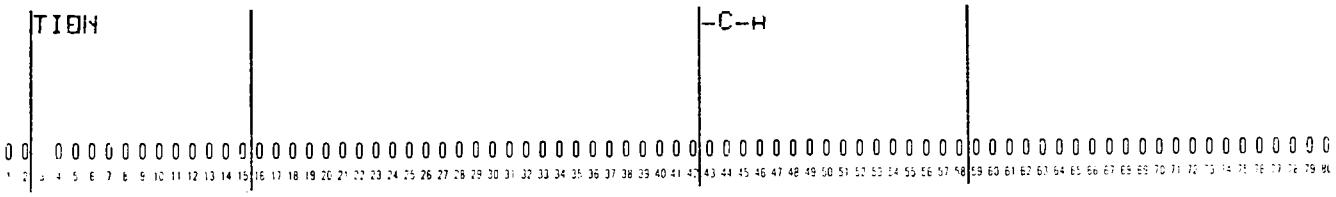
Das Minuszeichen symbolisiert ein Verbot; Plus- oder Leerzeichen symbolisieren eine Forderung.

Die Bedingung bezieht sich stets auf ein oder mehrere Zeichen des Stammes, und zwar gilt die erste Forderung des Feldes für das letzte Zeichen des Stammes usw.

Bei einem Plus- oder Leerzeichen muss das folgende Zeichen mit

dem entsprechenden Zeichen des Stammes übereinstimmen, bei einem Minus-Zeichen darf keine Übereinstimmung bestehen.

Eine Forderung oder ein Verbot kann sich auch auf eine Gruppe von Zeichen beziehen, die durch eine Ziffer von 1 bis 9 symbolisiert wird (siehe V2.3, Zeichengruppen).
Beispiel:



Das Suffix TION wird nur dann von einem Wort mit dieser Endung abgetrennt, wenn es nicht auf AC folgt, d.h. die Regel wird nicht angewandt auf ein Wort wie ACTION.

Iterationsparameter: (Spalten 59 bis 61 einer Regelkarte)

Der Parameter kann vier Zahlenwerte annehmen. Diese steuern die Iteration wie folgt:

Parameter	Beim i-ten Iterationsschritt (i > 1): Soll die Regel angewandt werden?	Beim ersten Iterationsschritt: Soll weiter iteriert werden?
010	nein	nein
011	nein	ja
012	ja	nein
013	ja	ja

(Die maximale Anzahl der Iterationsschritte wird durch den Parameter REP bestimmt.)

c) Anordnung der Regeln

Die Anordnung der Regeln wird durch den externen Parameter ORD bestimmt:

ORD≠1: Die Regeln werden nach dem End-Zeichen der Suffixe geordnet:
1. Buchstaben (A, . . . , Z)
2. Ziffern (0, . . . , 9)
3. Sonderzeichen (', . . . , ¢)

Maximal 30 verschiedene End-Zeichen sind vorgesehen. Die Ordnung innerhalb einer Gruppe von Suffixen mit gleichen End-Zeichen ist beliebig.

ORD=1: Die Regeln können beliebig angeordnet werden.

Bemerkung: Die Anordnung innerhalb einer Gruppe von Suffixen mit gleichen End-Zeichen bei ORD≠1 und die Anordnung aller Regeln bei ORD=1 richtet sich nach der Zielsetzung. Man beachte, dass jeweils die zuerst gefundene Regel angewandt wird. Dies erfordert eine "Hierarchie" innerhalb der Regeln, die der in 2.4 eingeführten Ordnungsrelation entspricht.

Beispiel: Die folgende Anordnung gewährleistet, dass immer das längste Suffix von einem Wort abgetrennt wird:

FICATION
ATION
ION

d) Fehlermeldungen

ϕ THE DIMENSION FOR INPUT DATA MUST BE REDEFINED.

D. h.: Die Anzahl der Regeln ist grösser als 500; der interne Parameter L1 und die Dimensionen der folgenden Felder müssen neu definiert werden:

```
KLASSI  (L1)
ISUFFI  (L1,13)
ISTEM   (L1,13)
ISTSUF  (L1,16)
IHIE    (L1)
IHI     (L1)
IHII    (L1)
ICOUNT  (L1+1)
LENG    (L1)
```

Verhalten des Programms:
Stop

$ RULE NUMBER.... AND THE FOLLOWING WITH THE FINAL LETTER.. ARE NOT IN CORRECT ALPHABETICAL SEQUENCE.

D.h.: Als externer Parameter für die Anordnung der Regeln wurde ORD≠1 gewählt; die Regeln sind jedoch nicht korrekt angeordnet.

Verhalten des Programms:
Stop

$ L7 AND DIMENSION OF ISUANF AND LETTER MUST BE REDEFINED.
$ (OR THERE MAY BE TOO MANY RULES NOT IN CORRECT ALPHABETICAL SEQUENCE.)

D.h.: Als externer Parameter für die Anordnung der Regeln wurde ORD≠1 gewählt.
Entweder sind die Regeln nicht korrekt angeordnet
oder die Suffixe haben mehr als 30 verschiedene End-Zeichen.
Im zweiten Fall müssen der interne Parameter L7 und die Dimensionen der Felder ISUANF und LETTER neu definiert werden.

Verhalten des Programms:
Stop

$ RULE.... IS NOT WELL DEFINED. COLUMN "24" MAY ONLY CONTAIN A NUMBER BETWEEN 1 AND 5.

D.h.: Spalte 23 enthält ein Minuszeichen; dann muss aber in Spalte 24 eine Ziffer von 1 bis 5 folgen.

Verhalten des Programms:
Stop

$ THE STEM OF THE FOLLOWING WORD EXCEEDS THE RANGE FORSEEN.

D.h.: Der Wortstamm wird nach der Transformation länger als 29 Zeichen.

Verhalten des Programms:
Continue

2.3 Zeichengruppen

Es können bis zu neun Zeichengruppen beliebig definiert werden, von denen jede maximal 30 Zeichen enthalten kann. Diese Zeichen werden als Daten von Lochkarten eingelesen.

In der 2. Spalte einer solchen Karte steht die Nummer der Zeichengruppe, in der 5. Spalte das Zeichen, das der Gruppe zugeordnet werden soll. Die Reihenfolge der Karten ist beliebig.

Beispiel:

```
1  A
1  E          Damit sind drei Zeichengruppen definiert:
1  I
3  &          1 := A∨E∨I∨O
2  X          2 := X∨Y
1  O          3 := &∨§
3  §
2  Y
```

In den Regeln können diese Zeichengruppen innerhalb der Felder für Konditionen und Transformationen zitiert werden.

Falls den Zeichengruppen mehr als 30 Zeichen zugeteilt werden sollen, so müssen der interne Parameter L5 und die Dimensionen der folgenden Felder neu definiert werden:

```
LIST1 (L5)
LIST2 (L5)
LIST3 (L5)
LIST4 (L5)
LIST5 (L5)
LIST6 (L5)
LIST7 (L5)
LIST8 (L5)
LIST9 (L5)
```

2.4 Format

Dieses variable Format bezieht sich auf die durch den externen Parameter INP bestimmte Eingabeeinheit für die zu analysierenden Wörter.

Beispiel:

```
(23X,29A1,81X)
```

2.5 Datenstapel

Die Lochkarten mit den Daten müssen folgendermassen angeordnet werden:

	Datentyp	Anzahl der Karten
1.	externe Parameter (siehe V2.1)	2
2.	Zeichengruppen (siehe V2.3)	maximal 9 x 30
3.	Leerkarte	1
4.	Eingabeformat (siehe V2.4)	1
5.	Regeln (siehe V2.2)	maximal 500

2.6 Interne Parameter

Mit den internen Parametern L2, L3, L4 und L6 können die Dimensionen von Feldern verändert werden.

Der Parameter L2 muss mit den Dimensionen der folgenden Felder übereinstimmen:

1. Feld für die Suffixe auf den Regelkarten.
   - ISUFFI -
2. Feld für die Transformationen auf den Regelkarten.
   - ISTEM -
3. Feld für die Endungskette, die von einem Wort abgetrennt wird.
   - NEWFI -

Das Maximum für L2 ist 20.

Der Parameter L3 muss mit der Dimension des Feldes für die zu analysierenden Wörter übereinstimmen.

- IWORD und IWOR -

Der Parameter L4 muss mit der Dimension des Feldes für die gebildeten Wortstämme übereinstimmen.

- IREDUC -

Der Parameter L6 muss mit der Dimension des Feldes für die Konditionen auf einer Regelkarte übereinstimmen.

- ISTSUF -

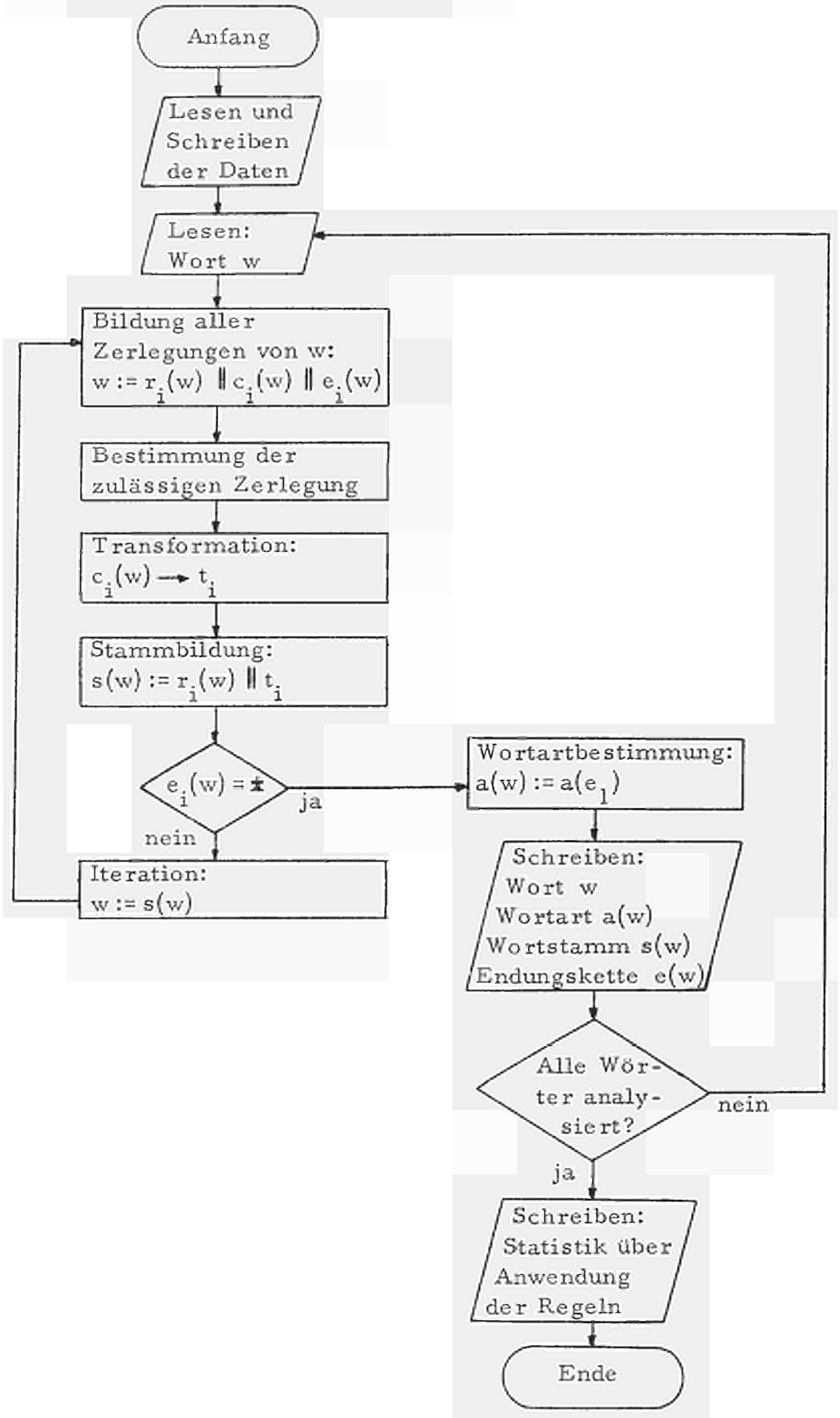
Das Maximum für L6 ist 16.

Folgende Programmänderungen sind erforderlich, falls diese Parameter neu definiert werden:

Parameter	Dimensionen	Formate
L2	ISUFFI (500, L2) ISTEM (500, L2) NEWFI (L2)	1000, 1020, 1040, 1180, 1290
L3	IWORD (L3) IWOR (L3)	1035, 1180, 1290, 1310 [1)]
L4	IREDUC (L4)	1020, 1035, 1180, 1290
L6	ISTSUF (500, L6)	1000, 1030, 1040

1) Die Änderung von L3 erfordert ausserdem eine Kontrolle des externen Parameters "Format" (siehe V2.4).

3 Vereinfachtes Flussdiagramm



4 Technische Daten

Das Programm ist geschrieben in FORTRAN IV. Es wurde getestet und angewandt auf einem IBM 370/165 System.

Für eine Analyse von 1000 Wörtern wurden auf der genannten Anlage nicht mehr als 0,4 Minuten CPU-Zeit benötigt. Die vorliegende Version beansprucht 144 K Bytes Speicherplatz.

Das Programm ist nicht optimisiert; es liesse sich bei einer Reprogrammierung wesentlich verbessern in Bezug auf Speicherplatz und Geschwindigkeit. Andererseits ist ein hoher Grad von Kompatibilität gewährleistet. (Nach geringfügigen Veränderungen konnte mit dem Programm auf einer Telefunken TR 440 gearbeitet werden.)

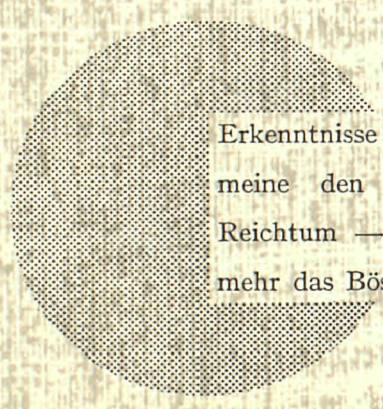
5 Programmprotokoll

Die folgenden Seiten sind Kopien der Listen eines Programmlaufes auf einer IBM 370/165 bei EURATOM (CETIS), Ispra. Es wurden 648 Wörter analysiert mit den in 3.1.1 beschriebenen EURATOM-Regeln.

AN UNSERE LESER

Alle von der Kommission der Europäischen Gemeinschaften veröffentlichten wissenschaftlichen und technischen Berichte werden in der Monatszeitschrift **„euro-abstracts“** angezeigt. Abonnement (1 Jahr: BF 1 025,—) und Probehefte sind erhältlich bei:

Amt für amtliche Veröffentlichungen
der Europäischen Gemeinschaften
Case postale 1003
Luxembourg 1



Erkenntnisse verbreiten ist soviel wie Wohlstand verbreiten — ich meine den allgemeinen Wohlstand, nicht den individuellen Reichtum —, denn mit dem Wohlstand verschwindet mehr und mehr das Böse, das uns aus dunkler Zeit vererbt ist.

Alfred Nobel

VERTRIEBSSTELLEN

Alle von der Kommission der Europäischen Gemeinschaften veröffentlichten Dokumente werden durch das Amt für amtliche Veröffentlichungen bei den untenangegebenen Anschriften zu dem unten angegebenen Preis verkauft. Bei schriftlicher Bestellung bitte die genaue Referenz und den Titel des Dokumentes deutlich angeben.

DEUTSCHLAND (BR)

Verlag Bundesanzeiger
5 Köln 1 — Postfach 108 006
Tel. (0221) 21 03 48
Fernschreiber: Anzeiger Bonn 08 882 595
Postscheckkonto 834 00 Köln

BELGIEN

Moniteur belge — Belgisch Staatsblad
Rue de Louvain 40-42 — Leuvenseweg 40-42
1000 Bruxelles — 1000 Brussel — Tel. 12 00 26
CCP 50-80 — Postgiro 50-80

Nebenstelle:
Librairie européenne — Europese Boekhandel
Rue de la Loi 244 — Wetstraat 244
1040 Bruxelles — 1040 Brussel

DÄNEMARK

J.H. Schultz — Boghandel
Møntergade 19
DK 1116 København K — Tel. 14 11 95

FRANKREICH

Service de vente en France des publications des Communautés européennes — Journal officiel
26, rue Desaix — 75 732 Paris - Cédex 15e
Tel. (1) 306 51 00 — CCP Paris 23-96

GROSSHERZOGTUM LUXEMBURG

Amt für amtliche Veröffentlichungen der Europäischen Gemeinschaften
Case postale 1003 — Luxembourg
Tel. 4 79 41 — CCP 191-90
Compte courant bancaire: BIL 8-109/6003/200

IRLAND

Stationery Office — The Controller
Beggar's Bush
Dublin 4 — Tel. 6 54 01

ITALIEN

Libreria dello Stato
Piazza G. Verdi 10
00198 Roma — Tel. (6) 85 08
CCP 1/2640

NIEDERLANDE

Staatsdrukkerij- en uitgeverijbedrijf
Christoffel Plantijnstraat
's-Gravenhage — Tel. (070) 81 45 11
Postgiro 42 53 00

VEREINIGTES KÖNIGREICH

H.M. Stationery Office
P.O. Box 569
London S.E. 1 — Tel. 01-928 69 77, ext. 365

VEREINIGTE STAATEN VON AMERIKA

European Community Information Service
2100 M Street, N.W.
Suite 707
Washington, D.C., 20 037 — Tel. 296 51 31

SCHWEIZ

Librairie Payot
6, rue Grenus
1211 Genève — Tel. 31 89 50
CCP 12-236 Genève

SCHWEDEN

Librairie C.E. Fritze
2, Fredsgatan
Stockholm 16
Post Giro 193, Bank Giro 73/4015

SPANIEN

Libreria Mundi-Prensa
Castello 37
Madrid 1 — Tel. 275 51 31

ANDERE LÄNDER

Amt für amtliche Veröffentlichungen der Europäischen Gemeinschaften
Case postale 1003 — Luxembourg
Tel. 4 79 41 — CCP 191-90
Compte courant bancaire: BIL 8-109/6003/200

AMT FÜR AMTLICHE VERÖFFENTLICHUNGEN DER EUROPÄISCHEN GEMEINSCHAFTEN
Case postale 1003 — Luxembourg
6298