gefördert vom
Bundesministerium
für Forschung
und Technologie

Verbmobil
Verbundvorhaben

Lexikalische Regeln in der IBM-Basisgrammatik

Stefan Geißler

IBM Informationssysteme GmbH

Vm
Report 20
August 1994

August 1994

Stefan Geißler

IBM Informationssysteme GmbH
Institut für Logik und Linguistik
Vangerowstr. 18
69115 Heidelberg

Tel.: (06221) 59 - 4482
Fax: (06221) 59 - 3200
e-mail: sgeissler@vnet.ibm.com

Gehört zum Antragsabschnitt: 6

Das diesem Bericht zugrundeliegende Forschungsvorhaben wurde mit Mitteln des Bundesministers für Forschung und Technologie unter dem Förderkennzeichen 01 IV 101 G gefördert. Die Verantwortung für den Inhalt dieser Arbeit liegt bei dem Autor.

Inhaltsverzeichnis

Abstact

The IBM Core Grammar makes use of the concept of *Lexical Rules (LRs)* to avoid systematic redundancies in the lexicon. These LRs can be specified in the base formalism STUF in order to model a *virtual lexicon* containing basic as well as derived entries.
We present an implementation of LRs and discuss the merits and shortcomings of our approach. Namely the treatment of empty elements can be combined with LRs applications in a advantageous way. Finally, we suggest some augmentations that will enhance the usefulness of the LR concept for large grammars.

Zusammenfassung

Die IBM Basisgrammatik verwendet *Lexikalische Regeln (LRs)*, um systematische Redundanzen im Lexikon zu vermeiden. Diese LRs können im Basisformalismus STUF spezifiziert werden, um ein *virtuelles Lexikon* zu modellieren, welches sowohl Basiseinträge, als auch abgeleitete Einträge enthält.
Wir stellen eine Implementierung von LRs vor und diskutieren die Vor- und Nachteile dieses Ansatzes. Insbesondere die Behandlung von Spuren kann mit speziellen LRs auf sinnvolle Weise verknüpft werden. Schließlich identifizieren wir eine Reihe von Erweiterungen, die die Verwendung von LRs in komplexen Grammatiken weiter unterstützen würden.

1 Einführung

In einer lexikalischen Grammatiktheorie wie der *Head-driven Phrase Structure Grammar (HPSG)*[1] werden Lexika von hoher Komplexität verwendet, für die die Vermeidung von Redundanzen von großer Bedeutung ist.

[1] vgl. [Pollard/Sag87, Pollard/Sag94].

[Pollard/Sag87] sprechen von *vertikalen Redundanzen*, wo die zu einem Lemma gehörende Information nicht spezifisch ist, d.h. aus seiner Zugehörigkeit zu verschiedenen Klassen abgeleitet werden kann. Diese Art Redundanzen können vermieden werden, indem ein hierachisch strukturiertes Lexikon mit multipler Vererbung verwendet wird, in dem die für eine Klasse von Einträgen gültigen Informationen an einer Stelle spezifiziert und dadurch an die Instanzen vererbt werden.

In diesem Papier sollen die Ausdrucksmittel zur Vermeidung einer weiteren Art von Redundanzen im Lexikon, den *horizontalen Redundanzen*, beschrieben werden. Unter horizontalen Redundanzen sind die Fälle zu verstehen, in denen das Vorhandensein einer Beschreibung *A* im Lexikon das Vorhandensein einer weiteren Beschreibung *A'* impliziert. Um ein Beispiel zu nennen: In der Regel gehört zu jeder aktivischen Form eines transitiven Verbs auch eine entsprechende passivische Form, wobei zwischen den beiden Formen ein systematischer Zusammenhang besteht. Ein Lexikon, welches für die Aktiv- und die Passivform zwei voneinander unabhängige Einträge enthält, versäumt es nicht nur, diesen Zusammenhang explizit zu machen, sondern enthält darüberhinaus massive Redundanzen mit allen negativen Konsequenzen (Fehleranfälligkeit, erschwerte Erweiterbarkeit und Wartung). Das übliche Ausdrucksmittel zur Modellierung solcher systematischen Zusammenhänge zwischen Lexikoneinträgen sind *lexikalische Regeln*.

Im folgenden soll das Konzept der lexikalischen Regeln (LR) vorgestellt werden, sowie deren Implementierung in STUF. Obwohl für diese Implementierung einige bekannte Probleme mit der Formalisierung von LRs einschlägig sind, erweist sich der Einsatz von LRs in der IBM-Grammatik als sinnvoll, zumal sich für die Modellierung von speziellen Phänomenen, namentlich der Kopfbewegung, ein erweitertes Konzept von lexikalischen Regeln anbietet, welches sich aus Sicht der Verarbeitung als vorteilhaft erweist. Als Fallbeispiel soll deshalb im folgenden die Beschreibung von Kopfbewegung mittels LR dienen. Insbesondere soll nicht versucht werden, einen Überblick über alle Phänomene, die in der Literatur mit LRs beschrieben wurden, oder alljene, für die eine auf LRs basierende Beschreibung in die IBM Basisgrammatik integriert werden wird.

Für beide bislang angesprochenen Phänomene wurden Vorschläge zur Beschreibung gemacht, die ohne LRs auskommen: Passiv wird in [Lebeth94] durch morphosyntaktischen Strukturaufbau beschrieben; eine Verb-Zweit-Analyse, die unterspezifizierte Verbformen vorschlägt, beschreibt [Frank94]. In beiden Fällen jedoch ist nicht geklärt, welche Konsequenzen eine Anwendung der jeweiligen Alternative auf weitere Phänomene hätte. So würde morphosyntaktischer Strukturaufbau in vielen Fällen zu der Annahme unterschiedlicher leerer Köpfe zwingen, während die Unterspezifizierung hinsichtlich mehrerer Phänomene in ein und demselben Eintrag zu großen Problemen bei der Kodierung führt[2].

In diesem Papier soll und kann keine Einführung in die Grundlagen der HPSG oder der verwendeten Beschreibungssprache STUF vermittelt werden — für die HPSG sei der Leser auf [Pollard/Sag87, Pollard/Sag94] verwiesen; eine Einführung in die Sprache STUF III liefert [Momma et al.94]; eine allgemeine Beschreibung der IBM Basisgrammatik vermittelt [Geißler/Kiss94].

2 Lexikalische Regeln in Unifikationsgrammatiken

Die grundsätzliche Form, sowie eine intuitive Interpretation von LRs kann wie folgt beschrieben werden:

(1) LR: $\mathsf{A} \rightarrow \mathsf{B}$
Für jeden Lexikoneintrag, der der Beschreibung A entspricht, enthält das Lexikon einen weiteren Eintrag, der der Beschreibung B entspricht.

A und B sollen im folgenden auch als *Input-* bzw. *Outputbeschreibung* einer LR bezeichnet werden. Der informelle Begriff *„entsprechen"* kann für die STUF-Implementierung von LRs präzisiert werden durch *„unifizieren mit"*. Das Kriterium für die Anwendbarkeit einer LR ist also die erfolgreiche Unifikation der Inputbeschreibung mit einem gegebenen Lexikoneintrag.

[2] vgl. [Keller94].

Das Prinzip sei anhand des folgenden Beispiels verdeutlicht: Angenommen wir wollten den systematischen Zusammenhang zwischen der Positiv- und der Komparativform eines Adjektivs mittels einer LR beschreiben. Die entsprechende Regel ließe sich wie folgt beschreiben:

(2)

$$\begin{bmatrix} \mathit{PHON} & \boxed{1} \\ \mathit{SYNSEM|LOC|CAT|HEAD} & \mathit{adj}[\mathit{AFORM} \quad \mathit{positive}] \end{bmatrix}$$

$$\Rightarrow \quad \begin{bmatrix} \mathit{PHON} & \boxed{1} + {}'er' \\ \mathit{SYNSEM|LOC|CAT|HEAD} & \mathit{adj}[\mathit{AFORM} \quad \mathit{comparative}] \end{bmatrix}$$

Die LR in (2) stellt den gewünschten Zusammenhang zwischen den beiden Adjektivformen her. Die Komparativform eines Adjektivs entspricht somit der Positivform bis auf die entsprechende kategoriale Information und das zusätzliche Suffix *'er'*[3].

Unter einer prozeduralen Interpretation stellt (2) somit eine Anweisung dar, für die in Input- und Outputbeschreibung angegebenen Pfade die entsprechenden Änderungen vorzunehmen und darüberhinaus alljene Information, die in dem Lexikoneintrag für die Positivform enthalten sind, die aber in der Inputbeschreibung nicht aufgeführt werden, in den abgeleiteten Eintrag zu übernehmen.

Diese implizite Anweisung muß jedoch für die Implementation von LRs explizit gemacht werden. Darüber hinaus muß beachtet werden, daß LRs wie in (2) *nicht-monotone* Operationen darstellen, da hier Information destruktiv überschrieben wird. Solche Nicht-Monotonien sind jedoch in bestehenden Unifikationsformalismen meist nicht verfügbar. Im folgenden Abschnitt soll anhand der STUF-Implementation gezeigt werden, wie solche Nicht-Monotonien in einer monotonen Beschreibungssprache modelliert werden können.

[3] Vereinfachend wurde in (2) von suppletiven Komparativbildungen (*gut* → *besser*), Komparativbildungen mit Umlautung (*lang* → *länger*) u.ä. abstrahiert; ebenso wurde darauf verzichtet, den semantischen Zusammenhang zwischen den beiden Formen zu beschreiben.

3 Lexikalische Regeln in der IBM-Basisgrammatik

Wie erwähnt soll in den folgenden Abschnitten die Behandlung von Verb-Zweit-Strukturen als Beispiel für die Anwendung von LRs in der IBM-Basisgrammatik dienen.

In dieser Grammatik werden Verb-Zweit-Sätze über eine Kopfbewegungsanalyse nach [Kiss/Wesche91] beschrieben. Danach selegiert ein Verb in Verb-Zweit-Position statt seiner Argumente eine Verbalprojektion mit leerem Kopf, in der alle Argumente (und evtl. Adjunkte) abgebunden werden. Eine Verbform kann somit in verschiedenen Kontexten (Verb-Letzt- vs. Verb-Zweit-Sätze) mit unterschiedlichen Eigenschaften auftreten. Der systematische Zusammenhang zwischen diesen beiden Formen wird in [Kiss93] durch eine Kopfbewegungs-LR beschrieben:

(3)

$$_{syn}\begin{bmatrix} LOC & \boxed{3}\begin{bmatrix} CAT|HEAD & \boxed{1}[VFORM fin] \end{bmatrix} \end{bmatrix}$$

$$\Rightarrow$$

$$_{syn}\begin{bmatrix} CAT & \begin{bmatrix} HEAD & \boxed{1} \\ SCT & < \begin{bmatrix} SYN & \begin{bmatrix} LOC & \begin{bmatrix} CAT & [HEAD \boxed{1}, SCT <>] \\ CONT & \boxed{4} \end{bmatrix} \\ NL & INH|DSL < \boxed{3} > \end{bmatrix} \end{bmatrix} > \end{bmatrix} \\ CONT & \boxed{4} \end{bmatrix}$$

Die konkrete Implementierung muß wie erwähnt diejenigen Werte, auf die in Input- oder Outputstruktur nicht explizit verwiesen wird, unverändert übertragen. Im vorliegenden Fall wurde die LR als STUF-Sorte mit zwei Argumenten formuliert, welche die maximal unterspezifizierte Struktur `top` beschreibt. Diese Formulierung, die wie weiter unten gezeigt wird, eine elegante Integration der LR-Komponente in den Lexikonzugriff ermöglicht, bewirkt zwischen den beiden Argumente (den Input- bzw. Outputstruktur) die entsprechende Propagation der constraints, wie sie in (4) beschrieben wird.

(4)
```
hmlr(  %%%% INPUT LEXICAL ENTRY %%%%%%%%%%%
             phon: [#Word] &
             syn: (loc: (#Loc &
                         head: (fin_verb &
                                agr:     #Agr &
                                mod:     #Mod &
                                anchor:  #Anchor &
                                reg_dir: left) &
                         gram_reln: refo : #Refo) &
                   nonloc: inh: (dsl: [] &
                                 slash: []))
                   ,
       %%%%% OUTPUT LEXICAL ENTRY %%%%%%%%%%%
                 phon: [#Word] &
                 syn: (loc: (head: (top_verb &
                                    v_form : front &
                                    agr:     #Agr &
                                    mod:     #Mod &
                                    anchor:  #Anchor &
                                    reg_dir: right) &
                             gram_reln : refo: #Refo &
                             subcat: [head(last_verb) & subcat([]) &
                                      syn: nonloc: inher: (dsl: [#Loc])]) &
                       nonloc: inh: (slash: [] &
                                     dsl: []))
        => top.
```

Wie nun kann diese Regel, die ja keinen abgeleiteten Eintrag „erzeugt", der gewünschten Funkionalität entsprechen? Nehmen wir zunächst an, die Aufgabe des Lexikonzugriffs ließe sich durch die folgenden Punkte beschreiben:

- Für ein gegebenes Zeichen im zu analysierenden Input ist die zugehörige lexikalische Information zu ermitteln.
- Dies kann im einfachsten Fall durch einen Zugriff auf das Kernlexikon, das die nicht-abgeleiteten Einträge enthält, geschehen.

- Alternativ dazu besteht jedoch die Möglichkeit, daß der gesuchte Eintrag der Outputspezifikation einer LR entspricht, deren Inputspezifikation ihrerseits (direkt oder indirekt) durch das Lexikon und die Menge der LRs lizensiert wird.

Der Lexikonzugriff muß also über eine rekursive Formulierung diejenigen Fälle beschreiben, in denen eine Struktur durch mehr als eine LR aus dem Kernlexikon abgeleitet werden kann. Der Zugriff erstreckt sich somit auf die mittels der LRs über dem Kernlexikon gebildeten transitiven Hülle[4]. Die gewünschten Eigenschaften werden durch die Klauseln in (5) realisiert:

(5)
```
lex_item( #X & phon: [#Phon] )  => #X & core_lex( #Phon ).
lex_item( #X )  => #X & lex_rule( lex_item( #In ), #X ).


lex_rule( #In,#Out ) => hmlr( #In,#Out ) & #Out.
lex_rule( #In,#Out ) => celr( #In,#Out ) & #Out.
```

Die beiden Klauseln für `lex-item` decken die beiden oben beschriebenen Fälle ab: Entweder die gesuchte Struktur ist ein Eintrag des Kernlexikons, oder sie ist aus diesem durch eine oder mehrere Anwendungen von LRs ableitbar.

4 Probleme mit der Implementierung von lexikalischen Regeln

Während die vorgestellte Umsetzung des LR-Konzeptes grundsätzlich die Vermeidung der eingangs beschriebenen horizontalen Redundanzen im Lexikon ermöglicht,

[4]Ebenso ist es denkbar, das Kernlexikon zur Compilationszeit durch die Anwendung der LRs — teilweise oder ganz — zu expandieren. Die Konsequenzen eines solchen Vorgehens — vereinfachter Zugriff aber evtl. erheblich vergrößertes Lexikon — sollen hier jedoch nicht erörtert werden.

bleiben eine Reihe von Problemen ungelöst. Diese Probleme[5] sollen in den folgenden Abschnitten beschrieben sowie bzgl. ihrer Relevanz für die Anwendung in der IBM-Basisgrammatik untersucht werden.

Der formale Status von lexikalischen Regeln

In [Pollard/Sag94] werden LRs als „implikative Relationen zwischen lexikalischen Einträgen"[6] verstanden. Lexikoneinträge jedoch seien Beschreibungen von Featurestrukturen, nicht (totale) Featurestrukturen selbst. Benötigt werde deshalb eine Beschreibungssprache höherer Ordnung, die Beschreibungen zueinander in Beziehung setzt. Im Gegensatz dazu sind Sorten in STUF über die durch sie beschriebenen Mengen von (totalen) Objekten definiert.

Diese Diskrepanz hat sich in der Praxis als hinnehmbar erwiesen. Nachdem STUF-Strukturen stets (nicht-totale) Beschreibungen darstellen, können Input- und Outputstrukturen von LRs auch Beschreibungen (in diesem Falle nicht-totale Lexikoneinträge) beschreiben. Hinsichtlich der Frage, ob LRs Beziehungen zwischen *Wörtern* (tokens) oder Beschreibungen von (Klassen von) *Wörtern* (types) etablieren, soll hier keine Entscheidung getroffen werden.

Die in STUF mögliche alternative Formulierung von LRs als Macros entspricht zwar eher dem Konzept, Beschreibungen ohne Bezug auf die durch sie beschriebenen Objekte zu manipulieren, erlaubt jedoch weder die hier angestrebte Bidirektionalität von LRs, noch weitergehende Eigenschaften bei der Behandlung leerer Elemente (vgl. Abschnitt (5)).

Koindizieren oder Kopieren

Ähnliche Fragen ergeben sich bei der Interpretation der Koindizierungsmarkierungen in (3). Diese Markierungen drücken Strukturteilung (token identity) zwischen den beteiligten Pfaden aus. Gewünscht wird an dieser Stelle allerdings nicht die Identität von Objekten, sondern die von Beschreibungen[7].

[5] Eine ausführliche Darstellung findet sich vor allem in [Meurers94].

[6] vgl. [Pollard/Sag94, S. 395].

[7] vgl. [Meurers94, S.29] .

Generalisierungen über Kategorien

Ernsthaftere Probleme ergeben sich beim Versuch, mit LRs Generalisierungen über Kategorien hinweg auszudrücken. Nehmen wir an, die Signatur einer Grammatik spezifiziere `present-verb` und `past-verb` als Untertypen zu `finite-verb`, wobei die angegebenen Typen jeweils Werte für das `HEAD`-Merkmal einer Struktur seien. Eine Regel wie die HMLR in (3), die auf einen Wert innerhalb der `HEAD`-Struktur verändernd zugreift (in diesem Falle z.B: das Merkmal `REG-DIR`, desses Wert von `left` auf `right` wechselt) kann die Information über den spezifischen Typ von `HEAD` nicht von der Input- auf die Outputstruktur übertragen[8].

(6)

$$\begin{array}{ccc} & fin\text{-}verb \begin{bmatrix} \text{MOD} & sign \\ \text{REG-DIR} & reg\text{-}dir \\ \text{AGR} & agr \end{bmatrix} & \\ \swarrow & & \searrow \\ past\text{-}verb & & pres\text{-}verb \end{array}$$

Da auf die Typinformation nicht gesondert zugegriffen werden kann, ist man gezwungen, verschiedene Instanzen einer LR zu spezifizieren. Die gewünschte Generalisierung über alle finiten Verben geht damit verloren.

Ein ähnliches Problem tritt bei LRs auf, die über Strukturen mit unterschiedlichem Merkmalsinventar generalisieren. So entsteht für eine LR, die auf Verben und Nomina gleichermaßen anwendbar sein soll, das Problem, wie die Merkmale `VFORM` bzw. `NFORM` übertragen werden sollen, die jeweils nur für eine Klasse angemessen sind, während ihre Integration in Strukturen der jeweils anderen Klasse zu Inkonsistenzen führt. Wieder kann mit einem Verzicht auf die Generalisierung diesem Problem begegnet werden.

[8] vgl. [Meurers94, S.32].

Attraktiver erscheint die Option, die Aufspaltung in mehrere LRs zu vermeiden und statt dessen die Disjunktion lokal zu halten, indem man die möglichen Fälle über *definite clause attachment* abfängt. Eine implementierbare Behandlung des in (6) dargestellten Problems ist in (7) gegeben:

(7)
```
sample_lex_rule(  %%%% INPUT STRUC
                  #Struc &
                  fin-verb &
                    (MOD      : #1 &
                     REG-DIR  : left &
                     AGR      : #2),
                  %%%% OUTPUT STRUC
                  past_or_pres(#Struc) &
                    (MOD      : #1 &
                     REG-DIR  : right &
                     AGR      : #2) )
               => top.

past_or_pres(#Struc & pres_verb) => pres_verb.
past_or_pres(#Struc & past_verb) => past_verb.
```

Die angegebene Methode — die auch auf das beschriebene Problem mit den unterschiedlichen Merkmalskonfigurationen anwendbar ist — vermeidet die Aufspaltung in spezifische Regeln. Da disjunktive Information damit lokal gehalten werden kann und zudem die benötigten Ausdrucksmittel von der STUF-Beschreibungssprache zur Verfügung gestellt werden, ist die obige Lösung einer Aufspaltung einer LR in spezifischere Regeln vorzuziehen.

Verschiedene Abfolgen von lexikalischen Regeln

Ist eine Struktur das Ergebnis der Anwendung von mehreren LRs so besteht die Gefahr, daß für unterschiedliche Permutationen von LRs weitgehend identische Lösungen einer Analyse entstehen, die sich lediglich in der Reihenfolge der

Anwendung der LRs unterschieden. Ohne zusätzliche Funktionalitäten in der LR-Komponente ist es damit in der Verantwortung des Grammatikautors, solche Ambiguitäten auszuschließen, indem die Input- und Outputspezifikationen im Hinblick auf die gewünschte Reihenfolge ergänzt werden[9].

5 Vorteile

In den obigen Abschnitten wurde deutlich, daß eine Implementierung von LRs innerhalb bestehender, monotoner Formalismen zu einer Reihe von Problemen führen kann. Während einige dieser Probleme weniger Einfluß auf die Klarheit und Ausdruckskraft einer LR-Komponente haben, ist man in anderen Fällen zu Formulierungen von einiger Komplexität gezwungen, um die gewünschten Eigenschaften der LRs zu modellieren.

Folgende Überlegungen lassen das Konzept der LRs dennoch attraktiv erscheinen. Zum einen eröffnet ein modifizierter Einsatz von LRs, der in Abschnitt (5) vorgestellt werden soll, effiziente Möglichkeiten beim Umgang mit leeren Elementen. Zum anderen erwiesen sich LRs als sehr flexibles Ausdrucksmittel für nicht-monotone Kovariationen im Lexikon, das eine modulare Kodierung der verschiedenen Phänomene erlaubt.

Lexikalische Regeln und Spurenvorhersage

Die in (3) beschriebene Analyse von Verb-Zweit-Strukturen erfordert (Kopf-)-Spuren, die den Komplementspuren in [Pollard/Sag94, S.161], ähneln. Statt der Koindizierung des `LOCAL`-Wertes mit dem `SLASH`-Wert erfogt hier die Strukturteilung mit dem `DSL`-Wert. Anders auch als im Fall von Komplementextraktion läßt sich Kopfbewegung nicht ohne Spuren beschreiben, will man nicht auf ein binär-verzweigendes Mittelfeld verzichten. Die Kopfspur läßt sich somit wie folgt beschreiben:

[9] Da anzunehmen ist, daß diese Angaben bei komplexeren LR-Komponenten schnell zu stark erhöhter Unüberschaubarkeit führen, wird für größere Grammatiken eine prinzipielle Lösung dieses Problems nötig sein.

(8)

$$\begin{bmatrix} PHON & <> \\ SYN & \begin{bmatrix} LOC & \boxed{1} \\ NONLOC|INH|DSL & \boxed{1} \end{bmatrix} \end{bmatrix}$$

Die extreme Unterspezifiziertheit dieser Struktur stellt jedoch die Verarbeitung von Analysen mit Spuren vor große Probleme. Die Struktur in (8) kann für beliebige Zeichen subkategorisieren, beliebige Zeichen modifizieren, oder beliebige Lücken abbinden.

Diese Unterspezifiziertheit ist jedoch vermeidbar, da eine gegebenes Verb in Verb-Zweit-Position sehr detallierte Beschränkungen für die mögliche Kopfspur impliziert. Eine sinnvolle Modifikation der Kopfbewegungs-LR berücksichtigt diesen Zusammenhang. Eine transitives Verb, welches durch diese Regel modifiziert wurde, impliziert das Vorhandenensein einer Spur wie in (9), die erheblich spezifischer als die Spur in (8) ist:

(9)

$$\begin{bmatrix} PHON & <> \\ SYN & \begin{bmatrix} LOC & \boxed{1} \begin{bmatrix} HEAD & \begin{bmatrix} \text{REG-DIR} & left \\ \text{V-FORM} & \text{last-verb} \\ ... & ... \end{bmatrix} \\ SUBCAT & < DP[nom], DP[acc] > \end{bmatrix} \\ NONLOC|INH|DSL & \boxed{1} \end{bmatrix} \end{bmatrix}$$

Die modifizierte Form der Kopfbewegungs-LR ermittelt diese Struktur, indem die entsprechenden Beschränkungen über ein drittes Argument hinzugefügt werden.

Komplexität der Einträge

Eine verallgemeinerte Lösung für die gleichzeitige Vermeidung disjunktiver Lexikoneinträge und die kompakte Formulierung von systematischen Variationen könnten *verteilte Disjunktionen* im Lexikon darstellen[10]. Die Idee hierbei ist, daß

[10]Eine ausführliche Argumentation in dieser Richtung findet sich in [Meurers94].

ein Eintrag die Information sowohl der Input-, als auch der Outputstruktur einer LR in disjunktiver Form enthält.

Das Prinzip läßt sich an folgendem Beispiel (aus [Meurers94]) verdeutlichen, das die Formen *fin*ite und *bse* (Grundform) eines Verbs als verteilte Disjunktion darstellt:

(10)

$$
{}_{cat}\left[\begin{array}{ll} HEAD & {}_{verb}[VFORM\{_v fin; bse\}] \\ VAL & \left[\begin{array}{ll} SUBJ & \{_v <>; <\boxed{1}>\} \\ COMPS & \{_v < (\boxed{1}[CASE \quad nom]).\boxed{2} >; \boxed{2}\} \\ SUBCAT & < \boxed{1}.\boxed{2} > \end{array}\right] \end{array}\right]
$$

Disjunktionen (geklammert mit $\{\ldots\}$) mit gleichem Subskript sind dabei derart verbunden, daß die Auswahl des n-ten Disjunkts an einer Stelle auch die Auswahl des jeweils n-ten Disjunkts an jeder anderen Stelle verlangt.

Statt abgeleitete Einträge über eine LR zu ermitteln, wird die relevante abweichende Information hier also direkt im Lexikon spezifiziert, wobei der Mechanismus der verteilten Disjunktion die entstehenden Disjunktionen lokal zu halten erlaubt. Für zwei Einträge (des LR-Ansatzes), die sich evtl. nur in einer kleinen Zahl von Details unterscheiden, entsteht so ein einzelner, an manchen Stellen disjunktiv spezifizierter Eintrag. Diese Vorgehensweise hat den Vorteil, daß ein Lexikonzugriff in vielen Fällen deterministisch ist und die enthaltene Disjunktion bereits vor ihrer Auflösung den Suchraum verkleinert[11].

Während diese Formulierung von Variationen im Lexikon für einfachere Fälle also klare Vorteile gegenüber dem vorgestellten LR-Konzept besitzt, werden die Grenzen des Verfahrens schnell deutlich, wenn man versucht, ein komplexeres System von LRs in der angegebenen Weise direkt im Lexikon zu kodieren. Da die dabei entstehende Disjunktion für den Grammatikautor schnell unbeherrschbar wird, müßte sie in einem Compilationsprozeß aus einer Menge von Spezifikationen erzeugt werden[12].

---

[11] Die Disjunktion '`CASE: (acc ; nom)`' etwa erlaubt es, '`CASE: dat`' auszuschließen, noch bevor die Disjunktion aufgelöst wird.

[12] Ein solcher Ansatz wird auch in [Meurers94] vorgeschlagen.

Schwerwiegender als das praktische Problem, Disjunktionen ab einer bestimmten Komplexitätsstufe zu spezifizieren, ist die Tatsache, daß in solcherart zusammengefaßten Beschreibungen verschiedener Phänomene die Lokalisierung der Information bzgl. eines bestimmten Phänomens nicht mehr möglich ist.

Die Komplexität, die bereits bei der Zusammenfassung zweier einfacher abstrakter LRs in eine verteilte Disjunktion entsteht, läßt sich an folgendem Beispiel verdeutlichen:

$$(1) \qquad \begin{bmatrix} A & + \\ B & \begin{bmatrix} C & - \\ D & <+,-> \end{bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} A & - \\ B & \begin{bmatrix} C & - \\ D & <+,+> \end{bmatrix} \end{bmatrix}$$

$$(2) \qquad \begin{bmatrix} A & - \\ B & \begin{bmatrix} C & top \\ D & <+,+> \end{bmatrix} \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} A & - \\ B & \begin{bmatrix} C & + \\ D & <-,-> \end{bmatrix} \end{bmatrix}$$

$$(1') \qquad \begin{bmatrix} A & \{_1+;-\} \\ B & \begin{bmatrix} C & - \\ D & <+,\{_1-;+\}> \end{bmatrix} \end{bmatrix}$$

$$(2') \qquad \begin{bmatrix} A & - \\ B & \begin{bmatrix} C & \{_2 top;+\} \\ D & \{_2<+,+>;<-,->\}> \end{bmatrix} \end{bmatrix}$$

$$(1') \oplus (2') \qquad \begin{bmatrix} A & \{_3\{_1+;-\};-\} \\ B & \begin{bmatrix} C & \{_3-;\{_2 top;+\}\} \\ D & \{_3<+,\{_1-;+\}>;\{_2<+,+>;<-,->\}\} \end{bmatrix} \end{bmatrix}$$

Die entstandene Struktur macht deutlich, daß derartige Ausdrücke im Hinblick auf leichte Erweiterbarkeit und Modifizierbarkeit nur sehr schwer handhabbar sind.

Während also die Verwendung von verteilten Disjunktionen aus Effizienzüberlegungen heraus wünschenswert ist, findet dieser Ansatz (ohne zusätzlichen Compilationsschritt) schnell seine Grenzen, wenn die beteiligten Stukturen komplexer werden.

6 Fazit und Ausblick

In einer Situation, in der systematische Variation im Lexikon durch Konstruktionen innerhalb der Beschreibungssprache (mit ihrer relationalen Erweiterung) beschrieben werden soll, in der also kein externes Modul eine Interpretation oder Compilation übernimmt, erweisen sich LRs als das vielseitigste, allgemeinste und für den Grammatikautor intuitivste Konzept zur Vermeidung horizontaler Redundanzen in Lexikon.

Die Implementation von LRs in einer monotonen Beschreibungssprache wie STUF bedingt allerdings Abstriche hinsichtlich der Ausdruckskraft und wirft komplexe Fragen bzgl. der Anwendungsreihenfolge und der effizienten Repräsentation und Verarbeitung der entstehenden top-level-Disjunktionen von lexikalischen Einträgen auf.

Zu den Anforderungen, die an ein Modul zur Verarbeitung von LRs zu stellen wären, gehören deshalb (mindestens) die folgenden Punkte:

- Koindizierung von Strukturen in Input- und Outputbeschreibungen sind in der Regel nicht als Strukturteilung, sondern als Kopieroperation zu lesen. Eine solche Kopieroperation wird jedoch in bestehenden unifikationsbasierten Beschreibungssprachen nicht zur Verfügung gestellt.
- Ebenso sind allgemeine Konzepte zum Umgang mit Typen und zur Frage unterschiedlicher Abfolgebeziehungen zwischen LRs zu entwickeln.
- Aus prinzipiellen Gründen ist es nicht möglich, zwischen solchen Koindizierungen zu unterscheiden, die durch die LR verändert werden sollen und solchen, für die dies nicht der Fall ist. Einen Fall, in dem dies zu unerwünschten Resultaten einer LR führt, beschreibt [Müller94].
- Ein komplexeres System von LRs würde im Rahmen der oben vorgestellten Implementation aus einem Basiseintrag eine größere Zahl disjunkter

Einträge „erzeugen“, die sich zum Teil vielleicht nur marginal unterscheiden. Eine effizientere Lösung könnte darin bestehen, die Outputstruktur einer LR als Disjunktion der beiden in Beziehung gesetzten Beschreibungen zu spezifizieren.

Die Frage jedoch, wie ein solcher Ansatz in einem komplexen LR-System zu realisieren wäre, ist ungelöst. Es müßten nicht nur die ursprünglichen Basiseinträge durch die abgeleiteten Disjunktionen ersetzt werden, da sie ja bereits durch eines der entstandenen Disjunkte beschrieben würden. Darüber hinaus ist unklar, wie gesichert werden kann, daß bei der Anwendung einer LR auf einen abgeleiteten Eintrag nicht Optionen verloren gehen, weil die Inputstruktur mit Teilen der Disjunktion inkompatibel ist.

Abgesehen von diesen ernsten formalen Fragen bzgl. der Interpretation und Mächtigkeit von LRs ist zu klären, mit welchen grundsätzlichen linguistischen Prinzipien sich die Menge der möglichen LRs einschränken ließe. Anders als für sprachliche Zeichen vom Typ *phrase*, die durch eine Reihe detallierter Prinzipien lizensiert werden müssen, um wohlgeformt zu sein, existiert bislang in HPSG keine detaillierte Beschreibung derjenigen Prinzipien, die die Wohlgeformtheit der Zeichen des Typs *word* beschreiben.

Lexikalische Regeln, die in diesem Papier als wichtige Abstaktionsmittel über komplexen, strukturierten Lexika beschrieben wurden, sind somit trotz ihrer Bedeutung für den Grammatikautor gleichermaßen formal nicht mächtig genug und linguistisch zu mächtig.

Literatur

[Frank94] Frank, Annette: *Verb-Second by Lexical Rule or by Underspecification*, Arbeitspapiere des Sonderforschungsbereichs 340, Nr. 43-1994.

[Geißler/Kiss94] Geißler, Stefan und Tibor Kiss: *Erläuterungen zur Umsetzung einer HPSG im Basisformalismus STUF III*, VM Report Nr. 19, 1994.

[Keller94] Keller, Frank: *Extraposition in HPSG*, Studienarbeit, IMS Uni Stuttgart, 1994.

[Kiss93] Kiss, Tibor: *Derivationelle Aspekte im merkmalsbasierten Lexikon*, In: Deklarative und prozedurale Aspekte der Sprachverarbeitung, Tagungsband der Sektion Computerlinguistik der DGfS, Hamburg, 1993.

[Kiss/Wesche91] Kiss Tibor & Birgit Wesche: Verb Order and Head Movement. In: Herzog, O. & C.-R. Rollinger: *Texte Understanding in LILOG*, Lecture notes in Artificial Intelligence Nr. 546, Berlin 1991, S. 216-240.

[Lebeth94] Lebeth, Kai: *Morphosyntaktischer Strukturaufbau im HPSG-Lexikon: Eine Alternative zu lexikalischen Regeln.* VM Report Nr 18. 1994.

[Meurers94] Meurers, Detmar: *On Implementing an HPSG Theory*, In: Hinrichs, Meurers, Nakazawa: *Partial-VP and Split-NP Topicalization in German, An HPSG Analysis and its implementation*, Arbeitspapiere des SFB 340, Nr 58, Tübingen, 1994.

[Momma et al.94] Momma, Stefan, Annette Opalka und Ingo Raasch: *STUF-III User Manual.* IBM Informationssysteme, Heidelberg, 1994.

[Müller94] Müller, Stefan: *Problems with Complement Extraction Lexical Rules*, unveröffentlichtes Manuskript, Berlin, 1994.

[Pollard/Sag87] Pollard, Carl & Ivan A. Sag: *Information-based Syntax and Semantics, Vol. I*, CSLI Lecture Notes Nr. 13, Stanford, 1987.

[Pollard/Sag94] Pollard, Carl & Ivan A. Sag: *Head-driven Phrasen Structure Grammar*, Univ of Chicago Press, 1994.