Einführung in die Grammatikentwicklung: Übung 7 (21. Oktober 2003)

Ziele:

1. Implementierung von Listenverknüpfung mit Hilfe von Differenzlisten
2. Hinzufügen semantischer Information zu Lexikoneinträgen und Regeln
3. Benutzung des LKB-Generators zur Bestimmung der Übergenerierung der Grammatik.

Vorbereitung: Es gibt mehrere Möglichkeiten, eine Startgrammatik für diese Übung zu bekommen. Beide benötigen den Schritt (i). Wenn Sie eine funktionierende Grammatik mit Lexikonregeln und Flexionsregeln haben, können Sie Ihre alten Typen, Ihr altes Lexikon und Ihre alten Regeln behalten, indem Sie die optionalen Schritte (ii) und (iii) ausführen.

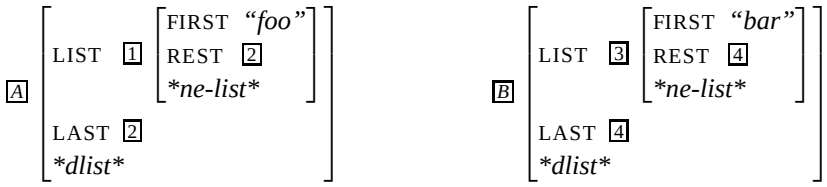
(i) Hohlen Sie eine neue Grammatik aus dem Versionskontrollsystem CVS, indem Sie in einem xterm
`cvs checkout Grammatik7`
eingeben. Das erzeugt das Unterverzeichnis `Grammatik7`, das die gesamte Grammatik enthält. *Diejenigen, die die Aufgaben der letzten Übung vollständig gelöst haben und ihre Grammatiken weiterbenutzen möchten, führen die Schritte (ii) und (iii) aus. Alle anderen beginnen direkt mit den Übungen.*

(ii) Kopieren Sie die Dateien 'types.tdl', 'lexicon.tdl', 'rules.tdl', 'lrules.tdl' und 'inflr.tdl' aus dem Verzeichnis, in dem Sie vorher gearbeitet haben, in das Verzeichnis Grammatik7.
```
cd Grammatik7
cp ../Grammatik5/*.tdl .
```
Dabei müssen Sie evtl. den Verzeichnisnamen an das von Ihnen verwendete Verzeichnis anpassen.

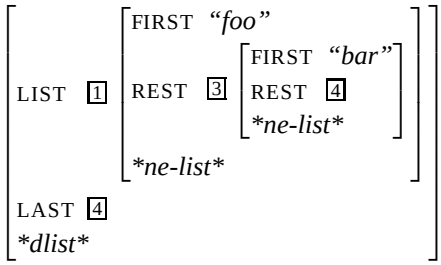
(iii) Fügen Sie den Inhalt der Datei 'extras.tdl' am Ende Ihrer Datei 'types.tdl' ein. Ändern Sie die Grammatik so, daß das Merkmal ORTH nicht mehr bei *lex-item* eingeführt wird, sondern bei *syn-struc*. Ändern Sie den Wert von ORTH von **list** auf **dlist**. Ersetzen Sie im Lexikon alle Vorkommen von ORTH durch den Pfad zum ersten Element der neuen Orthographie-Differenzliste: ORTH.LIST.FIRST. Stellen Sie sicher, daß sie die Grammatik noch laden und damit parsen können, bevor Sie weitermachen.

Übungen:

1. Der Formalismus für getypte Merkmalstrukturen, der im LKB-System implementiert ist, erlaubt keine Verwendung von relationalen Beschränkungen (*relational cosntraints*) wie z. B. *append* oder *reverse* für Listen. Stattdessen werden Differenzlisten verwendet, mit deren Hilfe man Listenverknüpfung allein mit Unifikation ausdrücken kann. Eine Differenzliste ist eine Liste mit offenem Ende, die in eine Behälterstruktur eingebettet ist, die einen Zeiger auf das Ende der Liste bereitstellt.



Indem wir den LAST-Zeiger auf das Ende der Liste A verwenden, können wir B an A anhängen. Wir müssen dazu den Anfang von B (d. h., den Wert von LIST) mit dem Ende (dem LAST-Wert) von A unifizieren und das Ende der Liste B als das neue Ende der Listenverkettung verwenden.



Siehe hierzu auch Abschnitt 4.8.2 im Online-Manual für das LKB-System.

Das Ziel dieser Übung ist es, die ORTH-Wert von Wörtern mit Hilfe von Listenverkettung zu verknüpfen und diese Verknüpfung als ORTH-Wert der entsprechenden Phrasen zu repräsentieren. Der oberste Knoten einer vollständigen Analyse eines Satzes ('S') sollte als ORTH-Wert eine Differenzliste haben, die alle Wörter des Satzes in der richtigen Reihenfolge enthält.

(a) Vergewissern Sie sich, daß ORTH beim Typ *syn-struc* eingeführt wird und daß der Wert von ORTH als **dlist** angegeben ist (siehe die oben gegebenen Vorbereitungshinweise). Vergewissern Sie sich auch, daß in allen Lexikoneinträgen die Orthographieinformation unter dem Pfad ORTH.LIST.FIRST repräsentiert ist.

(b) Die allgemeine Beschränkung für Werte von ORTH (**dlist**) erlaubt beliebig lange Listen. Damit die Listenverknüpfung richtig funktioniert, ist es wichtig, daß der LAST-Zeiger in jeder Differenzliste korrekt auf die Listenposition nach dem letzten Listenelement zeigt, d. h. für die leere Liste müssen LIST und LAST koindiziert sein, wohingegen LAST bei einer einelementigen Liste auf denselben Wert wie LIST.REST zeigt.

(c) Jede Grammatikregel muß die ORTH-Werte ihrer Töchter verketten und die sich ergebende Liste zum ORTH-Wert der Mutter machen. Führen Sie einen Typ *binary-rule* ein, der von *phrase* erbt und der die ensprechenden Beschränkungen für die Listenverkettung hat. Ein solcher Typ ist sinnvoll, da man – wenn man diesen Typ verwendet – die Verknüpfungsoperation in rules.tdl nicht bei jeder Regel neu spezifizieren muß.

(d) Damit die Regeln in rules.tdl von genau einem Typ erben, müssen wir eine Kreuzklassifizierung aus den Typen *binary-rule* und den Typen *head-initial* und *head-final*, die in Übung 4 eingeführt wurden, erzeugen. Verwenden Sie Mehrfachvererbung, um die Typen *binary-head-initial* und *binary-head-final* zu definieren. Ändern Sie die Definition von *root-hi* und *root-hf* so ab, daß diese Typen von *binary-head-initial* und *binary-head-final* statt von *head-initial* und *head-final* erben.

(e) Ändern Sie die Datei rules.tdl entsprechend ab, so daß die neuen, spezifischeren Typen verwendet werden. Überprüfen Sie die Korrektheit, indem Sie einige Sätze interaktiv parsen und den ORTH-Wert am Satz-Knoten angucken und verifizieren, daß er alle Wörter, die im Satz vorkommen, enthält. Benutzen Sie den Batch-Parse-Mechanismus mit der Datei test.items, um sicherzustellen„ daß sich die Abdeckung der Grammatik nicht verändert hat.

2. Im nächsten Schritt werden wir die Grammatik um semantische Information erweitern. Die grundlegende Operation für die Komposition semantischer Information ist die Listenverkettung. Semantische Relationen werden von Wörtern eingeführt und parallel zur syntaktischen Kombination von Wörtern (oder Wortgruppen) zusammengefügt, wenn größere Phrasen gebildet werden. Wir werden den Typ *relation* benutzen um Grundeinheiten der Semanik zu repräsentieren, die mit Wörtern verknüpft sind. Dieser Typ hat die Untertypen *arg1-relation*, *arg1-2-relation* und *arg1-2-3-relation* für Prädikate mit entsprechender Stelligkeit:

$$\begin{bmatrix} \text{PRED} & \textit{string} \\ \text{ARG0} & \textit{index} \\ \textit{relation} & \end{bmatrix} \qquad \begin{bmatrix} \text{PRED} & \textit{string} \\ \text{ARG0} & \textit{index} \\ \text{ARG1} & \textit{index} \\ \textit{arg1-relation} & \end{bmatrix} \qquad \ldots \qquad \begin{bmatrix} \text{PRED} & \textit{string} \\ \text{ARG0} & \textit{index} \\ \text{ARG1} & \textit{index} \\ \text{ARG2} & \textit{index} \\ \text{ARG3} & \textit{index} \\ \textit{arg1-2-3-relation} & \end{bmatrix}$$

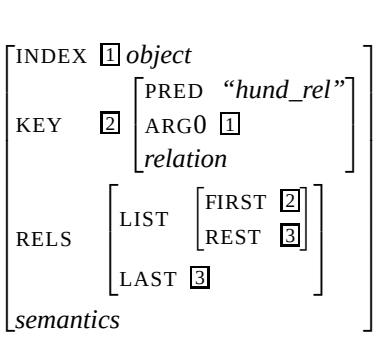
(a) Fügen Sie die atomaren Typen *index*, *object* und *event* ein. Diese Typen entsprechen den Variablen für die Zuweisung semantischer Rollen.

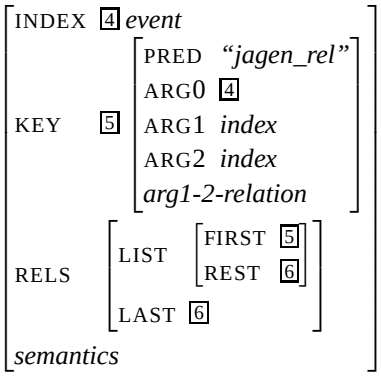
(b) Fügen Sie die oben beschriebenen Typen *relation*, *arg1-relation*, *arg1-2-relation* und *arg1-2-3-relation* als Untertypen von *feat-struc* ein.

Der Bedeutungsbeitrag von Wörtern und Phrasen wird als Wert des beim Typ *syn-struc* neu einzuführenden Merkmals SEM repräsentiert. Der Wert von SEM ist eine Merkmalstruktur, die wie folgt aufgebaut ist:

$$\begin{bmatrix} \text{INDEX} & \textit{index} \\ \text{KEY} & \textit{relation} \\ \text{RELS} & \textit{*dlist*} \\ \textit{semantics} & \end{bmatrix}$$

Das INDEX-Merkmal entspricht der externen Variable, die von anderen Ausdrücken gebunden werden kann. Der KEY-Wert zeigt auf die ausgewählte Relation, die für semantische Selektion benutzt wird (typischerweise kommt diese Information vom semantischen Kopf, siehe auch (**?**)). Der RELS-Wert enthält eine Liste von Relationen (siehe unten). Für die Lexikoneinträge für *Hund-* und *jag-* nehmen wir die folgende Semantik an (als Wert des SEM-Merkmals):





(a) Führen Sie den Typ *semantics* ein, fügen Sie das Merkmal SEM bei der Typdefinition von *syn-struc* ein und beschränken Sie dessen Wert auf *semantics*.

(b) Erweitern Sie die Typdefinition von *lexem* so, daß zum Ausdruck kommt, daß (i) lexikalische Elemente eine RELS-Liste mit genau einem Element haben, (ii) die KEY-Relation mit dem ersten (und einzigen) Element von RELS übereinstimmt und (iii) der INDEX das ARG0 des KEY ist.

(c) Erweitern Sie die Typen *det-lxm*, *noun-lxm* und *verb-lxm* (bzw. das Äquivalent in Ihrer Grammatik) so, daß der semantische INDEX für Determinirer und Nomen vom Typ *object* und für Verben vom Typ *event* ist.

(d) Fügen Sie für jeden Lexikoneintrag eine Relation als Wert von SEM.KEY.PRED ein. (Komplementpräpositionen werden vorerst ignoriert.) Laden Sie die Grammatik neu und überprüfen Sie mit dem Menüpunkt 'View – Lex Entry' den Lexikoneintrag für *Hund-* und *jag-*. Stellen Sie sicher, daß die Einträge so aussehen, wie es oben angegeben wurde.

In Analogie zum Zusammensammeln der Information unter ORTH wird die Bedeutung einer Phrase als Verkettung der Listen, die die Bedeutung der Töchter enthalten, berechnet: Der RELS-Wert einer Phrase ist die Konkatenation der RELS-Werte der Töchter. Wir benutzen eine Liste für die Repräsentation der Bedeutung, die Reihenfolge der Listenelemente ist allerdings irrelevant: Wir benutzen eine Liste, um eine Multimenge (Engl. *multi-set* oder auch *bag*) zu repräsentieren.

(a) In allen Phrasen werden INDEX und KEY vom semantischen Kopf beigesteuert. In unserer Grammatik entspricht in allen Konstruktionen der semantische Kopf dem syntaktischen Kopf. Fügen Sie entsprechende Konindizierungen in die Typedefinitionen für *head-initial* und *head-final* ein.

(b) Flexionsregeln verändern die Relation nicht, die von der Tochter beigesteuert wird. Stellen Sie die Verfügbarkeit des SEM-Wertes am Mutterknoten durch entsprechende Koindizierung der SEM-Werte von Mutter und Tochter in der Definition von *word* sicher.

(c) Laden Sie die Grammatik neu und entfernen Sie Fehler (falls es welche gibt). Stellen Sie sicher, daß die Abdeckung gleich bleibt und daß der RELS-Wert am 'S'-Knoten die Relationen aller Wörter im Satz enthält.

Was jetzt noch erreicht werden muß ist die Verbindung von syntaktischen Argumenten und semantischen Rollen (*Linking*). Wir werden den INDEX-Wert von Argumenten (auch einfach *Index* gennant) benutzen, um diese Verbindung explizit zu machen. Dazu müssen wir alle Lexeme erweitern, die mit Argumenten kombiniert werden, d. h., die eine nicht-leere SUBCAT -Liste oder einen nicht-leere MOD-Liste haben: Wir müssen den INDEX-Wert von jedem Argument mit einer semantischen Rolle in der Relation des Funktors (dem semantischen Kopf) koindizieren.

(a) Nomina identifizieren den INDEX des Determinierers mit ihrem eigenen INDEX (und damit auch mit ihrem ARG0).

(b) Alle Verben identifizieren den INDEX ihres ersten Arguments mit der ARG1-Rolle. Zusätzlich identifizieren bivalente Verben den INDEX ihres zweiten Arguments mit ihrem ARG2. Bei ditransitiven Verben kommt dann entsprechend noch eine Verbindung zu ARG3 hinzu.

(c) Modifizierende Präpositionen identifizieren den INDEX der *syn-struc*, die über MOD selegiert wird mit ihrem eigenen Index. Der INDEX des Elements in der SUBCAT -Liste wird mit ARG1 identifiziert. (Komplementpräpositionen werden vorerst ignoriert.)

(d) Laden Sie die Grammatik neu, stellen Sie sicher, daß alles funktioniert und bewundern Sie die Schönheit der semantischen Komposition. Überprüfen Sie die folgenden Wohlgeformtheitsbedingungen in den Semantikrepräsentation von Sätzen wie *Der Hund bellt.* und *Die Katze gab dem Hund das Schaf.*: (i) Alle Indizes sind maximal spezifisch, d. h. entweder *event* oder *object*, (ii) in allen Nominalphrasen teilen Determinator und Nomen die ARG0-Variable, (iii) alle Rollen von verbalen Relationen sind an die Indizes der korrespondierenden Argumente gebunden und (iv) die ARG0-Variable einer modifizierenden Präpositionalphrase ist an das ARG0 des *event* bzw. des *object* gebunden, das modifiziert wird.

3. Das LKB-System erlaubt es, semantische Formeln aus Merkmalstrukturen zu extrahieren und sie in verschiedenen Formaten auszugeben, (eingeschränkt) logische Schlüsse durchzuführen und von den Formeln ausgehend zu generieren. Da unsere Semantik zur Zeit ziemlich einfach ist, ist nur ein Teil dieser Funktionalität sinnvoll nutzbar.

   Wählen Sie in dem Menü, das Sie erhalten, wenn Sie mit der linken Maustaste in einen kleinen Syntaxbaum klicken, die Menüpunkte 'MRS' und 'Indexed MRS' aus, um eine lesbare Form der Bedeutungsrepräsentation zu bekommen, die durch die Grammatik aufgebaut wird.

4. Das LKB-System enthält einen Chart-basierten Generator, der von einer semantischen Formel ausgehend alle Wortfolgen generieren kann, die diese semantische Repräsentation haben. Damit der Generator funktioniert, müssen wir einige abschließende Änderungen an unserer Grammatik machen:

   (a) Aus Generator-internen Gründen müssen INDEX-Werte ein Merkmal INSLOC haben, dessen Wert vom Typ *instloc* ist. Fügen Sie den atomaren Typ *instloc* als Untertyp von **top** in die Grammatik ein und erweitern Sie die Definition von *index*. Laden Sie die Grammatik neu.

   (b) Um den vollen Zugriff auf den Generator zu bekommen, müssen Sie das Kommando 'Options – Expand Menu' ausführen und dann 'Generator – Index' aufrufen, um dem Generator das Lexikon zugänglich zu machen. Sollten hierbei Warnungen ausgegeben werden, überprüfen Sie Ihre Grammatik. Die automatische Indizierung gleich während das Ladens der Grammatik kann man einstellen, indem man in der Datei 'script' im Grammatikverzeichnis das Semikolon am Beginn der Zeile `;(index-for-generator)` entfernt.

   Wenn die Indizierung abgeschlossen ist, kann man in dem Menü, das man durch Klicken auf den kleinen Syntaxbaum bekommt, 'Generate' auswählen. Dann werden alle Wortfolgen mit der Semantik, die die angezeigte Struktur hat, generiert. Das ist bequeme Form der Eingabe der logischen Form einer vorher erzeugten Struktur in den Generator.

   Parsen Sie den Satz *Den Hund jagt die Katze.*. Generieren Sie von der entsprechenden Bedeutungsrepräsentation ausgehend alle möglichen Sätze. Überlegen Sie, woran es liegen könnte, daß so viele ungrammatische Sätze von der Grammatik erzeugt werden.