Einführung in die Grammatikentwicklung: Übung 4 (21. Oktober 2003)

Ziele:

1. Erweiterung der Grammatik zur Behandlung von Modifikation.
2. Einführung von Adjektiven.
3. Etwas über Grammatikregeln und Prinzipien lernen.
4. Erfassen von Generalisierungen im Lexikon.

Übungen:

1. LKB starten und Grammatik4 laden:
   (a) Setzen Sie sich ins Verzeichnis Grammars, indem Sie in einem xterm-Fenster `cd Grammars` eingeben.
   (b) Geben Sie
   `cvs checkout Grammatik4`
   ein.
   (c) Starten Sie emacs und geben Sie im Emacs-Fenster folgendes ein:
   `<Esc> x lkb`
   (d) Laden Sie die Grammatik, indem Sie Load / Complete grammar im 'Lkb Top'-Fenster auswählen und dann mit Doppelklick das Verzeichnis 'Grammatik4' und die Datei 'script' auswählen.
2. Erweitern Sie die Grammatik um eine Analyse der Modifikation, so daß Sätze wie *Bellt der Hund neben der Katze?* analysierbar werden. Führen Sie dazu ein neues Kopfmerkmal MOD und eine neue Grammatikregel für Modifikatoren ein.
   (a) Fügen Sie in der Datei types.tdl das Merkmal MOD zur Definition des Typs *pos* hinzu. Der Wert wird auf **list** beschränkt, denselben Typ, den auch SUBCAT hat.
   (b) Weisen Sie in types.tdl jedem Untertyp von *pos* einen entsprechenden MOD-Wert zu. Nominalgruppen im Genitiv können andere Nomina modifizieren (1a) und es gibt auch Nominalgruppen im Akkusativ, die Verben modifizieren (1b).

   (1) a. der Pullover des Mannes
   b. Er schläft den ganzen Tag.

   Diese Verwendungsweisen sollen in der Übungsgrammatik unberücksichtigt bleiben.
   (c) Fügen Sie in der Datei rules.tdl eine neue Regel head-modifier-rule ein, die der head-complement-rule ähnelt, in der aber der Modifikator die Kopftochter selegiert.
   (d) Fügen Sie in der Datei lexicon.tdl einen Eintrag für *neben* ein.
   (e) Speichern Sie Ihre Änderungen ab und testen Sie die revidierte Grammatik mit der Datei 'mod.items'. Sehen Sie sich das Ergebnis an und machen Sie die nötigen Änderungen.
   (f) Wenn Ihre Grammatik die Phrase *der Hund neben der Katze* nicht zuläßt, erweitern Sie die Grammatik entsprechend.
   (g) Fügen Sie weitere Sätze in die Datei 'mod.items' ein. Beobachten Sie, wie sich die Anzahl der Lesarten verhält, wenn Sie die Anzahl der Präpositionalphrasen im Satz vergrößern.
   (h) Finden Sie heraus warum der Satz *Hilft der Hund dem Hund neben der Katze?* vier statt wie erwartet zwei Lesarten hat.
   (i) Überlegen Sie, wie man die unerwünschte Struktur ausschließen könnte.
3. Erweitern Sie die Grammatik um pränominale Adjektive, so daß Wortgruppen wie *die kluge Katze* analysierbar werden.

(a) Fügen Sie in der Datei types.tdl eine entsprechende Definition für Adjektive (Typ *adj*) ein. Der MOD-Wert von Adjektiven muß so spezifiziert werden, daß die Phrase *kluge die Katze* ausgeschlossen ist, d. h. Adjektive dürfen nur Nomina modifizieren, die noch nicht mit einem Determinator kombiniert wurden.

(b) Fügen Sie in der Datei lexicon.tdl einen Eintrag für *kluge* ein.

(c) Fügen Sie in der Datei rules.tdl die Regel modifier-head-rule ein. Diese Regel ist analog zur head-modifier-rule, nur die Reihenfolge der Töchter (der Elemente in ARGS) ist vertauscht.

(d) Die Datei parse-nodes.tdl enthält Beschreibungen, die das LKB-System dazu benutzt, die Bäume darzustellen. Fügen Sie einen entsprechende Beschreibung für Adjektive ein, damit diese in den Bäumen korrekt angezeigt werden.

4. Diejenigen von Ihnen, die mit der HPSG vertraut sind, werden sich wundern, warum es nirgends ein Kopfmerkmalsprinzip gibt. Auch die, die nicht mit HPSG vertraut sind, haben eventuell bemerkt, daß in jeder Regel der HEAD-Wert der gesamten Phrase identisch mit dem HEAD-Wert eines der Elemente in ARGS ist. Das Argument, das den HEAD-Wert für die gesamte Phrase beisteuert, wird Kopf der Phrase (*head*) genannt. Für einige Phrasen ist die erste Tochter der Kopf, für andere die letzte Tochter. Verändern Sie die Spezifikation der Regeln, so daß diese Unterscheidung in kopfinitiale und kopffinale Phrasen erfaßt wird.

   (a) Fügen Sie die folgenden Typen in types.tdl ein:

   ```
   head-initial := phrase &
    [ HEAD #head,
      ARGS < [ HEAD #head ] ...> ].

   head-final := phrase &
    [ HEAD #head,
      ARGS < syn-struc, [ HEAD #head ] > ].

   root-hf := root & head-final.
   root-hi := root & head-initial.
   ```

   (Zu den Typen *root*, *root-hf* und *root-hi* wird später im Kurs noch etwas gesagt werden.)

   (b) Verändern Sie die Regeln in rules.tdl so, daß sie von diesen Typen erben. Die head-complement-rule sollte z. B. wie folgt aussehen:

   ```
   head-complement-rule := head-initial &
   [ SUBCAT #1,

     ARGS < [ SUBCAT < #2 . #1 > ],
            #2 > ].
   ```

   Kopffinale Regeln wie die complement-head-rule sollten von *head-final* erben.

5. Wenden Sie dieselbe Strategie an, um weitere redundante Spezifikationen in der Grammatik zu finden und zu eliminieren. Verbessern Sie die Organisation der Typhierarchie, so daß es einfacher wird, neue Wörter, die bereits vorhandenen ähneln, zur Grammatik hinzuzufügen. Betrachten Sie als erstes die Einträge für Nomina und stellen Sie fest, daß viele Lexikoneinträge gleiche Spezifikationen haben. Beschreiben Sie diese Generalisierungen als Beschränkungen für einen neuen Typ *noun-word*, von dem alle Lexikoneinträge für Nomina erben. Nutzen Sie den Batch-Parse-Mechanismus, um sicherzustellen, daß sich durch Ihre Änderungen keine Fehler in die Grammatik einschleichen.