Department of Computer Science
CSC 247/447 and BCS 235
Fall 2006

Lisp Assignment 2

Due: 3:15pm, Thursday, Oct. 12 (hand in to TA)

Head (and Head Word) Hunting

The goal is to find the head constituent(s) of any phrase given in Treebank form; this will then also enable finding the head word(s) of any constituent. For example, the head of

```
(NP (NP (NNP ATLANTA)) (POS 'S) (JJ RECENT) (NN ELECTION))
```

(*Atlanta's recent election*) is `(NN ELECTION)`, whose head in turn is `ELECTION` (the word itself), which therefore is also the head word of the original NP. For reasons about to be explained, however, the results should be `((NN ELECTION))` and `(ELECTION)` respectively, i.e., extra brackets should be added. As a second example, the head of

```
(S (NP (NNS cells) )
   (VP (VBN coalesced)
       (PP (IN into) (NP (NNS organisms)))))}
```

is `(VP (VBN COALESCED) (PP (IN INTO) (NP (NNS ORGANISMS))))`, and the head word is COALESCED, so for head extraction you should get `((VP (VBN COALESCED) (PP (IN INTO) (NP (NNS ORGANISMS)))))` and for head word extraction, (COALESCED).

Call your functions `head-of` and `head-word-of`. Obviously, the latter should make use of the former. For phrases with multiple heads (esp. coordination), a list of heads (and similarly, head words) should be returned. Therefore, even in the case where there is just one head, the result should have an extra pair of brackets wrapped around it (so that the length of the list returned tells us the number of heads). A link to a sizable sample of Treebank data is given below. The symbols used therein are explained below.

One small complication is this: Treebank trees often have complex proper nouns in them, such as *United States*. These are formed into NPs that have multiple "NNP" (proper noun) constituents. You should collapse these into single atoms. We don't need to identify heads in such multi-word proper nouns. For example, you can't really expect to figure out the structure of *Fulton County Grand Jury*, which strictly consists of a phrase *Fulton County* with head *County*, modifying *Grand Jury* with head *Jury*, because the Treebank "analysis" is just (NNP Fulton) (NNP County) (NNP Grand) (NNP Jury). So this should just be turned into an atom FULTON_COUNTY_GRAND_JURY, of lexical category NNP. If the last word of such a sequence is a plural proper noun (NNPS) (e.g., as in the case of *United States*), the category of the atom should also be NNPS. The reason for forming an atom rather than a list is that we don't want to get confused between multiple heads and multi-word heads.

This assignment is a step towards a larger goal (to be pursued in the remaining Lisp assignments) of performing pattern-matches on sentences in Treebank form, and using the result of the match to generate new sentences (again in Treebank form). This will

enable you, for instance, to generate simplified versions of sentences (e.g., with modifiers omitted), or to generate responses in the manner of an ELIZA-like agent, or even to generate inferences from certain sentences (e.g., inferring "*The cat is dead*" from "*The cat died*", both in Treebank form). (This is in the spirit of using natural language itself as a "logical" representation, which is not far removed from certain approaches to NL semantics, such as ones based on categorial grammar.)

Format of Treebank Trees

You can see plenty of examples of treebank trees (taken from the 1,000,000 word Brown corpus) at

http://www.cs.rochester.edu/~schubert/247-447/assignments/brown-extract

Study these, to get a feel for the kinds of phrase structures assumed. The following are the meanings of the symbols.

First of all, here are the phrase types used in Treebank annotations; these are mostly self-explanatory:

S, SBAR, NP, VP, PP, ADJP, ADVP, WHNP, WHPP, WHADVP, SINV.

Unfortunately SBAR stands for a variety of clause-like phrases, namely: S[that] (i.e., complementizer *that* followed by a sentence), clausal adverbials (e.g., *since she left*), clausal wh-nominals (e.g., *what she knows*), and relative clauses (e.g., *whom she despises*). WHNP stands for NP[wh] (e.g., *which house*), and similarly for WHPP, WHADVP. SINV is S[inv], i.e., an inverted sentence (in question formation, e.g., "*Is she here?*.

Next, here are the lexical categories (parts of speech tags) used in Treebank, along with more conventional symbols, descriptive phrases, and examples:

CC	CONJ[cc]	coordinating conjunction	*and*
CD	DET[num]	count determiner; A[num]?	*many, five*
DT	DET	ordinary determiner (sing or sing/plur)	*the, a, some*
EX	PRON[there]	existential pronoun	*there*
FW	FW	foreign word	*absurdo, Unsinn*
IN	P	preposition	*with, at, in*
JJ	A	adjective	*silly*
JJR	A[-er]	comparative adjective	*sillier*
JJS	A[-est]	superlative adjective	*silliest*
LS	LS	list element indicator	*(1), a.*
MD	V[mod]	modal auxiliary	*will, must, may*
NN	N[sing]	singular common noun	*mouse, milk*
NNS	N[plur]	plural common noun	*mice, cats*
NNP	N[name,sing]	singular proper noun	*Joe, Norway, United*
NNPS	N[name,plur]	plural proper noun	*Sioux, States*
PDT	DET[plur]	plural-only ordinary determiner	*all, both*
POS	POS	possessive indicator	*'s*
PRP	PRON	personal pronoun	*I, we, she, it, him*
PRP$	DET[poss]	possessive "pronoun" (determiner)	*my, their*
RB	ADV	adverb	*quickly, very, not*
RBR	ADV[-er]	comparative adverb	*faster, more, less*
RBS	ADV[-est]	superlative adverb	*fastest, most, least*

RP	PART	particle	*up, in*
SYM	SYM	symbol	Ω
TO	V[to]	infinitive *to*	*to*
UH	INTERJ	interjection	*uh, no, right*
VB	V[base]	verb in base form	*see, describe*
VBD	V[past]	verb in past tense	*saw, described*
VBG	V[-ing]	verb in progressive form	*seeing*
VBN	V[-en]	verb in past/passive participle form	*seen, described*
VBP	V[pres,plur]	verb in present plural form	*are, vote*
VBZ	V[pres,3per,sing]	verb in present, 3rd person singular form	*is, sees*
WDT	DET[wh]	wh-determiner	*which, what*
WP	PRON[wh]	wh-pronoun	*who, what*
WP$	DET[wh,poss]	possessive wh-determiner (possessive "pronoun")	*whose*
WRB	ADV[wh]	wh-adverb	*how, when, why*
\.	PUNC[final]	sentence-final punctuation	. ? !
\,	PUNC[comma]	comma	
\:	PUNC[pause]	punctuation signalling a pause or break	*; : –*
$	N[meas]	the dollar sign in front of dollar amounts	*$*

There is also a special category symbol -NONE- for empty constituents.

How to find heads ...

This is left largely to your own understanding and ingenuity. Keep in mind that the head of a sentence is the VP, the head of a VP is the first verb, the head of an NP is the last noun (preceding any PP, relative clause, or other postmodifiers), the head of a PP is the P, the head of an AP (ADJP) is the A, the head of an ADVP is the ADV or the PP (depending on how the ADVP is formed), the head of an inverted sentence is the fronted auxiliary verb, and the head of an SBAR is the S. Keep in mind that you are to treat compound proper nouns as single atoms of category NNP or NNPS. However, in the case of NN NN ... premodification sequences (i.e., common nouns rather than proper nouns, give head-finding a try (even if you can't be sure of getting it right). Recall that a premodifying nominal can itself be a premodified nominal, and therefore also has a head.

Write-up, etc.

The usual principles of good program design and documentation apply:

- The documentation should describe and explain the structure of the programs; point out potential problems or subtleties you noticed and whether (and how) you dealt with them, and any noteworthy features of your approach or program (graders tend to assign credit for good ideas!)
- The programs should contain headers briefly explaining what they do (including what the input parameters mean and what sorts of output are produced), and comments in the code where this is helpful;

- The programs should be as "elegant" as possible; i.e., they should be concise, clearly structured, and make good use of the constructs of LISP such as recursion and MAP functions (and other "fell swoop" functions) where appropriate.
- The programs should be efficient, especially if there is no great price to be paid in elegance; for instance, it is a good idea to avoid repeated searches of fixed lists by using property lists, hash tables or arrays. (This may or may not be relevant in this assignment.)
- Give plenty of examples of the outputs; in the present assignment, for example, it would be interesting to see various examples similar to the one mentioned illustratively above, to show that you are handling all the cases that can arise as well as possible.
- State your conclusions about your head hunter, and any ideas you have about how it could be improved.