

Computational Linguistics (Spring 2009)

— Final Exam —

General Instructions

- Please read through the complete exam once before starting to answer questions. About thirty minutes into the exam, the instructor will come around to answer any questions of clarification (including English terminology).
- Please follow the instructions closely. Most of the questions ask for short answers. When a maximum length is given (e.g. ‘in no more than two sentences’), please try to stick to those limits.
- As discussed in class, the exam is only given in English, but you are free to answer in any of *Bokmål*, English, or *Nynorsk*.

1 Finite-State Technology (10 + 10 + 10 + 20 = 50 Points)

- Draw the finite-state automaton (FSA) that corresponds to the regular expression $(ab|b^*)^+$.
- For each of the following strings, say whether it is part of the language recognized by this automaton or not: (i) *bbb*, (ii) *bab*, (iii) *abab*, (iv) *aabaab*.
- Recall the equivalence relation holding between non-deterministic and deterministic FSAs. What is the characteristic of a non-deterministic automaton, and is your solution to part (a) above deterministic or not?
- In a few sentences, sketch the basic idea in converting a non-deterministic FSA into a deterministic one. In the worst case, how many states can there be in the new, deterministic FSA, assuming that the original, non-deterministic FSA had n states.

2 Reverse English (15 + 15 = 30 Points)

Consider the language defined by the following grammar (assuming the conventional start symbol ‘S’):

$S \rightarrow VP NP$	$NP \rightarrow kim$
$VP \rightarrow PP VP$	$NP \rightarrow oslo$
$NP \rightarrow PP NP$	$NP \rightarrow snow$
$VP \rightarrow NP V$	$V \rightarrow adores$
$PP \rightarrow NP P$	$P \rightarrow in$

- For each of the following items, identify the number of readings (distinct analyses) that the grammar of Mirror English assigns:
 - oslo in snow adores kim*
 - kim adores snow in oslo*
 - snow adores oslo in kim*
- Draw the constituent tree for each of the readings.

3 Unification-Based Grammar (10 + 20 = 30 Points)

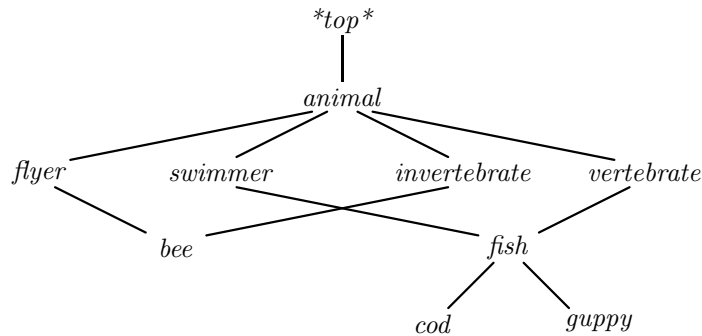
- (a) Show the feature structure representation of the following rule as it is used in our LKB grammars:

$$\text{phrase} \left[\begin{array}{l} \text{HEAD} \boxed{1} \\ \text{SPR} \langle \rangle \\ \text{COMPS} \boxed{3} \end{array} \right] \rightarrow \boxed{2} \text{ phrase} \left[\begin{array}{l} \text{SPR} \langle \rangle \\ \text{COMPS} \langle \rangle \end{array} \right], \quad \text{phrase} \left[\begin{array}{l} \text{HEAD} \boxed{1} \\ \text{SPR} \langle \boxed{2} \rangle \\ \text{COMPS} \boxed{3} \end{array} \right]$$

- (b) What is the (approximate) name of the above rule in our grammar? Sketch its functionality in a few sentences and provide two examples of types of phrases built using this rule.

4 Typed Feature Structures (10 + 20 + 10 + 20 = 60 Points)

- Assume the following type hierarchy (which should look familiar):



- (a) Which of the following pairs of types can unify? For each pair, name their greatest lower bound (aka glb) type.

- animal* and *bee*
- flyer* and *invertebrate*
- flyer* and *vertebrate*
- swimmer* and *whale*

- (b) Following is the description of a type hierarchy (taken from one of our grammars) in the Type Description Language (TDL). Draw the graph corresponding to this hierarchy, i.e. a diagram of super- and sub-type relationships in the same format as the ‘animal’ hierarchy above. For the purpose of this exercise, we ignore all declarations of appropriate features on types.

```

pos := *top*.
agr-pos := pos.
det := agr-pos.
modable := agr-pos.
noun := modable.
verb := modable.
premodifier := pos.
postmodifier := pos.
prep := postmodifier.
adj := premodifier.
adv := premodifier & postmodifier.
  
```

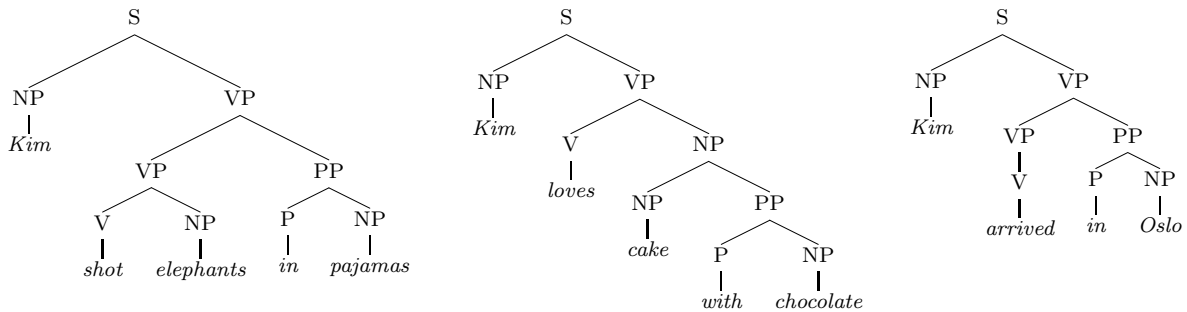
- In this hierarchy, are there any pair(s) of types for which the greatest lower bound (i.e. the result of unifying two types) is not uniquely defined? If so, which are these?
- In two or three sentences, what is the interpretation of the vertical dimension of the type hierarchy, i.e. what does it mean for a type to be a sub-type of another type? In this light, using one sentence each, explain any two out of the following four terms *specificity*, *inheritance*, *monotonicity*, or *subsumption*.

5 Linguistic Notions (10 + 15 + 15 = 40 Points)

- (a) In no more than five sentences, comment on the differences between specifiers and complements, on the one hand, and modifiers on the other hand. Use one or two examples.
- (b) In your own words, describe how the following sentences of English are ambiguous—for example paraphrasing each possible interpretation in English—and intuitively identify the nature of each ambiguity. Note that we are primarily interested in syntactic ambiguity here). *click*.
- (i) *the dog chased that cat near those aardvarks*
- (ii) *the chasers of the cat barked*
- (c) Sketch constituent trees for each of the possible readings you have identified for sentences (i) and (ii) above, using abbreviatory node labels like ‘Det’, ‘N’, ‘NP’, ‘VP’, et al. and annotating each branch as to whether the constituent dominated by it acts as a *head*, *specifier*, *complement*, or *modifier*.

6 Probabilistic Context-Free Grammars (20 + 20 = 40 Points)

- (a) When adapting CFGs to Probabilistic Context-Free Grammars, we made one extension to the elements of the original CFG and added an additional condition on the elements of the set P of productions. Sketch the formalisation of a context-free grammar (as a quadruple) that we used in class, and summarize the revisions that we made in extending CFGs to PCFGs.
- (b) Assuming the following (tiny) ‘treebank’ of correct parse trees of English.



In a sentence or two, what is the general procedure for estimating the rule probabilities of a PCFG? For each of the following rules, estimate rule probabilities from the treebank above (i.e. making the naïve assumption that such a tiny treebank was sufficient for the task).

P(RHS LHS)	Context-Free Rule
	S → NP VP
	VP → VP PP
	VP → V NP
	PP → P NP
	NP → NP PP
	VP → V