University of Toronto, Department of Computer Science
CSC 2501/485F—Computational Linguistics, Fall 2015

Assignment 2

Due date: 13h10, Tuesday 27 October 2015, on CDF.
This assignment is worth 18% of your final grade.

- Submit the ID file on the course website along with your submission.
- Please type your reports in no less than 12pt font; diagrams and tree structures may be drawn with software or neatly by hand.
- What you turn in must be your own work. You may not work with anyone else on any of the problems in this assignment, except for discussion in very general terms. If you need assistance, contact the instructor or TA.
- Any clarifications to the problems will be posted on the course bulletin board. You will be responsible for taking into account in your solutions any information that is posted there, or discussed in class, so you should check the page regularly between now and the due date.

1. A simple context-free grammar of English [30 marks]

Your task is to develop two context-free grammars for the chart parser in NLTK, each covering an interesting and not-entirely-trivial subset of English, along with a lexicon and a set of test sentences. In the first step, you'll build a base, which you'll then use for separate grammars in each of the next two steps. After testing each grammar, you'll write a report on what it can and can't do.

1.1 Very simple sentences

To get started, begin with very simple sentences consisting of a subject noun phrase (NP), an intransitive verb in simple past tense (like *ate* or *arrived*), plus some modifiers. To begin introducing recursive rules in your grammar, allow the NPs to be modified by prepositional phrases as well as adjectives. You should be able to parse sentences such as the following. (Note that punctuation and sentence-initial capitalization should not be used in the test sentences; but words that are names can always be capitalized.)

```
Nadia left immediately
the cat with the long soft fur slowly ate
she arrived
```

You should not, however, accept the following kinds of sentences. (Remember that '$*$' means a sentence is ungrammatical.)

`*Nadia with the long soft fur slowly ate` *(Can't attach a PP to a name)*
`*the cat with the tall her arrived` *(Pronoun can't take an adjective)*

These are obviously not the only sentences your parser and grammar should accept or reject. Part of the point of the assignment is for you to decide what grammar you need in order for your parser to correctly accept or reject a range of constructions of this *kind*. Note that this does not mean simply listing lots of different words in the lexicon; rather, you need to think about *similar grammatical constructions*.

1.2 The auxiliary system

Agreement is one of the complex phenomena in natural languages that makes writing good grammars difficult. This shows up in a particularly interesting way in the auxiliary system in English. In the next stage of development, extend your grammar to handle valid sequences of auxiliaries and verbs in English, such as those below.

```
Nadia will leave
Nadia has left
Nadia may have been leaving
*Nadia will left
*Nadia has could leave
*Nadia has had left
```

See lecture notes #4 and Jurafsky & Martin §12.3.6 for additional possible combinations of auxiliaries.

Note: To keep things simple for the next part of the grammar, in section 1.3, keep one rule `S -> NP VP` for sentences containing verbs in the simple past tense (without auxiliaries).

1.3 Subcategorization

Next, increase the coverage of your base grammar to deal with verbs other than those that are intransitive. A very important aspect of language (in which one constituent places strong constraints on other constituents) is the phenomenon of *subcategorization*. Subcategorization refers to the specification that a particular word places on the possible complements (objects) with which it can occur (see lecture notes #4).

Verbs take a wide variety of combinations of complements. Here are some example sentences your parser should be able to handle correctly; you should also think of other types of verbs (or get ideas from Jurafsky & Martin, §12.3.5) that you should be able to parse.

```
Nadia fondled the eggplant
the handsome poodle brought Ross to the autoclave
Nadia brought a package for the cheese
they told her to jump onto the elephant
she believed that Ross was already on the hovercraft
she really wanted help
she really wanted to help
cheese was always on the menu
the eggplant reminded Nadia of Ross
*Nadia found
*Ross brought to him
*they told to jump onto the elephant
```

There are many other complement possibilities (and impossibilities!).

Note: To keep your grammar a manageable size, do **not** try to combine the rules for the auxiliary system from section 1.2 with the rules for verbs with different subcategorizations. That is, **DO NOT** add grammar rules that generate any verbs other than intransitives in different forms for occurring with different auxiliary combinations. Verbs occurring with complements should only be generated in the simple past tense, from the rule `S -> NP VP`.

Note: In lecture notes #4, we look at more-complex verb subcategorizations than these — for example, involving embedded clauses (*Nadia expected Ross to promise to remind her to leave*) — and we develop the idea of feature-based grammars to account for them. In this question, we want to keep things much simpler. So you should use regular context-free rules, not features, to account for subcategorization. Features are featured in question 2, below.

1.4 Testing

Your grammar should handle all the test cases given in this handout, which are available at `www.cs.toronto.edu/~frank/csc2501/Assignments/A2-test.txt`. The list of words that they use is available at `www.cs.toronto.edu/~frank/csc2501/Assignments/A2-vocab.txt`; note that some words can have more than one syntactic

category and hence will have more than one entry in the lexicon. You are encouraged to add words to this set for your own testing. Any words.

In addition to the sentences given here, you will need to develop an additional set of your own test sentences to fully demonstrate that the grammar meets its specifications and to reveal over- and undergeneration — that is, types of sentences that it shouldn't accept but does, and types of sentences that it should accept but doesn't.

You will need to write testing code that uses NLTK to parse your test sentences with your grammar. A simple way to use NLTK's chart parser is demonstrated in the following:

```
from nltk import parse_cfg
from nltk.parse import ChartParser, BU_STRATEGY

grammar = parse_cfg("""
S -> NP VP
PP -> P NP
NP -> DET N | N | NP PP
VP -> V NP |  VP PP
DET -> 'the'
N -> 'Nadia' | 'man' | 'eggplant'
V -> 'rewarded'
P -> 'with'
""")

sentence = "Nadia rewarded the man with the eggplant".split()

parser = ChartParser(grammar, BU_STRATEGY)

for t in parser.nbest_parse(sentence):
  print t
```

This produces two parses for the given sentence:

```
(S
  (NP (N Nadia))
  (VP
    (V rewarded)
    (NP
      (NP (DET the) (N man))
      (PP (P with) (NP (DET the) (N eggplant))))))
(S
  (NP (N Nadia))
  (VP
    (VP (V rewarded) (NP (DET the) (N man)))
      (PP (P with) (NP (DET the) (N eggplant)))))
```

Refer to the NLTK API documentation for details. Your test code will also need to read input and write output, as described in section 1.6.4 below.

1.5 Limitations

At this point, you'll be able to parse lots of interesting sentences. But there will be lots of sentences, even ones similar to those above, that you *won't* be able to accept or reject correctly, because your grammar will necessarily be limited. For example, you aren't asked to implement subject–verb agreement.

In your report, you should discuss the kinds of limitations that still exist in your final grammar, including shortcomings revealed by your tests. This section of your report is not intended to cover every aspect of English with which you cannot deal correctly (that would be a very long report). Rather, you should focus your report on constructions that are very similar to those with which you have been asked to deal. For example, aspects of NPs and VPs would be reasonable to mention.

1.6 Implementation details

In order for the grader to be able to semi-automatically test your work, each file must have the exact name and format specified in the following subsections.

Some overall specifications for your files:

- The first line of each file must be a comment with your name, login ID, and student ID.
- Each grammar rule, lexical entry, or sentence must appear on a separate line in its appropriate file.
- The symbol `%` at the beginning of a line in your grammar, lexicon, and sentence files should indicate a comment.
- You should organize and comment your grammar and lexicon, just as you would your code, to make it easily understandable.

1.6.1 Grammar

Name of the file: `Grammar`
An example:

```
% Your name, login ID and student ID go here.
S -> NP VP
NP -> NPrp
NP -> Det N
% NLTK allows abbreviations like the following
VP -> V | V NP
VP -> V NP PP
PP -> P NP
```

The start symbol for the grammar must be S. Please use reasonable names for the other grammar symbols (nonterminals and parts-of-speech), following the conventions below. You might need symbols that aren't listed here—devise reasonable names for them based on your reading from the textbook or the terms we use in class.

Nouns: NP, N, NPrp (proper noun), NPro (pronoun), NDem (demonstrative pronoun)
Verbs: VP, V, Aux, Modal
Adjectives: AdjP, Adj
Prepositions: PP, P
Other: Det (article or determiner), Dem (demonstrative determiner), Adv (adverb)

1.6.2 Lexicon

Name of the file: `Lexicon`
An example:

```
% Your name, login ID and student ID go here.
Det -> 'a' | 'an'
N -> 'elephant' | 'rutabaga' | 'autopoiesis' | 'shot'
NPro -> 'i'
NPrp -> 'Nadia' | 'Marseilles' | 'Google'
V -> 'won' | 'smiled' | 'demanded' | 'shot'
```

Note: If a word has *n* possible parts-of-speech (lexical category designations), then it is listed *n* times in the lexicon, once for each part-of-speech, as with *shot* above.

1.6.3 Test sentences

Name of the file: `Sentences`
An example:

```
% Your name, login ID and student ID go here.
Nadia won an elephant
I could have demanded a rutabaga
autopoiesis always reminded her of Marseilles
```

Note: The sentences should not contain any punctuation marks.

1.6.4 Output parse trees

Your output parse trees should be in the pretty-print format already provided by NLTK. If a sentence is not accepted by the grammar, the file should contain the line `No parses`. All your output should be directed to a file name named `ParseTrees`. The file should have your name, ID, and student number as the first line. It should then contain each sentence and the parses, as shown below:

Name of the output file: `ParseTrees`
An example output file:

```
% myGivenName myFamilyName, myLoginID, 999999999
I saw Nadia on a boat with my elephant
(S
  (NP I)
```

```
  (VP
    (VP (VP (V saw) (NP Nadia)) (PP (P on) (NP (Det a) (N boat))))
    (PP (P with) (NP (Det my) (N elephant)))))
(S
  (NP I)
  (VP
    (VP (V saw) (NP Nadia))
    (PP
      (P on)
      (NP
        (NP (Det a) (N boat))
        (PP (P with) (NP (Det my) (N elephant)))))))
(S
  (NP I)
  (VP
    (VP (V saw) (NP (NP Nadia) (PP (P on) (NP (Det a) (N boat)))))
    (PP (P with) (NP (Det my) (N elephant)))))
(S
  (NP I)
  (VP
    (V saw)
    (NP
      (NP (NP Nadia) (PP (P on) (NP (Det a) (N boat))))
      (PP (P with) (NP (Det my) (N elephant))))))
(S
  (NP I)
  (VP
    (V saw)
    (NP
      (NP Nadia)
      (PP
        (P on)
        (NP
          (NP (Det a) (N boat))
          (PP (P with) (NP (Det my) (N elephant)))))))))
```

1.7 What to submit

Provide a single PDF or Word docx comprising the following items, in the following order:

- A written report describing the design of your grammar and lexicon, and how they meet the grammatical requirements stated above. Don't forget to also discuss the limitations of your grammar; see section 1.5 above.
- A written report on your testing strategy—in particular, why you chose the sentences that you used for testing the grammar and lexicon. This should be no more than one page.
- A printout of the `ParseTrees` file that you generate for your test sentences.
- A printout of your input files (`Grammar, Lexicon, Sentences,` in that order).

Note: You do *not* need to submit the code that you use to run your tests, but you are certainly welcome.

In addition, you must submit your grammar and your output. Please include:

- All the input files (`Grammar, Lexicon, Sentences`) that you used to test your parser.
- Your output file, `ParseTrees`

Submit all required files using the `submit` command on CDF:

```
% submit -c <course> -a A2 <filename-1>...<filename-n>
```

where `<course>` is `csc485h` for undergraduates and `csc2501h` for graduate students, and `<filename-1>` to `<filename-n>` are the *n* files you are submitting. Make sure every file you turn in contains a comment at the top that gives your name, your login ID on CDF, and your student ID number. Also submit the provided ID file.

Grading scheme

We will test your grammar on the examples in this handout as well as on some held-out test sentences (i.e., sentences that you haven't seen (e.g., [redacted])).

Grammar 1: simple sentences	5 marks
Grammar 2: auxiliaries and modals	5 marks
Grammar 3: subcategorization	5 marks
Your testing (including output parses): meeting specifications; tests of overgeneration and undergeneration	10 marks
Your report: description of grammars and their limitations	5 marks
Total	**30 marks**

2. Using features in grammars [10 marks]

Grammar 1 handles NPs of various different types. Grammar 2 is much simpler and easier to read, but doesn't appropriately constrain the NPs generated.

Grammar 1
Rules:
S → NPnom VP
VP → V NPacc
PP → P NPacc
NPnom → NP
NPnom → PROnom
NPacc → NP
NPacc → PROacc
NP → Det N
NP → Det N PP
NP → Npl
NP → Npl PP
N → Nsg
N → Npl

Lexicon:
She: PROnom
fed: V
the: Det
dog: Nsg
puppies: Npl
him: PROacc
with: P

Grammar 2
Rules:
S → NP VP
VP → V NP
PP → P NP
NP → N
NP → Det N
NP → Det N PP
NP → N PP

Lexicon:
She: N
fed: V
the: Det
dog: N
puppies: N
him: N
with: P

A. (1 mark) Show the parse tree for the following sentence according to Grammar 1:

She fed the dog with puppies with him.

B. (7 marks) Augment Grammar 2 with features that capture the intended restrictions on NPs in the Grammar 1.

C. (2 marks) Show the parse tree for the sentence in part A above, using your new Grammar 2. Annotate each constituent with the features assigned by your grammar.

Hint: The mnemonics *nom* and *acc* stand for *nominative* and *accusative*, i.e., subject and object.