AI2 Practical 3 (Module 2)
Parsing SMS text messages

Deadline: 4pm, Friday, 3 December 2004

General Instructions

This document describes the first assessed assignment for AI2 Module 2. For this assignment, you will be creating several files. You should put them all in a single sub-directory which contains only the work for this assignment. All filenames must be as specified in this document. The format of the submission command for this assignment is:

```
submit ai2 ai2a A4 DIR
```

where `DIR` is the name of the sub-directory in which you have put all your work. Make sure that if you are specifying a relative pathname that you are in the right directory to do so.

Some Comments on Plagiarism

1. Submitting another student's code (even if modified) as if it is your original work is plagiarism.
2. Assisting another student to plagiarise (eg. by sharing code without the borrowing student acknowledging the use) is also penalised.
3. Discussing the assignment in broad terms is ok, but not at the level of coding details.
4. If you cannot figure out how to code some portion of your program, you **could** borrow someone else's code, **provided** you clearly acknowledge whose code it was and what portion of your submission is theirs. You will not get credit for the other person's code.
5. Even partial, non-working submissions get partial credit.
6. A failed assignment does not mean a ruined career. Getting caught at plagiarism might.
7. We use various techniques to detect plagiarism, including automated tools .
8. If you can't do the assignments, discuss this with the course organiser or your director of studies.
9. Change the protections on your homework files and directories so potential plagiarists cannot access your solution. If your file is called XXX, then use the unix command: `chmod go-rwx XXX`. Don't assume that no one would ever plagiarise from you.

SMS Text Messaging

Most of you are familiar with SMS text messaging, whether it is SMS texting in English, French or Punjabi. While the grammar underlying SMS text is basically the same as the grammar underlying the standard language – for example,

SMS English: sum1 sum where dreamz of ur smiles.
standard English: *Someone somewhere dreams of your smiles.*

SMS English: y r u l8?
standard English: *Why are you late?*

SMS English: c u 2morrow!
standard English: *See you tomorrow!*

new coinages are constantly being added to the SMS lexicon. Fortunately, people adept at texting can usually figure out what word or phrase each new coinage represents.

In this assignment, you will look at parsing SMS texts, and in particular, at some problems a machine would have with dealing with new SMS coinages. At the end, you should

1. have a better understanding of the standard parser for Definite Clause Grammars (DCGs), as well as the CKY and Earley chart parsing algorithms;
2. be more aware of why you often don't notice that an utterance is ambiguous;
3. know a bit more about English grammar.

Task 1: A Lexicon for SMS texts (20%)

The file `sms-dcg.pl` (shown at the end of this handout) contains the grammar that you will use in this assignment. It doesn't cover all of English, but it should be enough for all the texts that you will consider. Sentences that this grammar can recognise/generate include:

Later I played my favorite guitar.
I saw the film before you read the book.
The two boys slept and john waited until they awoke.

These sentences could be translated into the following SMS texts:

L8r I played my favorite guitar.
I saw d film b4 u red d book.
d 2 boys slept n john w8d til they awoke.

That is, SMS texting uses a combination of words from the source language (here, English), along with its own new coinages.

Q1.1. Add to the lexicon in `dcg_sms.pl` entries for the SMS coinages in the table below, so that the sentences above (and others) can be recognised by the same grammar. (If you don't know whether a word is a noun or a preposition or a conjunction, etc., look it up in a dictionary. Remember that a word can have different senses, each with a different part of speech.)

N.B. Prolog doesn't accept terms starting with numbers, so you will have to quote "4get" and "2moro" in your lexicon entries.

SMS	Standard English	SMS	Standard English
1	one	l8r	later
2	to, two	n	and
2moro	tomorrow	nuf	enough
4	for, four	s	these
4get	forget	r	are
b4	before	til	until
c	see, sea	u	you
cuz	because	ur	your
d	the	w8	wait
l	although	w8ed	waited
l8	late	red	read, red

At this point, your lexicon contains words from standard English and SMS coinages. (You can add other SMS coinages as well as those given in the above table.)

Q1.2. Construct three different sentences that the above grammar and your extended lexicon can recognise. Make sure each sentence uses a different rule for `s` expansion and a different rule for `vp` expansion. Make sure each sentence can be successfully recognised.

Include each sentence as a comment at the bottom of your `dcg_sms.pl` file, in a form that can be evaluated – e.g.

```
% s([i, saw, d, film, b4, u, red, d, book], []).
```

To be submitted:

- Your augmented version of `sms-dcg.pl`, with three example sentences (in comment lines) at the end. (This constitutes the answers to **Q1.1** and **Q1.2**.)

Task 2: Parsing utterances containing unknown words (60%)

New coinages are a problem for grammars, and new coinages are frequent in SMS texts.

Q2.1 What happens when you try to use the standard DCG parser to parse a sentence that contains a word that isn't in your lexicon? (Try it and describe what happens.)

One way for a grammar to deal with a word it hasn't seen before is to (1) assume that it belongs to *every part of speech* in the grammar (i.e, `adj`, `adv`, `conj`, etc.) and (2) see which of them produces a recognisable sentence of the language. If a sentence doesn't have too many words that are outside the lexicon, this isn't a bad strategy, especially if you are using a chart parser. This is because a chart parser efficiently computes all possible parses.

Q2.2 Choose whatever example you want to use as illustration. It should have *a single unknown coinage* (i.e., one that isn't in your lexicon). Hand simulate how the strategy would work with a CKY parser. In your example, what *parts of speech* for the unknown word find their way into an analysis of some *substring* of your example? Which *parts of speech* for the unknown word find their way into a *complete analysis* of the example as a sentence? If either figure is less than the full set of 13 different *parts of speech* used in the grammar, what does this say about the difference between the *potential* ambiguity of an unknown word and the *actual ambiguity* found by the parser?

Q2.3 With the same example as illustration, hand simulate how this strategy would work with an Earley parser. Does the Earley parser use the same *parts of speech* in its substring analyses? Does it find the same complete analyses?

Q2.4 Finally, comment briefly which algorithm does more work in dealing with the unknown word and why.

To be submitted:

- A single file containing your answers to Q2.1 – Q2.4

Task 3: Dealing with SMS Phrases (20%)

SMS texts often contain coinages for *phrases* as well as for individual words, for example

btw	by the way
fyi	for your information
hi	here is
hru	how are you?
10q	thank you
wyw	while you were
wim	where is my

Some of these coinages correspond to individual *phrases* in the grammar (e.g., btw and fyi both correspond to prepositional phrases, while hru corresponds to a full sentence). Other coinages span parts of phrases (e.g., hi spans the noun “here” and the copula verb “is”.)

3.1 Write rules that allow pp to be realised as btw or fyi.

3.2 Comment briefly on how the grammar could be made to handle coinages like hi and wyw, which don’t correspond to existing rules. What kind of rules could you add? How could you determine that the rules you added would parse hi and wyw in all and only the same ways as the original rules would parse the sequences “here is” and “while you were”.

To be submitted:

- A single file containing your answers to Q3.1 and Q3.2.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%  Base Grammar for Assignment A4 -- Parsing SMS text messages
%  (Version 2)
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

s --> s0, subconj, s.
s --> s0, conj, s.
s --> s0.
s0 --> adv, s.       % eg "Frequently John eats potatoes."
s0 --> pp, s.        % eg "In January, John eats potatoes."
s0 -->  np, vp.

np --> npr.         % proper name, eg "shrek", "edinburgh"
np --> pro.         % pronoun, eg "someone", "i", "he"
np --> det, nom0.
np --> nom0.
np --> det, nom2.

nom0 --> num, nom1.  % an NP can have one number premod -- eg "the two girls"
nom0 --> nom1.       % or no number premod -- eg "the girls"
nom1 --> adj, nom1.  % an NP can have >=1 adj premods -- eg "two big red eyes"
nom1 --> nom2.
nom2 --> n.          % an NP can have 0 pp postmods -- eg "the two girls"
nom2 --> nom3, pp.   % or one or two pp postmods.
nom3 --> n.
nom3 --> n, pp.

pp --> prep, np.

vp --> iv.
vp --> tv, np.
vp --> dv, np, np.   % for vps like "give Shrek a potato", "read Shrek a story"
vp --> dv, np, pp.   % for vps like "give a potato to Shrek", ...
vp --> copula, np.   % for vps like "is the winner", "are great films"
vp --> copula, adj.  % for vps like "are great", "is enough"

  %%%%%%%%%%%%%%%%%
  %%   Lexicon   %%
  %%%%%%%%%%%%%%%%%

%% Adjectives
adj --> [Word], {adj(Word)}.
adj(favorite).
adj(great).

%% Adverbs

adv --> [Word], {adv(Word)}.
adv(frequently).
adv(later).
adv(today).
```

```
%% Conjunctions

conj --> [Word], {conj(Word)}.
conj(and).

%% Copulas

copula --> [Word], {copula(Word)}.
copula(am).
copula(is).

%% Determiners

det --> [Word], {det(Word)}.
det(my).
det(the).

%% Ditransitive verbs

dv --> [Word], {dv(Word)}.
dv(gave).
dv(loaned).
dv(read).

%% Intransitive verbs

iv --> [Word], {iv(Word)}.
iv(awoke).
iv(played).
iv(slept).

%% Nouns

n  --> [Word], {n(Word)}.
n(book).
n(boy).
n(film).
n(guitar).

%% Number.

num --> [Word], {num(Word)}.
num(one).
num(two).
num(three).
num(four).

%% Prepositions.

prep --> [Word], {prep(Word)}.
prep(in).
prep(with).
```

```
%% Pronouns

pro --> [Word], {pro(Word)}.
pro(i).
pro(someone).
pro(they).

%% Proper nouns

npr --> [Word], {npr(Word)}.
npr(ai2).
npr(john).
npr(shrek).

%% Subordinate conjunction

subconj --> [Word], {subconj(Word)}.
subconj(because).
subconj(before).
subconj(until).

%% Transitive verbs

tv --> [Word], {tv(Word)}.
tv(played).
tv(read).
tv(saw).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%% The symbol "distinguished" is used to identify what can be the root
%% of complete grammatical analysis. Usually in English, it is the
%% sentence (s), though in spoken language, one also finds utterances
%% interpretable in context that are simply nps, vps, pps, etc.
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

distinguished(s).
```