CSCI 3813/780 – Spoken Language Processing
Prof. Rosenberg

Homework 3
Due November 2, 2011 @ 3:00pm

Note: This assignment is **all** group work. There is only one individual work component, see #6 in the Submission section.

Speech Understanding

In this homework, you will design and build your own speech understanding system for the same TTS domain you chose in Homework 2. This system will be used to map an input spoken utterance to the concepts (degrees of freedom) in your domain. You should assume as input a grammatical English utterance (single sentence) in question form. The system will consist of two main components.

1) **The speech recognition system.**

Download and install CMU Sphinx
http://cmusphinx.sourceforge.net/

Follow the tutorial that can be found here:
http://cmusphinx.sourceforge.net/wiki/tutorial

You will need to download and install:

- sphinxbase
- sphinx4
- US English acoustic models
- A US English language model
- You may also want sphinxtrain and cmuclmtk

The sphinx code is downloaded from here:
http://cmusphinx.sourceforge.net/wiki/download/

The models are available here:
https://sourceforge.net/projects/cmusphinx/files/Acoustic%20and%20Language%20Models/

This tutorial should help a lot:
http://cmusphinx.sourceforge.net/wiki/tutorialsphinx4

Note: Installing programs with this complexity can take a non-trivial amount of time. Start on this early, making sure that you have something working sooner rather than later.

2) **The spoken language understanding system.**

The input to this component is the ASR transcript produced by SPHINX.
The output will be a table that contains a (possibly partial) set of components that represe nts the query. You can structure this however you like with as many fields unspecified as makes sense in your domain. For example a reservation system may know about times, dates, destinations and origins, but you want to allow questions that reference only some of these items. You will need to include vocabulary that allows question answering.

For example, let's assume your TTS domain involves train reservations and contains the following degrees of freedom:

- Departure city: New York, Boston, Baltimore, Newark, Jersey City, Washington, Albany, Poughkeepsie, Pittsburg, Philadelphia
- Destination city: (same set of cities as above)
- Departure day: Sunday, Monday, Saturday
- Departure Time: Morning, Noon, Afternoon, Evening, Night, Anytime

Here are some examples, given the following utterances:

1. What time can I travel from Boston to New York on Friday?

Departure City	Boston
Destination	New York
Day	Friday
Time	UNSPECIFIED

2. What trains leave from Washington on Monday evening?

Departure City	Washington
Destination	UNSPECIFIED
Day	Monday
Time	Evening

3. Is there a train from Washington to New York on Monday evening?

Departure City	Washington
Destination	New York
Day	Monday
Time	Evening

You should create a grammar that covers as many different ways of asking questions for your domain as you can think of. The goal is to make your system as flexible as possible. However, your grammar should be limited enough so that the ASR perplexity is not so high as to affect performance. You should experiment to see what trade-offs between flexibility and performance work best. Part of the homework assignment is to see how well you can determine how much you can cover with reasonable performance. Note that your success will be judged on concept accuracy, not transcription accuracy.

You will use a Grammar for the language model with Sphinx. (see this page for information: http://cmusphinx.sourceforge.net/wiki/tutoriallm) To make your life easier use capital letters for all words.

Grammar Format:

> Variables are surrounded by angle brackets, e.g. <city>
> X Y is concatenation e.g. I WANT
> X | Y means X or Y , e.g. (WANT | NEED)
> [X] means X is optional e.g. [ON] FRIDAY

Example grammar covering above examples

```
<city> = BOSTON | NEWYORK | WASHINGTON | BALTIMORE;
<time> = MORNING | EVENING;
<day> = FRIDAY | MONDAY;

public <sentence> = SENT-START ((WHAT TRAINS LEAVE) | (WHAT TIME CAN I
TRAVEL) | (IS THERE A TRAIN)) (FROM | TO) <city> [(FROM | TO) <city>] ON
<day> [<time>] SENT-END
```

Assignment Requirement: Record at least five (5) test utterances from your domain as wav files for each team member. For best recognition, leave ~1 second of silence at the start and ~1 sentence at the end when recording. Call the files test[1-5].wav. Evaluate the performance of the recognizer on these five files.

Note: If your domain contains unusual words or proper nouns, you may have to modify the default dictionary.

Assignment Requirement: Now you must write a program that takes a recorded wav file, runs the ASR system, and generates concept tables as in the above examples. Each table should consist of concepts and their values separated by tabs with one concept value pair per line.

Example: Input test2.wav
```
RecognizeConcepts.sh test2.wav
```

Output (to the command line or a <u>user specified</u> filename):

```
    Departure City:     Boston
    Destination:        New York
    Day:                Friday
    Time:               UNSPECIFIED
```

It is critically important that your program be runnable on other machines and system configurations. **Therefore, do not encode any paths or filenames directly in your compiled code.** The ***only*** exception to this rule is a single configuration to this file. Your program may read a configuration file that you may assume to have a static name and be in the same directory as the program itself. Your documentation **must** define how to set up the configuration file so that your system can run in a new environment. Things that may be important to include in the configuration file include but are not limited to: 1) the grammar file, 2) the location of the sphinx executable, 3) a dictionary file.

Your system will be tested on the wav file you provide for your domain, and other files that are generated using questions covered by your grammar. Your grade will be based on grammar coverage and concept accuracy.

Note: in the next project, you will be creating a dialog system combining the ASR and TTS systems for your domain. This may require modifications to the ASR grammar and to the TTS system. You should be thinking about how you will combine these systems, and what changes might be needed to construct your grammar for the ASR.

Note: The work you do in this assignment will be used in the term project. The broader your grammar, the better coverage your dictionary has, and the more reliable your submission is at this point, the less work you will have to do in the future when preparing the final project.

Submission

You should submit the following:

1. A readme file that
    a. Explains how to run your program with a command line example. Do not build a GUI interface to your program
    b. Explains the coverage of your grammar and any heuristics you employed, tradeoffs you made or other interesting aspects of your approach
    c. Describes which acoustic models work best for you and why you chose it
2. The five input wave files that you've tested with your ASR system. This should include five files for **each team member** asking a different question in all wave files.
3. The grammar for use in sphinx.
4. (optional) a dictionary file if your domain requires it.
5. Your program that runs the ASR and extracts concepts.
6. A file prepared **individually** by each team member, describing how each team member (including yourself) contributed to the submitted project. Note: **do not write one description for the whole team and submit it N times, one for each team member.** This need not take very long to write. However, do not collaborate with your teammates when preparing this write up.