Instructions
There are 8 questions, each will be worth 12.5 points. The maximum score on the test will be 100. You will have approximately 1:50 minutes to complete this test (50% more time than you had for the midterm). If you feel that your test is complete, you may hand in your test and leave early. It is essential that you PUT YOUR NAME ON ALL TEST MATERIALS. It can be difficult identify the author of an unsigned test and it would be better to avoid this problem. The test materials will include this printout and one blank test booklet. I suggest that you fill in all answers directly on this printout and use the blank test booklet as scrap paper. However, if you run out of space, you have the option of using the test booklet. However, please include a clear note on the test so I know where to look for your answer. This test is an open book/open notes test: Please feel free to bring your text book, your notes, copies of class lectures and other reading material to the test. A calculator is also permitted. However, please do NOT: communicate with others (texts, email, etc.) or run actual programs to calculate answers.

Answer all questions on the test. If you show your work and you make a simple arithmetic mistake, but it is clear you knew how to do it, you will get partial credit.

Note that this is a sample test. It is intended to approximate the content of the actual test. Of course questions on the actual test may end up being more or less difficult for you.

Questions
(1) Draw a Phrase Structure Tree representing one parse of the following sentence. Use Penn Treebank POS tags (see table above) as the terminal nodes with unary branches down to the words (similar to the trees used for question 7).

*Eleanor Rigby picks up the rice in the church where a wedding has been.*
(2) Write a set of phrase structure rules in Chomsky Normal Form that are equivalent to the following rules, i.e., both sets of rules generate the same language. Leave out rules going from parts of speech to words. For illustration purposes, sample words for each part of speech are provided.

1. \( S \rightarrow NP \; VP \)
2. \( NP \rightarrow DT \; ADJP \; NN \; NNS \)
3. \( NP \rightarrow DT \; ADJP \; NN \; NN \)
4. \( ADJP \rightarrow RB \; JJ \)
5. \( VP \rightarrow RB \; VBZ \; NP \)
6. \( VP \rightarrow VBZ \)

Part of speech examples:

- \( NN \): book, clam, chicken, idea, truck, human
- \( NNS \): books, clams, chickens, ideas, trucks, humans
- \( DT \): the, a, an, every, each, this, that
- \( JJ \): angry, red, blue, dark, similar
- \( RB \): very, slightly, slowly, hesitantly, possibly
- \( VBZ \): eats, sees, flies, buys
(3) Calculate cosine similarity between the query and the list of documents below, given the vectors provided below. Each vector provides weights for the terms: city, Spain, United States, paella, gumbo, shellfish, beer. Note that paella is a Spanish rice dish.

**Vectors**

- Vector for Query *Which Spanish city has the best paella?*: (10,7,0,20,0,5,0)
- Wikipedia Entry for Valencia, Spain: (7,30,1,25,0,5,0)
- Wikipedia Entry for Seville, Spain: (7,25,3,0,0,6,0)
- Wikipedia Entry for New Orleans, USA: (7,5,30,0,20,10,0)
(4) Do the following semantic role labeling annotation task. Each Attribute noun takes up to two arguments: a theme and a value, both of which are optional. For example, in the phrase:

*The building’s HEIGHT of at least 25 feet.*

the word height (in all capital letters) is an attribute noun referring to an attribute of its theme (the building) and the value of that attribute is *at least 25 feet*. The attribute noun describes a characteristic of its theme (ARG1), e.g., height names a possible characteristic of a building. The value (ARG2) provides a measure for that characteristic and/or answers *What is the ATTRIBUTE of* questions as specified in the text, e.g., *What is the height of the building?* could be answered with *at least 35 feet*.

In each of the following examples, assume that the word in all capital letters is an ATTRIBUTE noun. Circle any theme arguments that you see and label them ARG1. Circle any value arguments that you see and label them ARG2. Remember both ARG1 and ARG2 are optional. So some examples may have one ARG1 and one ARG2. Others may have just an ARG1 and still others may have just an ARG2.

1. *Mergers often trigger longer lunch hours and increased ABSENTEEISM.*

2. *Unice is an ACRONYM for the Union of Industrial and Employers’ Confederations of Europe*

3. *He assumed that post at the AGE of 35*

4. *the song’s original BEAUTY comes through*

5. *Excluding the higher tax RATE, the company would have met analysts’ expectations*
You are writing a rule based component to handle out of vocabulary items for a POS tagger. The input is a token. Your rule detects tokens that are likely to be past tense verbs (VBD). The answer to this question is a regular expression that captures all and only: (a) words whose last three letters are a consonant followed by *ed*, e.g., *computed*, *averred*, *red*; and (b) words that share endings with the word lists below. The part of your rule that matches the ending of the word should consist of a series of individual patterns separated by vertical bars, i.e., \((\text{pattern1})- (\text{pattern2})- \ldots-(\text{patternN})\).

These ending patterns should be between 3 and 5 letters long and should be based on the common prefixes of each set of words. Ending patterns that differ by exactly one letter should be merged using square brackets containing the letters that vary, e.g., '123[ab]cd' represent the disjunction of '123acd' and '123bcd'.

- *bethought bought brought fought outfought overbought rethought sought thought wrought*
- *taught caught*
- *gainsaid inlaid laid mislaid overlaid overpaid paid prepaid said underpaid waylaid*
- *forwent forwent lent misspent overspent rent sent spent underspent underwent went*
- *clung flung hamstrung rung hung overhung slung strung stung swung wrung*
- *betook forsook mistook overtook partook retook shook took undertook*
- *misunderstood stood understood withstood*
- *bore foreswore forswore outwore overbore swore tore wore*
- *drove test-drove strove wove dove interwove*
- *rewrote smote underwrote wrote*
- *bespoke broke spoke woke*
- *bestrode outrode overrode rode strode*
- *foretold resold retold sold told undersold*
- *held beheld held upheld withheld*
Calculate B-cubed precision, recall and F-score given the coreference entities in the system output and answer key. **Note:** Only consider the NPs that are part of the answer key and system output. Other NPs in the passage are irrelevant. A with subscripts indicate one answer key entity and B with subscripts indicate another answer key entity. The actual context is provided below with the entities marked in square brackets.

**Answer Key**:

1. $A_1$=Sarah Palin, $A_2$=Palin, $A_3$=her, $A_4$=the 2008 GOP vice presidential nominee
2. $B_1$=Nebraska state Sen. Deb Fischer, $B_2$=she

**System Output**:

1. $A_1$=Sarah Palin, $A_2$=Palin, $A_3$=her, $B_2$=she
2. $B_1$=Nebraska state Sen. Deb Fischer

The above entities were taken from the following passage from *Sarah Palin weighs in on Texas, Neb. Senate races*, by Catalina Camia, USA Today, May 10, 2012. The NPs that are part of the entities are in bold.

**Sarah Palin**[$A_1$] is getting more active in key Senate races, endorsing Republican candidates in Texas and Nebraska ahead of their upcoming primaries.

**Palin**[$A_2$] today gave **her**[$A_3$] support to Ted Cruz, a former Texas solicitor general, who is in a crowded field running to succeed retiring Sen. Kay Bailey Hutchison.

**Nebraska state Sen. Deb Fischer**[$B_1$], meanwhile, announced on Wednesday **she**[$B_2$] received the **2008 GOP vice presidential nominee**[$A_4$]’s support.
(7) Figures 1 and 2 provide the parse trees for two consecutive sentences. Annotate the trees with arrows indicating the search for the antecedent of the possessive pronoun *her* using the Hobbs search algorithm, diagrammed as figure 3. Put an X through each NP that was considered as an antecedent, but was rejected (due to agreement, semantics, etc.). Circle the antecedent found by the Hobbs Search. Next to each arrow, indicate the number of the step associated with this move (as per figure 3).
1. Go to NP immediately dominating pronoun

2. Go up to 1st dominating NP or S node. This node = X path to X = p.

3. Traverse branches below X to the left of p, left to right and breadth first. Propose each NP n as an antecedent if there is an NP or S between n and X.

4a. Is X the highest S in the sentence? (Recursive)

4b. Search previous sentences in order from right to left. Search each tree from left to right, breadth 1st, proposing each NP as antecedent

5. From Node X, go up to the 1st NP or S. Call this node X and the path to X p.

6. If X = NP and there is no N’ in p, propose X as antecedent

7. Search for antecedent in branches below X, left to right, breadth first. Propose each NP.

8. If X is an S node, traverse all branches of X following p, left to right, breadth first, but not going below any S or NP node found. Propose each NP.

Figure 3: Hobbs Search Algorithm for Finding Antecedents of Pronouns
Given the following unigram and translation probabilities, what is the overall probability scores that a decoder would assign to *dog bad* being a translation of the Spanish *perro malo*?

What would need to be added to the model to get the correct word order?

**Unigram Probabilities for English:**

- *dog*: $1.7 \times 10^{-5}$
- *cat*: $8 \times 10^{-6}$
- *bird*: $5 \times 10^{-6}$
- *bad*: $1.6 \times 10^{-4}$
- *good*: $4.9 \times 10^{-4}$
- *stupid*: $1.1 \times 10^{-5}$

**Translation probabilities**

<table>
<thead>
<tr>
<th>English</th>
<th>perro</th>
<th>gato</th>
<th>pjarl</th>
<th>malo</th>
<th>bueno</th>
<th>estupido</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>dog</em></td>
<td>.6</td>
<td>.03</td>
<td>.01</td>
<td>.12</td>
<td>.15</td>
<td>.09</td>
</tr>
<tr>
<td><em>cat</em></td>
<td>.04</td>
<td>.55</td>
<td>.04</td>
<td>.15</td>
<td>.11</td>
<td>.11</td>
</tr>
<tr>
<td><em>bird</em></td>
<td>.01</td>
<td>.02</td>
<td>.8</td>
<td>.03</td>
<td>.02</td>
<td>.12</td>
</tr>
<tr>
<td><em>bad</em></td>
<td>.01</td>
<td>.03</td>
<td>.02</td>
<td>.87</td>
<td>.05</td>
<td>.02</td>
</tr>
<tr>
<td><em>good</em></td>
<td>.03</td>
<td>.01</td>
<td>0</td>
<td>.11</td>
<td>.83</td>
<td>.02</td>
</tr>
<tr>
<td><em>stupid</em></td>
<td>.05</td>
<td>.05</td>
<td>.05</td>
<td>.75</td>
<td>.05</td>
<td>.05</td>
</tr>
</tbody>
</table>