# EDAN20
# Final Examination

Pierre Nugues

October 18, 2010

The examination is worth 144 points. The distribution of points is indicated with the questions. You need 70% to have a mark of 4 and 85% to have a 5.

## 1 Closed Book Part: Questions

**In this part, no document is allowed. It is worth 62 points.**

**Chapter 1.** Peedy is a 3D animated character designed by Microsoft that responds to spoken commands to change discs. Figure 1 shows it architecture. Describe, notably in terms of input and output, the modules: whisper, names, NLP, semantics, and dialogue, or imagine the purpose. For each module, name one relevant technique to carry out the processing

5 points

**Chapter 1.** Describe why speech and language processing is difficult. 2 points

**Chapter 2.** Describe what a concordance is and give the concordances of the string *jord* in the text below.

> Den siste av arbetarna, Luis Urzua, var då befriad från underjorden. Den 54-årige förmannen Urzua kom upp till jordytan nästan 23 timmar efter den först räddade kamraten Florencio Avalos. Urzua blev den som fick stanna längst tid i gruvan efter olyckan för 69 dygn sedan, den 5 augusti då ett ras spärrade in de 33 arbetarna. Han är därmed också den som överlevt längst under jord efter en gruvolycka.
> Web site of *Svenska dagbladet*, November 14, 2010.

2 points

**Chapter 2.** Describe what a finite–state automaton is. Try to give the formal definition. 3 points

**Chapter 2.** Describe the main purpose of regular expressions. Write and describe two regular expressions including two different repetition metacharacters. 2 points

**Chapter 2.** Describe what the regular expressions `<b>.*</b>` and `<b>.*?</b>` do and give an example of what they match. 2 points

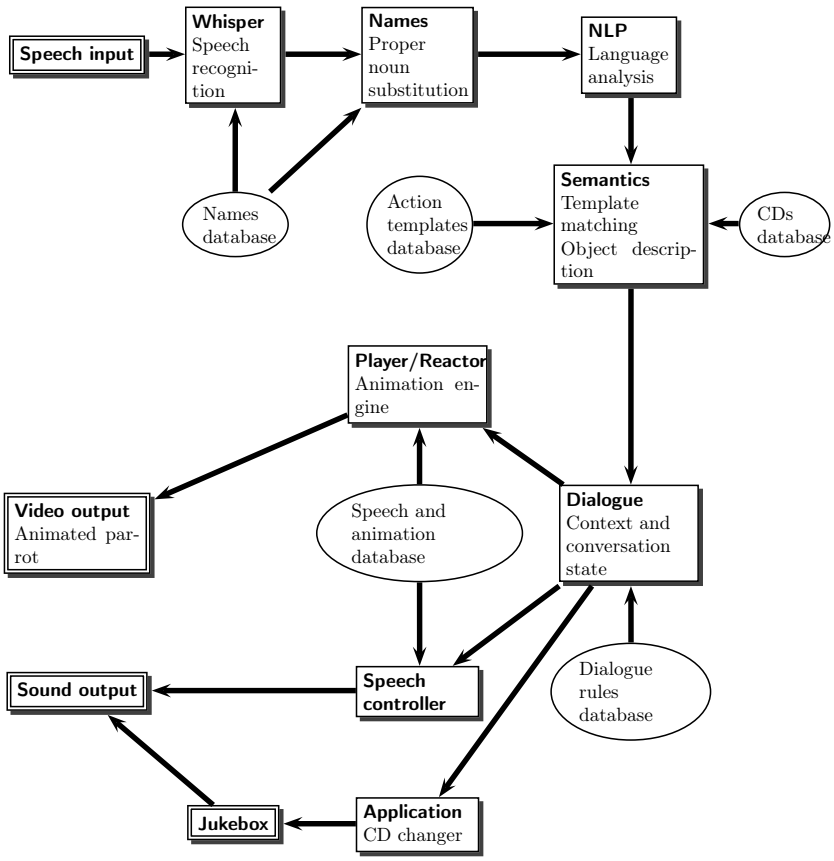


Figure 1: Architecture of the Persona conversational assistant. After Ball et al. (1997).

**Chapter 2.** What does this Unix command mean:
`tr 'A-Z' 'a-z' <file` 1 point

**Chapter 2.** What does this Unix command mean:
`tr -cs 'A-Za-z' '\n' <file` 1 point

**Chapter 2.** Give three edit operations used in approximate string matching. 2 points

**Chapter 2.** In approximate string matching, define what is the minimum edit distance? 2 points

**Chapter 2.** In approximate string matching, how is the minimum edit distance computed? 2 points

**Chapter 3.** What is the mathematical definition of entropy? 3 points

**Chapter 3.** What is a decision tree? 1 point

**Chapter 3.** In machine learning, what is a set of examples – a data set? How are these examples organized? (Think of the ARFF file structure.) 2 points

**Chapter 3.** Give an outline of the ID3 algorithm to induce a decision tree from a set of examples. 3 points

**Chapter 4.** Give the mathematical formula of the probabilistic model of the sentence: *Den siste av arbetarna* and how it is expressed using a product of conditional probabilities. 2 points

**Chapter 4.** Give the unigram approximation of this probabilistic model with the example *Den siste av arbetarna*. 2 points

**Chapter 4.** Give the bigram approximation of this probabilistic model with the example *Den siste av arbetarna*. 2 points

**Chapter 4.** Describe how the bigram probabilities are estimated and computed. 2 points

**Chapter 4.** Let us imagine that the bigram *sista av* has not been seen before in the corpus. Give one method to estimate its value (for instance the one used in the first laboratory). 1 points

**Chapter 6.** Using the parts of speech: article, noun, pronoun, modal, and verb, annotate the two sentences: *The can rusted* and *I can do it*. 2 points

**Chapter 6.** How an automatic system would decide the part of speech of the word *can*. Give a brief description only. You can choose the method you want. 2 points

**Chapter 6.** What is a confusion matrix? 1 point

**Chapter 7.** In machine translation, what is an alignment of parallel corpora?

**Chapter 8.** What are the **noun phrases** in the sentence *The big boy hit the ball*. 1 point

**Chapter 8.** Write the phrase–structure rules describing the **noun phrases** in the sentence *The big boy hit the ball*. Using the DCG notation is recommended, but not compulsory. 2 points

**Chapter 9.** Annotate the noun phrases in the sentence *The big boy hit the ball* using the IOB notation. You can use either IOB or IOB2. 1 point

**Chapter 10.** What is a dependency graph? Give an example of it using the sentence *The big boy hit the ball*. 2 points

**Chapter 10.** Describe what is projectivity in a dependency graph. Modify the graph of the sentence *The big boy hit the ball* to create one nonprojective arc. All the other arcs will remain the same as in the previous question. 2 points

**Chapter 11.** Apply the shift/reduce algorithm to the sentence *The waiter slept* and show the steps the analysis. You will use the rules:

```
s --> np, verb.
np --> det, noun.
det --> [the]. noun --> [waiter]. verb --> [slept].
```

3 points

**Chapter 11.** Describe what are the extensions used in Nivre's parser to parse a dependency graph and create arcs. 2 points

**Chapter 14.** Describe what the rhetorical structure theory (RST) is and illustrate it with the example:

1. *Concern that this material is harmful to health or the environment may be misplaced.*
2. *Although it is toxic to certain animals,*
3. *evidence is lacking that it has any serious long-term effect on human beings.*

and the relations *elaboration* and *concession*. 2 points

# 2 Problem

**In this part, documents are allowed. It is worth 82 points.**

The objective of this part of the examination is to investigate and program a semantic parser – also called a semantic role labeler. We will follow a simplified version of the description of Johansson and Nugues (2008) to implement a predicate identifier and an argument identifier. This text is provided as a matter of reference and to satisfy a possible curiosity. You should not need to use it during the examination.

## 2.1 The Format

The CoNLL 2008 format is a syntactic and semantic annotation format. It uses columns to describe the index, words, lemmas, parts of speech, dependency structure, and finally the semantic predicates and arguments of a sentence. We will use a simplified version of it in this examination. The (simplified) annotation of the sentence

He believes in what he plays, and he plays superbly.

is shown in Table 1. Sentences are separated by a blank line.

Table 1: Simplified annotation of the sentence *He believes in what he plays, and he plays superbly.* in the CoNLL 2008 corpus.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | He | he | PRP | 2 | SBJ | _ | A0 | _ | _ |
| 2 | believes | believe | VBZ | 0 | ROOT | believe.01 | _ | _ | _ |
| 3 | in | in | IN | 2 | ADV | _ | A1 | _ | _ |
| 4 | what | what | WP | 6 | OBJ | _ | _ | A1 | _ |
| 5 | he | he | PRP | 6 | SBJ | _ | _ | A0 | _ |
| 6 | plays | play | VBZ | 3 | PMOD | play.01 | _ | _ | _ |
| 7 | , | , | , | 2 | P | _ | _ | _ | _ |
| 8 | and | and | CC | 2 | COORD | _ | _ | _ | _ |
| 9 | he | he | PRP | 10 | SBJ | _ | _ | _ | A0 |
| 10 | plays | play | VBZ | 8 | CONJ | play.01 | _ | _ | _ |
| 11 | superbly | superbly | RB | 10 | MNR | _ | _ | _ | AM-MNR |
| 12 | . | . | . | 2 | P | _ | _ | _ | _ |

In this question, you will draw two graphs representing respectively the syntactic and semantic structures:

**Syntactic graph:** The first six columns are identical to the format used in dependency parsing and represent the dependency graph. The fourth column is the lemma.
Represent graphically (draw) these syntactic dependencies. 3 points

**Semantic graph:** The rest of the columns corresponds to the predicates and their arguments:

- The seventh column indicates the predicates. How many predicates are there in this sentence? Underline the predicates. 1 point
- The columns to the right of the predicates represent their respective arguments in their order in the sentence. How many arguments does the first predicate has, the second one, and so on?
- For each predicate, draw the arcs corresponding to its arguments. You will draw these arcs under the sentence. 2 points
- Although an arc points to a single word, each argument consists of phrase. This phrase corresponds to the syntactic dependents of the argument head word – the syntactic subtree starting from the pointed word. Draw a box around the two arguments of *believe* and of the second occurrence of *play*. 2 points

## 2.2 Parsing Pipeline

### 2.2.1 Description

As input data, the semantic system uses the dependency graph i.e., the first six columns. The parsing procedure consists of a pipeline of four classifiers, where

1. The first step is the identification of the predicate: Is this word a predicate or not?

2. The second step is the predicate disambiguation: If the word is a predicate, what is its sense number?

3. For each predicate detected in the first step: Is a word of the sentence an argument of this predicate or not? For the example sentence, this information is contained in the columns 8, 9, and 10 in Table 1.

4. If the word is an argument, what is its label?

All the predicates and their arguments (roles) are itemized in the Propbank database available to the system. The verb *play* in the sentence has nine possible senses in Propbank and *believe* has one. Table 2 shows the two first senses of *play* and the sense of *believe* as well as their arguments.

Table 2: The two first senses of *play* and the sense of *believe* in Propbank as well as their arguments.

| **Predicate** | **Description** | **Arguments** | **Description** |
|---|---|---|---|
| **play.01** | play a game | **Arg0** | player |
| | | **Arg1** | game/music |
| | | **Arg2** | instrument used to play game/music |
| **play.02** | play a role | **Arg0** | actor |
| | | **Arg1** | role |
| ... | | | |
| **believe.01** | believe | **Arg0** | believer |
| | | **Arg1** | believed |

### 2.2.2 Questions on the Pipeline Architecture

1. Draw graphically the processing pipeline using boxes. 1 points

2. The argument `AM-MNR` (manner) is not in the list above. Why and what does it means? 2 points

3. At each step of the pipeline, you will use a classifier trained on annotated data. Define and discuss for each step the set of values the classifier can output and the possible options to collect these values. 5 points

### 2.2.3 Programming

In this question, you will design and write a program to load a CoNLL 2008 corpus. Using the Java language is recommended.

1. Propose a class structure, preferably in Java, to represent a word. You will call it `Word` and you will only describe the fields (data members). As the number of columns of CoNLL 2008 files is variable, the most significant part is how to represent the arguments. 4 points

2. Write the class `CoNLLCorpus` to load the corpus. You will have to:

   - Propose a (Java) data structure to store a complete file consisting of a list of sentences, where each sentence consists of `Word`s.
   - Write the method that will load all the words and sentences of the corpus file. You will call this method `loadFile()`
   - Write a constructor for `Word`.

   15 points

## 2.3 The Predicates

### 2.3.1 Collecting the Predicate Features

The features used by Johansson and Nugues (2008) to identify whether a word is a predicate include: the part of speech of the word's parent (head), the function linking the word to the parent, the set of the parts of speech of the word's children, and the set of dependency functions of the word's children. All these features are extracted from the dependency tree.

1. Give the values of these four features for the three first words in the sentence as well what should be the decision of classifier. You will use the notation `{A, B, C}` to describe the set of elements `A`, `B`, and `C`. 4 points
2. Write a `Features` class that will represent the features of one word and write the methods to extract the four features. You can call them: `extractParentPposs()`, `extractFunction()`, `extractChildPpossSet()`, and `extractChildDepSet()`. 12 points
3. Write the code to create `Features` objects to collect the features for all the words of a sentence. 3 points
4. Once you have collected all the features of all the words in the corpus and stored them in a file, how would you train a classifier? 2 points

### 2.3.2 Identifying the Predicates

You have now trained a classifier and you are given a test set that has only the first six columns.

1. How will you extract the features from the test set and apply the classifier to determine whether a word is a predicate or not (Think of what you did during the labs). 2 points
2. The possible output of the ID3 (J48) classifier for the predicate identification is true or false. We will now suppose that the classifier outputs a probability together with the value. For instance, for the word *He*, the classifier output could be true, =.95 and false=.05. Imagine an output of probabilities for the example sentence: *He believes in what he plays, and he plays superbly.* 2 points
3. Using your probabilities, give the second best sequence? 3 points
4. Write a procedure that computes the $N$-best predicate sequences. You can use $N = 4$. 10 points

### 2.4 The Rest of the Pipeline

Describe how you would implement the rest of the pipeline to identify

1. the predicate sense, 3 points
2. the arguments, and 3 points
3. the argument labels. 3 points

## References

Ball, G., Ling, D., Kurlander, D., Miller, J., Pugh, D., Skelly, T., Stankosky, A., Thiel, D., Dantzich, M. V., and Wax, T. (1997). Lifelike computer characters: the Persona project at Microsoft Research. In Bradshaw, J. M., editor, *Software Agents*, pages 191–222. AAAI Press/MIT Press, Cambridge, Massachusetts.

Johansson, R. and Nugues, P. (2008). Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of CoNLL-2008: The Twelfth Conference on Computational Natural Language Learning*, pages 183–187, Manchester.