# Extraction of Lexico-syntactic Similarities

Due Date: 24/1

Questions: Meni

## Abstract

In this assignment you will experience with a research paper, modify its algorithm, re-design for map-reduce, and evaluate the results.

## Your Job

1. Read the paper of I. Dekang Lin and Patrick Pante: DIRT – Discovery of Inference Rules from Text.
2. Read the paper again, and make sure you understand it.
3. Examine the various aspects of the system, and identify components that may utilize the map-reduce pattern.
4. Design a parallel algorithm, based on the Map-Reduce pattern, according to the notes described bellow.
5. Implement the system.
6. Run the experiments described bellow.
7. Analyse and report your results, as defined bellow.

## Notes on the implementation of the article

- Regarding the second constraint at Section 3.2 (the second bullet at page 3): in contrast to MiniPar (the parser which is used in the paper), Stanford parser (the parser of your input) does not omit propositions from the tree nodes, so your dependency path should include dependency relations that connect prepositions (IN and TO, beside the content words: noun, verb, adjective, adverb).
- As mentioned in the paper, you should only refer to dependency paths in which the head is a verb and the X/Y slots are nouns.
- As mentioned in the paper, you should only refer to heads which are verbs. It is high recommended to filter auxiliary verbs - *e.g.*, is, are.
- Here you can find the list of the part of speeches that appear in the syntactic ngrams.
- As should be clear from the paper, there might be **some** dependency paths in a given syntactic ngram.
- The words in the Google Syntactic N-Grams dataset are not stemmed. For example, the words 'involve' and 'involves' are considered different words even though they share the same baseform - 'involve'. Since the predicates in the testset are given in a baseform, you should use a stemmer, which unifies suffix-based inflections of a same baseform (involve-involves, involve-involving). You can use any stemmer for English you wish, Porter Stemmer for example (Java code).

## The System

### Input

The input of the system is the Biarcs dataset of the syntactic NGRAM resource, recently released by Google (generated by Yoav Goldberg - one of the course's formers).

The format of the datasets is described in the README file. While extracting dependency paths and their arguments.

Note that the input files are not located in S3. I suggest that each student in the course will upload to files to his/her S3 account and you' share them all (with this shared table)

### Output

The output of the system should be composed of:

- The similarity measure between each pair of the given test-set.
- The *mi(p,Slot,w)* value for each path-slot-word found in the corpus (in order to support further similarity calculations of path pairs which are not included in the test-set).

## Test Set

Omer Levy kindly provided us a set of *positive predicate rules* and *negative predicate rules*. The files are tab-separated, where the proposition template on the left entails/not-entails the template on the left.

## Experiments

You should run the system on two types of inputs:

- **Small**: 10 file
- **Large**: 100 files (if you still have a budget)

## Reports

Your work should be followed by two reports:

- **Design**: You should provide a document which describes the design of your system: the various components of the system and their functionality in terms of input and output. In case a component is based on Map-Reduce, you should describe the characteristic of the keys and the values of the mappers and the reducers, with the number of the key-value pairs and their size, and an estimation of the memory usage
- **Analysis**
  - Calculate F1-measure for the given test-set, for each of the input sizes (small, large). Choose a threshold for the True/False decision, based on the similarity score, according to the data.
  - Draw a *Precision-Recall Curve*[1] for the given test-set, for each of the input sizes (small, large).
  - Error Analysis
    - Make a list of 5 examples for each true-positive, false-positive, true-negative, and false-negative categories.
    - Compare the similarity measures for these examples for the small and large inputs.
    - Try to identify common errors and behaviours.

[1] A *precision-recall curve* is a plot of the precision (y-axis) and the recall (x-axis) for different thresholds.

## Submission

You should submit the design document, the code, and the analysis report.
For the frontal check, you should come with the output of your system (feature vectors, similarity measures).

## Grading policy

- Article understanding - 30%
- System design - 25%
- System implementation - 25%
- Experiments and analysis report - 20%

**Good Luck!**