INLP, Definite Clause Grammar lab

Andrés F. Lamilla

January 14, 2015

Contents

1 Goal

For this exercise we have to build a DCG in prolog to recognize different dates format present in a text file. Each line in the text file has a date, but the format between the lines are different. Some of the dates extracted from the file are:

- birth_date [[December 24]], [[1960]]
- birth_date 1948|02|24
- birth_date August 5, 1958
- birth_date [[1821]]
- birth_date unknown

2 Code

I wrote three prolog files to do this exercise. There is a file called tokenizer.pl which has a function that receive a string and return a list of atoms. The file called parser.pl has all the DCG grammar rules to parse a date. Finally the file called labo3.pl read a text file and call the tokenizer and parser functions for each line in the file.

I also wrote two bash scripts to help with the execution and analysis of the prolog files. labo3_exec.sh execute the prolog file using SWI-Prolog. And get_stats.sh read the results from the prolog code and print the accuracy obtained with it.

The code was tested using SWI-Prolog Version 7.1.28.

3 Results

Beside of recognize whether a date is present in a text, the program returns the date in an established format. Some of the dates analyzed were the following:

Raw date	parsed date
birth_date [[December 7]], [[1964]]	date(1964,12,7)
birth_date [[626]] or [[627]]	or(date(626),date(627))
birth_date c. [[1412]]	date(1412)
birth_date 1965|1|26	date(1965,1,26)
birth_date Tuesday, [[April 4]], [[1943]]	date(1943,4,4)
birth_date Unknown	date(Unknown)
birth_date ~ [[1200]]	aprox(date(1200))
birth_date ~ 1117 AD	aprox(date(AD 1117))
birth_date 09/07/1957	date(1957,7,9)
birth_date about [[1000]] to [[1007]]	aprox(between(date(1000),date(1007)))
birth_date about 1961	aprox(date(1961))
birth_date [[February]], [[1980]]	date(1980,2,nil)

If the program don't find a date in the line, it print void followed by the line text.

To test the accuracy of the program we use a file called examples_birth_date.txt. This file has a total of 12296 lines with different dates format.

The program could recognize 12055 lines from the file and was unable to recognize 241 lines.

The accuracy was 98.04

3.1 FSA

In DCG it's more easy to define a rule than in FSA. Some of the dates that were not recognized using FSA were perfectly defined in DCG. Some of them are:

- birth_date [[2 Dec]] [[1946]]
- birth_date c. [[945]] [[950]]
- birth_date 1117 AD
- birth_date "c." [[419]]-[[422]]
- birth_date "ca". 69

The accuracy obtained with FSA was of 95.69. It may seem not a big difference with the obtained with DCG(98.04). But we have to think that it's very difficult to improve an already high accuracy.

4 Conclusion

With only a few rules it was posible to reach most than 80% of the accuracy. The last 18% needed much more rules.

I found that DCG in prolog is a really great tool to parse text and join different formats in a single one. It was very easy to write the rules and to process the result.

Given the flexible way of define the DCG rules in prolog it's possible to return dates in a nice format like:

date(1890,5,19), between(date(468),date(470)), etc.