Search Engine Tool For Local Documents

Final Year Project Proposal
September 2020

Jack Power
Applied Computing (IoT)
20080169

Introduction

I propose to create a document searching tool which utilises approximate string matching and conceptual searching. It will operate on documents stored locally on the user's device.

Most document viewers provide the user with the ability to search the text, usually limited to a basic string match, possibly offering globbing or regular expressions. When using something such as a web search engine, users are often subtly assisted. Search terms can be massaged into what the user likely meant and results may be returned which correspond to the semantic value of the search term. E.g. Searching for 'canine' may return results containing 'dog'.

Problem

It is often the case that a user wishes to extract only the relevant information from some document they are reading. When citing a paper or scanning a technical manual, we often do not have the luxury of reading hundreds of pages of unrelated material.

Solution

Fortunately the problem of string searching is endemic to computer science and a slew of algorithms and data structures are available to tackle it. I aim to integrate multiple search strategies, two of these are central to the desired behaviour.

Approximate string matching (fuzzy searching)

This is what enables the input massaging mentioned earlier. Given some particular input, small substitutions may be performed on the pattern to coerce it into a form that most closely matches the searched against text. (E.g. co**a**t → co**s**t)

Conceptual searching

Using semantic analysis to associate terms and concepts so that the meaning of a query is captured rather than simply the characters. (E.g. canine → dog)

Multiple algorithms exist for both, one advantage of this particular problem is that the text to be searched may be preprocessed in its entirety to build useful data structures to improve the search. This is often not the case with online searching performed by web search engines. (Navarro, 2001)

The finished product will consist of a program, its basic interface modelled after tools such as *grep* so that it may be easily integrated into a pipeline. Building upon this one could add a text user interface, a graphical user interface, or even integrate the program into your favourite text editor or document viewer as a plugin.

Notes

I expect that to get a satisfactory result the implementation will have to be tweaked repeatedly. Important criteria will of course include the accuracy of the results, as well as execution time and the size of the program, both on disk and in memory.

I aim to base my design on well-established patterns and idioms, and to use highly optimised implementations of the core algorithms.

In the interests of speed and size, as well as it being the language I am most familiar with thanks to my work placement, I expect to implement the program in C. Should the program stray into more substantial data analysis or even machine learning I may consider implementing aspects in Python.

I hope to learn a lot about maintaining good program structure and design, as well as the implementation of high performance algorithms.

References

Navarro, G. (2001). A Guided Tour to Approximate String Matching. *ACM Computing Surveys*, [online] Volume 33(1), pp. 31-32. Available at: https://doi.org/10.1145/375360.375365 [Accessed 9 Oct. 2020]