String Pattern Recognition

Project 3

Suffix Trees and Applications

Paco Gómez Martín

DEADLINE: December 20th 2009, 23:59 p.m.

1 Introduction

This project consists of the following goals:

- Reinforcement of the knowledge acquired by students about suffix trees and Ukkonen's algorithm.
- Deep understanding of the properties allowing the linear-time algorithm as well as the data structures and implementation details involved.
- Actual programming of Ukkonen's algorithm.
- Gathering raw data from the execution of the algorithm with the goal of obtaining a better understanding of the algorithm.
- Solve a few string matching problems with suffix trees.
- Conducting a computational experiment.
- Comparison of theoretical and experimental results.

- Interpretation of results and presentation of conclusions.
- Writing an academic paper.

2 Ukkonen's Algorithm

Ukkonen's algorithm must be fully implemented. That includes the following:

1. Use of implicit suffix trees.
2. Use of all speed-ups of the algorithm: suffix links, skip/count, edge compression, leaf rule and halt condition.

Once Ukkonen's algorithm is implemented it is proposed an experiment to check the linearity of the algorithm. The experiment have to be carried out according to the following directions.

1. Strings of size n has to be generated, where n varies from 100 to $5,000$ and increases by 100 each time (that is, $n = 100, 200, \dots, 5,000$).
2. Those strings have to be generated at random; its characters will be drawn from the alphabet $\Sigma = \{\texttt{0},\texttt{1}\}$.
3. For each string, Ukkonen's algorithm has to be run and running times will be annotated.
4. Running times will be plotted against input size (again, see Exercise 1).
5. A linear regression line will be fitted. This can be accomplished by using a mathematical software such as StatGraphics, Derive or Maple. Should students need help at this point, they should contact me.
6. From the fitting functions estimate the constants associated with the complexity.

The last part of the project is to use the generated suffix trees to solve the exact string matching problems. The test data will be the patterns and texts given below

Pattern	Text	Alphabet
Pattern1.txt	Text1.txt	Binary
Pattern3.txt	Text3.txt	Binary

All the files can be found at:
http://www.eui.upm.es/˜fmartin/webpgomez/Docencia/Pattern-Recognition-09-10/Project-3-Files/

Solve the SMC problem and after that answer the following questions:

1. Is pattern P in T?
2. How many occurrences of P in T are found?

3 Programming

Implementation of algorithms may be done in any language of student's choice. However, the language and its compiler should support certain features in order to be able to run the experiments properly. The choice of C, C++, Maple or the like should be enough. Source code and a .exe file have to be handed over.

4 Written Paper

A paper describing the following points must be handed over.

- Brief explanation of the algorithms.
- Brief explanation of the implementations. It can be done by including sufficiently detailed comments in the code.
- Brief description of the experiment.
- Interpretation of experimental data.
- Conclusions. Draw your own conclusions (be creative, but not extravagant or too showy).

The paper has to be written in correct Spanish or English; it also has to possess clarity of thought. Show me what you know; do not force to search for it through a poorly written paper.

5 Grading

The whole project counts 40% (2.5 points out of 10) of your final grade. The point left in the grade will be assigned to class participation, which has been so far very satisfactory. I will take points off when:

- There is spelling mistakes (either in Spanish or in English, but I will be tougher if they occur in Spanish).
- It is plenty of irrelevant material. Down with the irrelevant!
- It lacks clarity of thought.
- It is lengthy, long-winded or poor in content.
- Code is not properly commented.
- Code is not properly structured.
- Variables have absurd names.
- There are run-time errors.

6 Questions and Office Hours

I am willing to answer your questions about algorithms, complexity or the experiment. I will not answer questions about coding errors as it is my feeling that, at this point, writing error-free code is your responsibility.