KING SAUD UNIVERSITY
COLLEGE OF COMPUTER & INFORMATION SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Course: Algorithms Design and Analysis - CSC311
Semester: First Semester 2019/2020
Instructor: Dr. Mohamed Maher Ben Ismail

Theoretical and Empirical Analysis of Algorithms

DEADLINE: ***Dec 12th2019, 23:59***.

1. Introduction

Plagiarism is a serious problem in research. In this project, you will implement a very simple string matching algorithms. Your input will be a corpus of existing documents and a pattern you are looking for. Your output will be the set of documents that contain your query pattern.

2. Project Requirements

This project consists of comparing several basic strings matching both from a theoretical and experimental point of view. Goals of this project include:

- Acquisition of knowledge about string matching algorithms.
- Actual programming of these algorithms.
- Conducting a computational experiment.
- Comparison of theoretical and experimental results.
- Interpretation of results and presentation of conclusions.
- Writing an academic paper.

3. String matching algorithms

Implement the following basic detectors and compare their performance:

a) Brute force algorithm.
b) Knuth-Morris-Pratt’s algorithm (KMP).
b) Boyer-Moore's algorithm.
I will provide you with a tutorial on these algorithms. Also, you can read up in Wikipedia or any other source.

4. Experiments

Experiments have to be carried out according to the following directions.

1. Each algorithm has to be run on test documents. Data will be generated in a random way.

2. Each algorithm has to run on a text document of size **n**, where n varies from **100** to **10, 000** and increases by **100** each time (that is, **n** = **100, 200, . . . , 10, 000**).

3. If necessary, an algorithm may be run on test data a fixed number of times so that the running times obtained are meaningful

4. Running times will be plotted against input size.

5. From the plots, the input size for which an algorithm is faster than another will be estimated.

5. 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++, Java, Maple, Matlab or the like should be enough. Source code has to be handed over.

6. Project Demonstrations

Once the project is completed, the following is expected of you:

a. A demonstration video (5 to 10 minutes) of your project where you show the various features of your system, such as its correctness, efficiency, etc. You should provide all details on the system design and implementation during this demo. Your code will be examined to check for code quality, code documentation, etc.

b. You should also hand in a completed project report which contains details about your project, such as main data structures, main components of the algorithm, design of the user-interface for input/output, experimental results, e.g. charts of running time versus input size, etc.

c. You should also turn in your code and associated documentation (e.g. README files) so that everything can be backed up for future reference.

d. Email your code and all associated files to Dr. Maher with "CSC 311-Project<Lastname>" as subject.

7. 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. Comparison of experimental data with theoretical complexities.
- Conclusions. In this section students must answer the following questions:
    1. What algorithm is best and under what circumstances?
    2. Classify the algorithms in terms of input size.
    3. Draw your own conclusions (be creative).

The paper has to be written in correct 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.

8. Grading

The whole project counts 20% of your final grade. I will take points off when:

- There is spelling mistakes
- 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.

9. Questions and Office Hours

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