CS 677 High-performance Computing
Programming Assignment #1

Measuring Complexity

Overview

This is intended as a "Warm-up" exercise. The main purpose of this assignment is to demonstrate your problem-solving ability and basic coding skills, and to practice conducting experiments and communicating results.

Background

One way of characterizing data is to measure and compare its relative complexity. This requires a definition of complexity and a method of comparison.

Consider linguistic complexity. We could define the complexity of written text as the ratio of the number of unique words to the total number of words in a sample. Think of a book for young readers ("*See Spot. See Spot run.").* The intuition is that repeated words and a small vocabulary make for simpler comprehension.

This complexity metric is very easy to compute. The text sample "*Now is the time for all good men to come to the aid of their country."* is a sentence consisting of 16 words, of which 14 are unique. Hence its complexity is 0.875 as defined.

Different types of documents naturally have different complexities. Speeches and sermons are often less complex, both because of their oral delivery mode and their desire to establish a pleasing or memorable rhythm in the ear of the listener. Novels are often of low *total* complexity simply because they are lengthy (i.e. resulting in a large denominator)

Interestingly, it's not uncommon for complexity to change in a given document. For example, the introductory chapter of a textbook might use less discipline-specific words and hence be less complex than later chapters. Measuring *local* complexity requires the use of a *sliding window*, which is used to compute the changing complexity at varying points in a sample. Using the text sample from above:

Now is the time for all good men to come to the aid of their country.

Now is the time for all good men to come to the aid of their country.

In this case the size of the sliding window is defined as 5 words. At the location given in the first example, all five words are unique, resulting in a complexity of 1. The window is then slid forward one word. The complexity is now 0.8, as there are only four unique words within the window.

CS 677 High-performance Computing
Programming Assignment #1

Specifications

Your assignment is to write a program that determines the linguistic complexity, as defined above, of the posted documents.

To receive a grade of 'B' your program should:

- Read in the text sample
- Clean it
  - Convert all text to lowercase
  - Eliminate all non-alphabetic characters
- Tokenize the remaining text (break it into words)
- Determine the *total* linguistic complexity of the document

To receive a grade of 'A' your program should additionally:

- Measure the *local* complexity using a sliding window approach
  - Note: the average sentence consists of 15-20 words
- You should also graph the local complexity as computed by your program. This can be accomplished using a graphing utility/program of your choice.

Notes:

- This assignment is a good opportunity to learn/refresh your C/C++ skills.
- Be sure to demonstrate good programming style and practices.

Experiments

The Project Gutenberg website contains thousands of documents in text format.

- Is there a relationship between complexity and document length?
- Has linguistic complexity changed over time (for example, are today's books easier to read than those written in the 1700's)?
- How does a novel compare to a textbook? How does a Trump speech compare to an Obama speech?
- Other ideas of your choosing…

Deliverables

- Submit a **single PDF**, formatted as a Report. The report should describe your approach (data structures, algorithms), problems encountered, the experiments you conducted, and your results/analysis/conclusions. Include source-code, sample output, and complexity graphs.
- Be prepared to present and discuss your solution in class.