CS 677 High-performance Computing
Programming Assignment #1

Measuring Complexity

Overview

This is a simple "Warm-up" exercise. The main purpose of this assignment is to demonstrate your problem-solving ability and basic coding skills.

Specifications

One way of characterizing data is to measure and compare its relative complexity. This assumes that the definition of complexity is given and that the method of comparison is specified.

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. For example, 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.

It is also common for complexity to change locally. For example, the introductory chapter of a textbook might use less discipline-specific, unique words and hence be less complex than later chapters. Measuring changing complexity requires the use of a *sliding window*, which is used to determine the local complexity at varying parts of a sample.

Your assignment is to write a program that determines the linguistic complexity, as defined above, for the text samples posted on the course info page.

Your program should:

- Read in the text sample
- Clean the sample
  - Convert all text to lowercase
  - Eliminate all non-alphabetic characters
- Determine the total complexity of the document
- Measure its local complexity using a sliding window approach (of user-defined size)
- Graph the local complexity

Notes:

- You may use the programming language/platform of your choice.
- Try to optimize your solution for running time and/or memory usage.
- Be sure to demonstrate good programming style and practices.

CS 677 High-performance Computing
Programming Assignment #1

Enhancements

Note that other forms of data could be considered a sample and examined.

- Numeric data: analyze the digits of *pi* (where each digit is considered a word)
- Genomic data: analyze a sequence of genomic data (where each triplet, or codon, is considered a word)
- Other ideas of your choosing…

Deliverables

- Submit a hard-copy of your source-code, sample output, complexity graph, and a design document.
- Be prepared to present and discuss your solution in class. What data structures did you employ? What algorithm(s) did you use? What interesting problems (and solutions) did you discover?