Project Rubric and Sample Proposals

Consult the Schedule for proposal and project submission dates.

Here is the basic marking scheme.

20% Choice of topic.
Evaluated on relevance to CS350 topics (but an original or interesting topic or approach is possible),

20% Description of testing procedures. This is not a programming assignment so the experiment or trials, etc. should be described here. In scientific journals this is the Methods Section.

30% Execution and discussion, including presentation modes and analysis of results.

15% Write-up. This should clear and should

indicate where approriate limitations of the Method or unexpected results. Length is not in itself a good thing. A concise, carefully presented report need not be more than 5 or 6 pages.

15% References to literature, including any open source software used and background on the algorithms being tested or compared. At least 4 *relevant* citations with full bibliographic information.

A typical project might be a comparison of two string matching algorithms from the standpoint of ease of implementation, practical execution behavior (you decide on the range of input sizes, etc.) and potential bottlenecks. For example, algorithm A may perform better for large alphabets while algorithm B is faster for the binary alphabet.

It is expected that the experimental information will invlve executing programs. The

interest is in what computing resources are monitored. Clock time is not by itself a good measure (see testing procedures). You are not required to code from scratch. If you wish, you may submit source but only as reference material. It will not be marked in terms of programming techniques.

Sample Proposals

1. Compare running times for Dijkstra's Algorithm using two
   different implementations of a Min-Queue.
   The sample space of digraphs
   will be carefully defined and results will
   be analyzed in terms of graph type
   and Min-Queue implementation.
2. Compare running times for two string matching algorithms (e.g.
   Boyer-Moore and Knuth-Morris-Pratt).
   Detailed study of the influence of

alphabet size will be part of the analysis.

3. Compare running times of Topological Sort via DFS and an intelligent implementation of iterated removal of in-degree 0 vertices (as in week 4 notes).