CSC721 – Laboratory #1*

David John

Fall 2008

1 Purpose ...

Theoretical and empirical analyses of algorithms are powerful tools for measuring the complexity of algorithms. The ability to predict performance as a function of the *size* of the input has proved to be most valuable. In cases where reasonable theoretical predictions are not available, runtime measurements are most helpful. Even when theoretical measures are at hand, run time experimental measures help in understanding the nature of the constants and small size performance associated with the algorithms.

The theoretical measures of Ω, Θ and O are usually presented for an algorithm. However, the derivation of these can be often quite tricky. Empirical measurements to approxiamate, or complement, these values often are taken from well designed test sets. Designing a test set is itself quite a challenging issue. Once test sets are designed, and the algorithm is tested using them, then meaningful statistics must be generated and presented.

In this lab, and all subsequent labs, there will some issue related to complexity to explore. In addition to implementing some algorithm(s), it will be necessary to discuss the algorithm(s). Included in this discussion must be appropriate theoretical complexity presentations and pseudo-code descriptions. Discussion of the test set generation must be included. A digest of empirical measurements, usually including graphs, must be included. Finally, a summary of the lab and its results appear. All labs must be written in the form

*Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and I am aware of a trademark claim, the designations have been printed in initial caps, all caps or italics.

of an article that would be submitted to a journal. Part of journal article submission is peer review, you may ask (or be asked) to review someone's assignment, this is permissible.

You must use a word processing system and other appropriate tools in the construction of your article. I will be glad to help you in this matter. There are several systems available to you.

2 Sorting with repeated items ...

Often a common assumption in the presentation of sorting algorithms is that the list of items to be sorted contains no repeated item. The reason for this simplifying assumption is either to make the formal analysis a bit more straightforward or to construct a very clean algorithm. However, repeated items in a list are certainly not unusual and are often expected. The whole notion of stable sorting algorithms is concerned with complexity of algorithms when items repeat.

When there is an integer list with repeated elements there are two important parameters which will contribute to the complexity of the algorithm: the size of the list, n, and a distribution function, $f : Z \rightarrow R$, that indicates the probability of occurrence of each integer in the list. As f varies, there are different behaviors of a particular algorithm.

Bingo Sort [1, Section 4.6] is an algorithm designed to address repeated elements in a list. Please read over the discussion of *Bingo Sort*. For our purposes assume that the list to be sorted consists of integers.

Another similar comparison based sort is *Selection Sort*. This basic sort does not take into account repeated elements. *Bingo Sort* is an improvement of the *Selection Sort*.

Your task is to compare *Bingo Sort* and *Selction Sort*, and to conclude which is *better* than the other with a thorough justification.

3 Article outline ...

Below I suggest a format for your article. You certainly can use any other format you choose, but note that the particular issues addressed by the given format must be included in your article. It is absolutely necessary that articles be written for others to read. Hence, correct grammar as well as clear exposition is necessary (the reviewers will help with this).

As much as I like electronic versions of papers, it is necessary to initially turn in a printed copies of your article. When the final version is submitted for a grade, it must also be submitted on paper.

3.1 Introduction

Both *Bingo Sort* and *Selection Sort* must be presented at the intuitive level. Pseudo-code expression of the algorithms must be given. Perhaps a simple example can be given to compare and contrast the two algorithms. You are to briefly outline what will be shown in the next sections of your article.

3.2 Theoretical Complexities

Theoretical complexities for both *Bingo Sort* and *Selection Sort* must be presented as functions of n and f. The derivations must be included.

3.3 Experimental Strategy

Present the strategy of your experiments. Include discussion on the sizes, design, generation and number of test sets. Explain the statistics to be gathered.

3.4 Experimental Results

Display the results of the experiments. Graphical display of the results is most appropriate. Discussion is mandatory.

3.5 Summary

What has been shown by the experiment? What has not be shown by the experiment? What are some other questions that could be pursued.

3.6 Appendix A-Code for *Bingo Sort*

The implementation of algorithms must appear. Your code must be well written and relatively easy to read. Appropriate comments and spacing styles must be present. It is certainly appropriate to have the algorithm implemented by using several external routines that are compiled and linked under the control of a *makefile*, or similar tool.

3.7 Appendix B-Code for ***Selection Sort***

The implementation of algorithms must appear. Your code must be well written and relatively easy to read. Appropriate comments and spacing styles must be present. It is certainly appropriate to have the algorithm implemented by using several external routines that are compiled and linked under the control of a *makefile*, or similar tool.

3.8 Appendix C-Misc

Include in this appendix, or extra appendices, any other item you wish. For example, if you have written a test file generator, include it here.

3.9 References

A bibliography must be given for all works used. This includes papers, books and on-line references. We will talk about an appropriate reference style in class.

4 Time frame . . .

The lab is due to me no later than 5pm on Friday, September 19.

Implementing the algorithms is not the hard part of the assignment. Measuring the performance of the algorithms will take some time. Constructing the article will also. Please do not wait to the last minute to work on this.

5 Who can help whom . . .

All work turned in for a grade must be your own work. This does not mean you can not talk with each other about ideas and problems. In fact, the referee process guarantees you will have another member help you significantly. You may not work together on the lab, you must implement your own version of the two sorting algorithms, you must generate your own test sets of data and you must write up your own article. Certainly if you have a problem deciphering a compiler error message, talk to a fellow student.

I am always available for consultation. Please feel free to contact me directly or via electronic mail.

References

[1] Kenneth A. Berman and Jerome L. Paul. *Fundamentals of Sequential and Parallel Algorithms*. PWS Publishing Company, 1997.