**Project**

CMSC-441 Algorithms, Alan T. Sherman, CSEE, UMBC Fall 2019

# Overview

Each student will deeply explore two closely related algorithms, implementing and empirically comparing their actual time and space usage. The project should be focused around an interesting, well-motivated question, whose answer is initially unknown. For example, what is the smallest input size for which Strassen’s algorithm runs faster than the basic matrix multiplication algorithm? The algorithms and question must be approved by the instructor. Present your findings in a written report (about 10 pages long) in the format of a computer science technical report and in a 15 min oral presentation with Powerpoint (or similar) slides. This project will count for 25% of the semester grade.

**Two Choices**

Each student may choose one of the following two options for selecting an algorithm to study. Students are encouraged to select Option 1, but your choice of option will not affect your grade. Many students learn more if they follow their own curiosity, take a risk, and create their own project. On the other hand, the standard project is a carefully constructed interesting task with proven learning value and appropriate difficulty.

1. *Custom Project.* Compare any two algorithms for solving the same problem of your choice. Often the best projects flow from life experiences (*e.g.,* Did you ever encounter an interesting and important algorithms problem in a job?) Your proposal must be approved by the instructor. Standard projects from the instructor from recent previous terms are not allowed. Each custom project must begin with a published research “reference paper” or two, which provide relevant background to start your project.
2. *Standard Project from Instructor.* See separate handout. This term, the standard project is to implement two algorithms for matrix multiplication (Strassen and basic matrix multiplication), and to compare their performance.

Regardless, in addition to other elements described in this project specification, each project should briefly provide some history of the algorithm, theoretical analysis, what is known about the optimality of the algorithm, current research questions pertaining to the algorithm, and discussion of how well the experimental results match the theory. Comparing two closely related algorithms is helpful in calibrating and interpreting experimental findings.

# Purpose

The purpose of this project is to gain a deep understanding of a focused topic and to gain hands-on experience with the experimental evaluation of algorithms. In addition, the project provides an opportunity for each student to demonstrate and integrate all skills learned in this course, including communicating algorithms findings in a technical report and oral presentation.

**Deliverables and their Due Dates**

Proposal (custom or standard) - Sunday February 20

Reference paper (for custom projects only) - Sunday February 20

Progress report - Sunday March 13

Complete draft report and draft Powerpoint presentation - Sunday April 24

Final report and Powerpoint presentation

(with source code as an appendix) - Sunday May 15

Confidential review of group members (for groups

with more than one member) - Sunday May 15

*Each deliverable will be submitted via BlackBoard. See syllabus for lateness policy.*

# Groups

Students may work in groups of size at most three (size three is recommended). Students are encouraged, but not required, to work in a group. Typically, each student in the group will receive the same grade. Group work can promote learning. Each custom project group should explore a different pair of algorithms. Potential groups of size greater than three should reorganize into two or more groups.

For groups with more than one student, each group member must submit a confidential evaluation of the group members explaining what each person did (including yourself) and how well they accomplished their task.

**Grading Policy**

# The project will count for 25% of the semester grade. The project grade will be based on the final report (55%), presentation (20%), source code (10%), proposal (5%), progress report (5%), and draft (5%). The report will be evaluated on the basis of

# appropriateness to the assignment,

# scientific merit (correctness, significance, originality, non-triviality, scientific completeness), and

# effective presentation (clarity, organization, document completeness, spelling, English usage).

# Carefully read and follow Sherman's advice on writing technical reports.

**Machines, Languages, and Measuring Time and Space**

You may use any computer language and machine of your choice. You must determine for your machine the best way to measure time and memory. All operating systems have system calls that return the current time and number of bytes of memory used by the calling process. By taking the difference between two such calls, you can compute the time and space used between calls. Be sure to identify the most appropriate such system call—one that is least likely to be affected by confounding factors, such as time and space consumed by other users and other processes. As a default choice, you may wish to use the C or C++ language on one of the UMBC mainframes running Unix. In the discussion section of your report, be sure to identify any possible sources of error that may affect your time and space measurements.

Stress test your implementation by running inputs of various sizes, including some very large inputs (e.g., that take several hours or more to run). What is the largest input that you can handle?

**Proposal**

The proposal should explain what you want to do, why, and how. Typically, the body of a proposal includes sections for motivation, specific aims, plan of attack. You must submit a proposal even if you are doing the standard project. It must contain the following elements:

1. project title,
2. executive summary (at most one page),
3. group members and their roles,
4. which pair of algorithms you will study,
5. clear statement and motivation of a main focused question your report will answer,
6. list of references,
7. list of difficulties expected and how you plan to overcome them, and
8. a schedule for carrying out your work

**Progress Report**

The progress report is a preliminary draft of the final report. It must include the following elements:

1. detailed outline of final report,
2. clear statement and motivation of a main focused question your report will answer,
3. explanation of your choices for computer language, machine, and how to measure running time,
4. one graph showing the actual running times for your two algorithms, and one graph showing actual space usage for your two algorithms,
5. list of references,
6. list of difficulties encountered and how you plan to overcome them, and
7. a schedule for completing your work.

**Draft Report and Presentation**

A complete draft of the entire report. The outline should be stable. The main ideas and results should be present. A complete set of draft Powerpoint slides for your 15mins. presentation (about 10 slides).

**Technical Report**

Each group will present its findings in a computer science technical report. Focus sharply on what is new and significant about your work (your main question and its answer). Be sure that your document is complete, as described in my essay, ``Some advice on writing a technical report,’’ http://www.cs.umbc.edu/~sherman/Courses/documents/TR\_how\_to.html

In particular, please note that the front matter includes title, author names and affiliations, date, abstract, and keywords. The back matter includes acknowledgments and references. There is no length requirement (typically, I would expect about 10 pages for the report).

The organization of the body of the report should be some variation of the standard outline for all scientific work: motivation, previous work, methods (including how you measured time and space), results, discussion, open problems and conclusions. The report should include a discussion of the history of prior work on the algorithms and open problems for the future. Although the exact organization is up to you, you must clearly explain and separate motivation, methods, results, and discussion.

Include printouts of any source code as a separate appendix. Be sure to document your source code clearly.

Please hand in these items on 8 ½ x 11 inch white paper, one-sided, with no staples. Put one large binder clip around package. Write your name(s) on each item. Do not use folders or covers. Keep a copy of everything you hand in for your records, in case something gets lost.

Prepare your report using Latex on Overleaf so that you are able to write mathematical prose properly and edit your drafts easily. Do not use any other document preparation system. Latex is the only available system for which it is possible to produce beautiful mathematical prose—see Wiki.

Your report should look like a computer science research paper (for example, see any IEEE Transactions). Do not include a separate title page.

**Graphs**

Two crucial elements of your report are the following two graphs: a graph showing the actual running times for your algorithms for a variety of input sizes, and a graph showing the actual space usage for your algorithms for a variety of input sizes. Each graph should show the results for both algorithms on the same set of axes.

Be sure to label the axes of the graph appropriately, clearly showing the type (*e.g.,* time), units (*e.g.,* seconds), and scale (*e.g*, each notch on axis represents 10-6 seconds). To fit all data on a single graph, you may find it convenient to use a semi-logarithmic scale (*i.e.,* plot lg *T*(*n*) as a function of *n*, where *T*(*n*) is the observed running time.)

Because the observed running time (similarly, space) can vary for different inputs of the same size, be sure to compute the average running time as observed from several (say, 5 or 10) randomly-chosen inputs of each size*. Also compute the observed standard deviation of these sampled times* as a measure of the variation in the actual running times for each input size. Choose a variety of input sizes, including some very large sizes. For example, you might try input sizes of approximately 2*k* for various *k*. For each input size you try, plot on each graph the corresponding average and standard deviations for the observed running times. For example, the spreadsheet Excel can easily plot such data using “error bars” to show the observed standard deviations. There are variety oftools for graphing, including Mathematica, Matlab, and Excel.

**Regression Analysis**

You must apply *regression analysis* to fit your data to the theoretical running time of each algorithm. For example, if the running time of an algorithm is *c*3*n*3 + *c*2*n*2 + *c*1*n*1 + *c*0, then you could find the real constants *c*0, *c*1, *c*2, *c*3 that best fit your experimental data. There are standard tools for doing so, and many spread sheets have this functionality.

Be sure to provide regression analysis that fits the theoretical running to the observed data. Do not fit some other arbitrary function (e.g., *en*) to the data, as many spread sheets will do unless otherwise instructed.

**Presentation**

Prepare a 15 minute presentation (including questions) supported by slides. Emphasize your new and significant findings. Present a small number of interesting relevant selected technical details. Include graphs, properly labelled. Minimize text (write in short bulleted lists, not complete English sentences). Use color, images, and figures. Dress business casual with no coats or hats. Look at audience. Find ways to engage the audience. Begin with your main results (the talk should not be a murder mystery).

**Hints**

1. Start early and do not wait to the last minute.
2. In the introduction, clearly explain what problem you are solving and why. Why is experimental evaluation needed or useful?
3. In the introduction, clearly explain what is new and significant about your work.
4. Do not simply list previous work, but comment critically upon it, explaining how your work improves and extends upon this previous work. (This project does not expect you to do much with previous work.)
5. Be sure to label each graph axis with its type, unit, and scale.
6. Be sure to compute both the average and standard deviation of all observed running times and space usage.
7. Be sure your technical report has all required parts, including abstract, keywords, acknowledgments, and references.
8. Choose an informative title. Do not title your report “CMSC-441 Project.”

(9) Be sure to write your abstract in an informative style. The abstract should explicitly state what you accomplished and should serve as a substitute for the entire paper; it should not be confused with the introduction. Concretely state your findings. Get to the main point in the first sentence of the abstract. Do not write the abstract in a descriptive style, which vaguely hints at what you did.

(10)Be sure your report includes all required components (time and space graphs, documented source code as appendix) and that it answers all required questions.

(11) Throughout, sharply focus on what is new and significant about your work (the experimental evaluation).

(12) Write in the active voice. Avoid the passive voice.

(13) Do not simply list experimental results; discuss and interpret them insightfully.

(14) Carefully read and follow the project checklist.

(15) Be sure that you include author names, affiliations, date, keywords, and acknowledgments (many students erroneously omit one or more of these items).

(16) Carefully read and follow these hints, including Hint 12.