Benjamin Swann
Jesus Rivera III
Tomas Vasquez

Project Proposal

1.1 Problem Description

We will be conducting performance testing on AMD, and Raspberry Pi's ARM processors. We think that performance testing graph algorithms on these processors would be fundamental because of the shift to smaller portable devices are on the rise. In the last year Linux has announced that they will be using Arch Linux as a portable gaming system named Steam Deck. It is an important topic because graph algorithms improve performance from Amazon driver's deliveries to a simple social media platform. Graph algorithms improve either google searches to connected library databases or flight patterns for pilots.

We will ~~also~~ test the performance of the same program but using different versions of the compiler at different levels of optimization (-O1,-O3) to examine what effect using an updated compiler vs a very outdated one would have on the program. We believe that there will be a non-insignificant gain in performance which will help show the need to keep tools updated.

1.2 Methodology

We plan to use a program that gets passed an adjacency matrix. It is either an undirected, weighted, or a directed unweighted graph. It performs a Breadth First Traversal and Depth First Traversals. It adds vertices and deletes them. It adds an edge between two distinct points. If it already exists, it doesn't place one. If an edge gets removed, it will disconnect the graph. We have accounted for that in the program by counting and listing them at end of program.

We plan to use several performance metrics tools from class in addition to others. Some of the planned tools are the "Perf" and other Linux built-in measurement tools.

We will need to source various machines to test on as well as older version of the g++ suite to test older compiler versions. Some planned machines to use include a home built x86_64 AMD based desktop, a Raspberry Pi 4 with 8GB of RAM, a Raspberry Pi 3 with 1GB RAM, and possibly even more hardware if the opportunity presents itself.

1.3 Deliverable

- The program code used for benchmarking with documentation (What the program does)

- In class presentation/discussion on our findings and utilization of the program to preform our test.

- Formal written report of entire project (Purpose, Procedures, results, Analysis)

1.4 Team

Benjamin Swann- Test program on Pi 3 with 1GB ram and other hardware if necessary, and formal analysis of results. Report proofreader and secondary report writer.

Jesus Rivera- Test program on Pi 4 with 8GB ram, and on x86_64 AMD desktop, primary report writing.

Tomas Vasquez- Development/acquisition/documentation of code used for benchmarking. Acquiring relevant benchmarking tools. Team lead.