Title: Profiling the Serial Implementation of Dijkstra's Algorithm Using GPROF

Abstract:

This document outlines the utilization of the GPROF profiler to assess the efficiency of a serial program. The study focuses on the implementation of Dijkstra's algorithm for identifying the shortest path in a graph. By employing GPROF, the execution of the algorithm is profiled and analyzed to gain insights into its performance.

Methodology:

The methodology employed for this endeavor encompasses the following systematic steps:

Serial Algorithm Implementation:
Dijkstra's algorithm, a renowned technique for finding the shortest path, is realized in a serial manner using C++. The implementation involves various functions, notably "min_dist" and "dijkst."

Command Used: g++ -pg -o dijkstra_serial dijkstra_serial.cpp
Purpose: Compiles the dijkstra_serial.cpp program with the -pg flag to enable profiling.

Command Used: ./dijkstra_serial
Purpose: Runs the compiled dijkstra_serial executable to execute the Dijkstra's algorithm program.

Command Used: gprof dijkstra_serial gmon.out > profile_report.txt
Purpose: Generates a profiling report using the GPROF profiler, based on the information in the gmon.out file.

Results:

The profiling report provides a breakdown of execution time distribution across different functions within the serial program. Key findings from the profiling report are summarized below:

The "min_dist" function contributes 100% of the overall execution time.
The "dijks" function accounts for 100% of the total execution time.

Collective execution of other functions makes up 100% of the overall execution time.

The report presents details such as call counts, execution times, and the percentage of total time spent within each function.

Analysis:

The profiling outcomes unveil crucial aspects of the serial program's performance:

A significant portion of execution time is consumed by the "min_dist" function. This suggests that optimizing this function could yield performance enhancements.
The "dijkstra" function also has a notable impact on execution time, indicating potential optimization opportunities within this function.
Execution time is influenced by the graph's size and complexity. As the number of vertices and edges grows, the program's execution time increases proportionally.

Conclusion:

In conclusion, the employment of the GPROF profiler offers valuable insights into the performance attributes of the serial implementation of Dijkstra's algorithm. The profiling report identifies the functions that consume the most execution time and pinpoints potential optimization avenues. Profiling tools like GPROF play a pivotal role in recognizing bottlenecks and guiding optimization endeavors in software development. This undertaking has provided firsthand experience in profiling and scrutinizing the performance of a serial program.