Cristian Poll
CSC398 A3: Assignment Proposal

Problem:

Efficiently utilizing several computers that communicate through TCP/IP sockets in order to find the largest common substring of two strings.

Target:

We intend to program our solution in C and use the 40 computers (cs01 to cs40) in the SB1158 as our working machines.

Description:

We would like our program to effectively utilize the processing power of several computers that have similar specs. We have chosen the SB1158 machines because they have constant uptime but often see little use of their processing power apart from peak hours (during the day, especially during class).

We have selected the LCS problem as our problem to solve for several reasons:

- We have already demonstrated in A2 that it benefits from parallelization, and have single-system multi-core benchmarks with which to compare our solution to.
- LCS has real-life applications, such as in biology, with very large input where performance is a major concern.
- Other dynamic programming problems can be solved with a similar/same methodology as the one we will use for solving LCS. Thus, we can apply our LCS solution to any dynamic programming problem we may encounter in the future, or perhaps adapt our solution to a more general-purpose dynamic program solving framework at a later date.

Goal:

Our goal is to create a program with performance that reasonably scales with the amount of resources (equally performant computers) we provide it.

Apart from correctness, we will measure our success based on a variety of sub-goals, most or all of which we hope to achieve:

- The program scales with amount of resources allocated; that is, adding equally performant computers to help calculate the result should decrease the overall runtime. Preferably, runtime should scale linearly with number of computers working on the task.

- The program has comparable performance to our A2 program, given comparable resources (number of cores, core speed, architecture). It will be difficult (likely impossible) to match the A2 program, but the difference should not be enormous.
- The program has some provisions to handle non-equal resources; that is, a computer that is less performant than the other ones running the program should not significantly bottleneck the program. At the very least, adding a computer should never decrease performance.

Further goals are collecting statistics on performance and comparing performance to other implementations of LCS solvers (including the one implemented in A2). We hope to analyze our results and deliver conclusions on the feasibility, scalability and adaptability of the projects.

Constraints and Assumptions:

Distributed computing in general has already been explored by projects such as Berkeley Open Infrastructure for Network Computing (BOINC), which utilize volunteer computing. Our solution, however, is designed for a closed system, and we can thus make more assumptions for the sake of reducing overhead and improving performance:

- We assume that each machine running the program is dependable. That is, it will not return incorrect values, either due to hardware error or malicious modification.
- We assume that each machine running the program is approximately equivalent. No machine should bottleneck machines dependent to it, and minimal order-changing should be required.
- As an extension, we assume that the machines have approximately equal performance at all points of runtime. That is, performance should not spike, and there should be little or no competition for processor resources. In the case of the SB1158 machines, we assume that we have nearly exclusive use of one of their two processors, and no other users are logged on (no unexpected performance drops).

Summary

Our goal is to create a program that works for a specific set of assumptions on computers that otherwise idle wastefully, specifically the computers in SB1158. We know the machines will preserve integrity, we can make several assumptions we could not make with a voluntary computing scheme like BOINC. We also intend to take benchmarks and improve the program in order to meet our goals pertaining to efficiency.