CITS3402 High Performance Computing

Mitchell Pomery
21130887

October 1, 2014

Introduction

To gain the ability to impliment parallel processing in the code the variable scopes needed to be fixed. The code was originally C89 standard, which relies heavily on global variables. This mean that every loop was using the same variable.

Development

The code was placed into a github repository to make it easy to track changes and backtrack if issues arrive. This came in handy when I accidentally changed a double to an integer and was suddenly seeing two files coming out diferently to before. To check that the code was creating the right output after each modification, a makefile was used. It compiles the code, runs breather and then runs `diff {file} orig/{file} | wc -l` so that is was easy to which files, if any, had differing output. Jenkins CI (Continuous Integration) was going to be used to automate the testing, however I was unable to get my Jenkins server to SSH to the High Performance Server, despite both machines being located at UWA.

Performance

Performance was tested by running the code on `hpc.csse.uwa.edu.au`. Unfortunately, no system had been implimented to prevent multiple people from attempting to test at the same time, so testing had to be done while other people were unlikely to be trying. There was also nothing to prevent people from testing or running other code on the machine, and so it was not unusual to see running python or java processes chewing up large amounts of CPU time and physical memory.

The original program was run to get a base timing, however due to how quickly it ran, it was modified so that the chain length was 200. This meant that the code ran slow enough to see if any improvements had been made, but not so slow as to hog resources.