Universiteit van Amsterdam

Assignment 1

POSIX Threads

November 5, 2014

Student:
Robin Klusman
10675671

Maico Timmerman
10542590

Supervisor:
Dr. A. Pimentel

Course:
Concurrency and Parallel Programming

Course code:
5062COPP6Y

Contents

1 Introduction **2**

2 Method **2**
2.1 Wave Equation Simulation 2

3 Results **2**

1 Introduction

For this assignment a parallel programming solution needs to be implemented for two problems, a wave equation simulation and the Sieve of Eratosthenes. For the wave simulation the user can specify the amount of wave amplitude points, the amount of steps it needs to simulate and the desired amount of threads. The program then calculates all the wave values until it has done the specified amount of steps.

The Sieve of Eratosthenes ¡...¿

2 Method

2.1 Wave Equation Simulation

First the specified amount of threads need to be created, these threads will then all start executing the function *calc_wave*. *calc_wave* first checks if there is an amplitude point in the row $t + 1$ that needs calculation. This check is done in ascending order, using a variable *current_index* that keeps track of which amplitude point was the last one being calculated. *current_index* is mutex locked to check and increment it before starting calculation on that particular amplitude point, so that no two threads waste their time calculating the same point. Once the *current_index* reaches the last point in the wave, i_max, threads will wait until all other threads finish their calculations. When the row is completely finished the *current_index* is reset and the rows are rotated, after which an event is generated telling all threads to restart their routine. As an extra feature we added the option to also print the state of the wave for each row, thus animating the flow of the wave.

2.2 Sieve of Eratosthenes

3 Results

4 Discussion