Serial Code Optimisation

David Sharp:ds16797 Candidate 36688

October 2018

1 Introduction

In summary, I managed to partially optimise the 5-point stencil but not to the benchmarks listed in the assignment. In this report I will detail the optimisation techniques that worked for me and also outline the optimisations I hoped to utilise but couldn't fully implement.

2 Optimisations

2.1 Initial State

Compiler settings	1024 image	4096 image	8000 image
No flags gcc4.8.4	8.23s	318s	735s
No flags gcc7.1.0	8.20s	433s	577s
-O3 gcc7.1.0	6.81	121s	362s
-Ofast gcc7.1.0	2.34s	108s	125s
No flags icc16	3.40s	49.3s	176s
-xHost icc16	3.42s	49.3s	175s
-Ofast icc16	2.13s	93.6s	105s

These initial optimisations already show some interesting results, with gcc7.1.0 not being consistently faster across different scales and the Intel compiler icc version 16 being significantly faster than gcc7.1.0. -O3 and -Ofast each gave significant speedups on gcc, while -xHost - designed to adapt the code for the specific Intel processor - offered no speedup over the icc defaults.

2.2 Profiling

After getting the initial simple speedup from compiler flags, I ran gprof to get an intuition for where the majority of the runtime is located. The results were as following on the 1024 image.

Percentage Time	Self Seconds	Calls	Name
99.84	8.19	200	stencil
0.37	0.03	1	init$_{i}mage$
0.12	0.01	1	output$_{i}mage$
0.00	0.00	2	wtime

From the gprof profiling, it's very clear that the stencil function is the limiter on the code speed, since it is run 200 times. As such it makes the most sense to focus optimisation on this one function.

2.3 Applied Optimisation

My first thought for optimising the code was to reduce the math operation for each part of the stencil, changing the value change by multiplication then division by simplifying into a single multiplication. While this change may be already carried out by the more rigorous compiler flags, it makes the code more readable and from my own research multiplication is at least as fast as division in many languages. Following this

Following this change I