Parallel Programming Exercise 9 – 10

Author:	許秉倫 (b05901011@ntu.edu.tw)
Student ID	B05901011
Department	電機系

(If you and your team member contribute equally, you can use (co-first author), after each name.)

1 Problem and Proposed Approach

A **perfect number** is a positive integer whose value is equal to the sum of all its positive factors, excluding itself. The first two perfect numbers are 6 and 28:

$$6 = 1 + 2 + 3$$

$$28 = 1 + 2 + 4 + 7 + 14$$

The Greek mathematician Euclid (c. 300 BCE) showed that if $2^n - 1$ is prime, then $(2^n - 1)2^{n-1}$ is a perfect number. For example, $2^2 - 1 = 3$ is prime, so $(2^2 - 1)2^1 = 6$ is a perfect number. Write a parallel program to find the first eight perfect numbers.

I use the "interleave" method. For the ith processor, it is responsible for n == i, i+p, i+2p…

The final result is

6

28

496

8128

33550336

8589869056

137438691328

2305843008139952128

(Brief your problem, and give your idea or concept of how you design your program.)

2 Theoretical Analysis Model

There are totally n numbers. For each number, we should compute $\sqrt{2^n - 1}$ times to verify whether it is a prime. After applying parallel technique, the complexity becomes

$$O(\frac{n}{p} * \sqrt{2^n})$$

I print the perfect number in each processor respectively, so there are no communication between processor. Therefore, we can't not compute iso-efficiency metric.

(Try to give the time complexity of the algorithm, and analyze your program with iso-efficiency metrics)

3 Performance Benchmark

(Give your idea or concept of how you design your program.)

	A	B	C	D	E	F	G	H	I
1	Processors	1	2	3	4	5	6	7	8
2	Real execution time	0.001095	0.001047	0.000805	0.000802	0.000976	0.000844	0.000789	0.000832
3	Estimate execution time	0.001095	0.000548	0.000365	0.000274	0.000219	0.000183	0.000156	0.000137
4	Speedup	X	1.046157	1.361506	1.365614	1.122394	1.298513	1.388868	1.316666
5	Karp-flatt metrics	X	0.911758	0.601721	0.643028	0.863691	0.724134	0.673346	0.725136

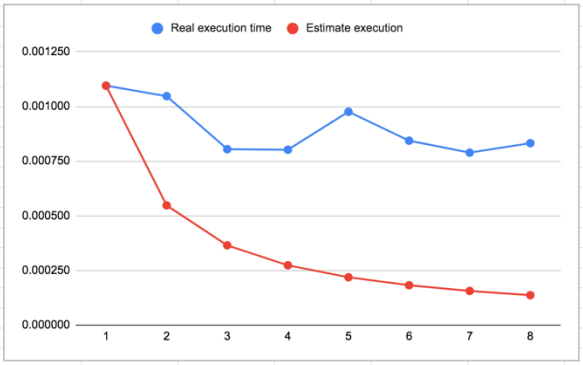

Figure 1. The performance of diagram

4 Conclusion and Discussion

1. Speed-up 會隨著 processor 浮動，可能是因為有隱藏的 overhead
2. 把判斷是否為質數的那個函數也平行化
3. Real 跟 Estimated 差距極大，可推測出 overhead 對程式影響甚劇
4. 由 K 值可推測出 overhead 佔了相當大的比例

(Discuss the following issues of your program

1. What is the speedup respect to the number of processors used?
2. How can you improve your program further more
3. How does the communication and cache affect the performance of your program?
4. How does the Karp-Flatt metrics and Iso-efficiency metrics reveal?

)

Appendix(optional):

(If something else you want to append in this file, like picture of life game)