HOMEWORK #5 CS 2045

QIMING GUO

Question 1

a. What is the loopcarried data dependence?
Every step i depends on step $i-1$.

b. Can you see a way to eliminate this dependence and parallelize the loop?
I think there're two ways to eliminate the dependence. The first one is pretty straightforward. If we could find the general formula for the n^{th} term. we could obtain the result of step i directly. The second one is scan parallel pattern.
(a): Actually, the result of step i could illustrated by triangular number:

$$a_n = 1 + 2 + 3 + ... + n = \frac{n(n+1)}{2}$$

With this new equation, the new loop is:

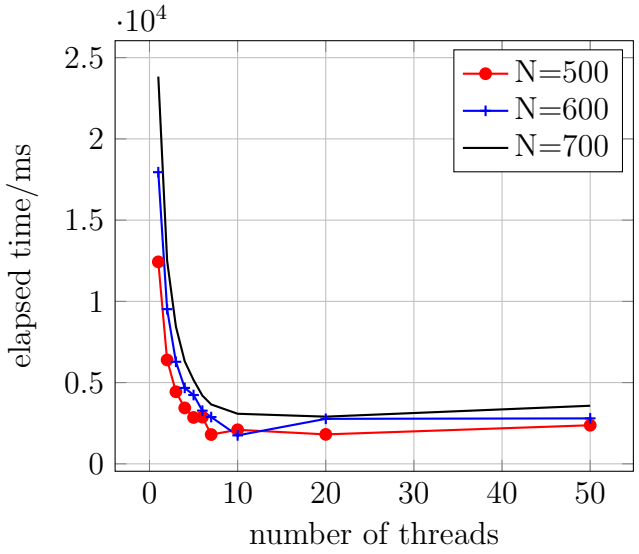
```
a[0] = 0;
for (int i = 1; i < n; ++i)
  a[i] = i * (i + 1) / 2;
```

(b). We could use scan pattern in slides to mitigate the data dependency. For example, we could add up [a[i],... ,a[i+N] and [a[i+N+1],... ,a[i+2N]] at the same time.

Question 2

The sequential version and parallel version has been completed and sent to TA.
Considering overhead of creation and destruction of threads, we should use a proper world size N to make the best use of parallel computing. After several attempts, I set N equals to 500, 600, 700 and M equals to 2000.

Date: October 11, 2015.



As we can see, as long as the number of threads increases, the running time decreases significantly. However, the relationship between x and y looks like reciprocal rather than linear. Maybe it is because the following reasons:

a). size of the problem is not big enough. In other words, compared with number of threads, size of the problem should be ten times or hundreds times.

b). Because of Amdahls's law. Not all parts of a serial program is paralleled. We could parallelize some parts, but cannot do the same thing with other parts.

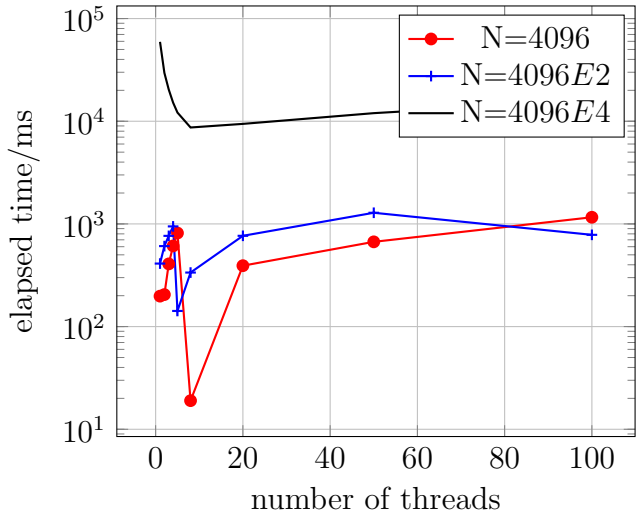
c). Overhead of creation and destruction of threads. As long as the number of threads increases, creation and destruction consume more and more system resource, even cancel out benefits of parallel computing.

The parallel pattern of Conway?s Game of Life is Stencil. Each node in the world is applied a function to neighborhoods, specifically, nearest eight nodes.

Question 3

The sequential version and parallel version has been completed and sent to server.

like Question 2, we should choose different M and observe the performance of parallel computing. Specifically, I use M = 4096, 409600, 40960000:



When N = 4096 or N = 4096E2, the performance seems doesn't be improved. I guess maybe it is because the overhead of parallel computing:

running time of serial version:

$$T(M) = (\log_2\lceil M^2\rceil + 1)M^2 \log M$$

running time of parallel version:

$$T(M) = (\log_2\lceil M^2\rceil + 1)(\frac{M^2}{N}\log M + N \times overhead) \tag{1}$$

where N means number of threads

We could see, the order of running time doesn't change. Therefore, It's uncertain the performance of parallel computing would be better when M isn't huge enough.

When M is huge enough, we could say, with the increase of number of threads, the performance would be better.