

PL HW4

Matrix Multiplication

第18組

張哲 孫上智 郭彥松 許雁婷





使用語言

- C
- OpenMP



傳統矩陣乘法 (Serial + Parallel)

平行for迴圈

openmp for

#pragma omp parallel for

```
#pragma omp parallel for
for (int i=0; i < row; i++){
    for (int j=0; j < col; j++){
        for(int k=0; k < tmp; k++){
            ansMatrix[i][j] += fMatrix[i][k]*sMatrix[k][j];
        }
    }
}
```

傳統矩陣乘法 (Serial + Parallel)

```
Randy1005 > PL_hwd ls
24466 genMatrix.c naive str1.c strassen.c st
0.000 input naive.c strassen strassen_parallel

Randy1005 > PL_hwd ./naive input/input256x256.txt
First Matrix:
Second Matrix:
256*256 matrix
Serial Naive MM Time: 0.144488 sec
Parallel Naive MM Time: 0.038562 sec

Randy1005 > PL_hwd ./naive input/input1024x1024.txt
First Matrix:
Second Matrix:
1024*1024 matrix
Serial Naive MM Time: 25.146799 sec
Parallel Naive MM Time: 2.378790 sec

Randy1005 > PL_hwd ./naive input/input2048x2048.txt
First Matrix:
Second Matrix:
2048*2048 matrix
Serial Naive MM Time: 239.483646 sec
Parallel Naive MM Time: 24.181358 sec

Randy1005 > PL_hwd ./naive input/input4096x4096.txt
First Matrix:
Second Matrix:
4096*4096 matrix
Serial Naive MM Time: 1970.736213 sec
```

Strassen's ALGO (Serial)

```
Randy1005 > PL_hwd ./strassen input/input16x16.txt
First Matrix:
Second Matrix:
Serial Strassen's Algorithm: 0.000457 sec.
Randy1005 > PL_hwd ./strassen input/input256x256.txt
First Matrix:
Second Matrix:
Serial Strassen's Algorithm: 0.076512 sec.
Randy1005 > PL_hwd ./strassen input/input1024x1024.txt
First Matrix:
Second Matrix:
Serial Strassen's Algorithm: 3.590760 sec.
Randy1005 > PL_hwd ./strassen input/input2048x2048.txt
First Matrix:
Second Matrix:
Serial Strassen's Algorithm: 27.002743 sec.
Randy1005 > PL_hwd
```

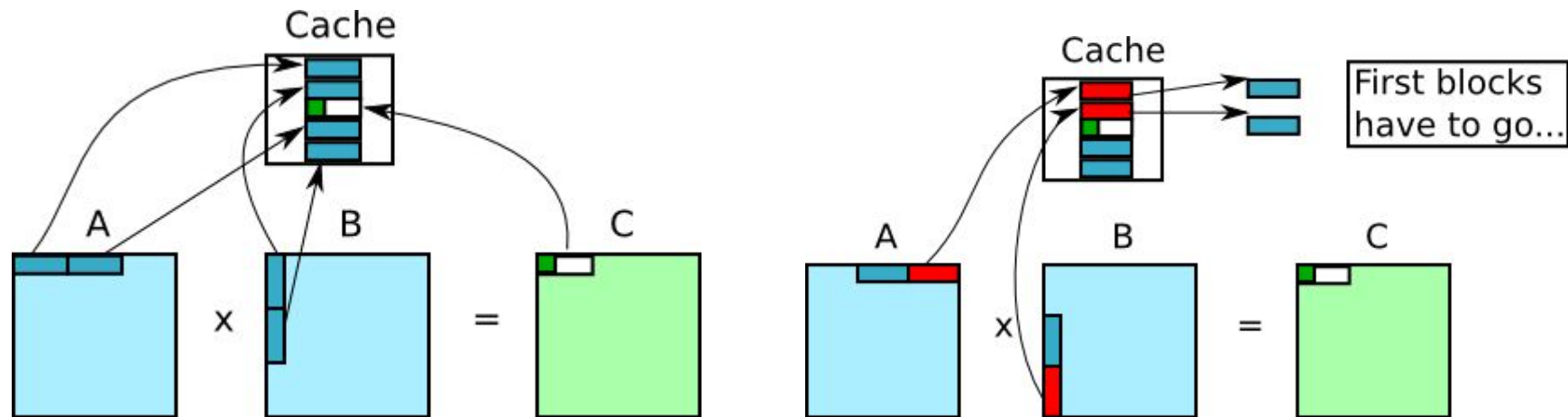


Strassen's ALGO (Parallel/no optimization)

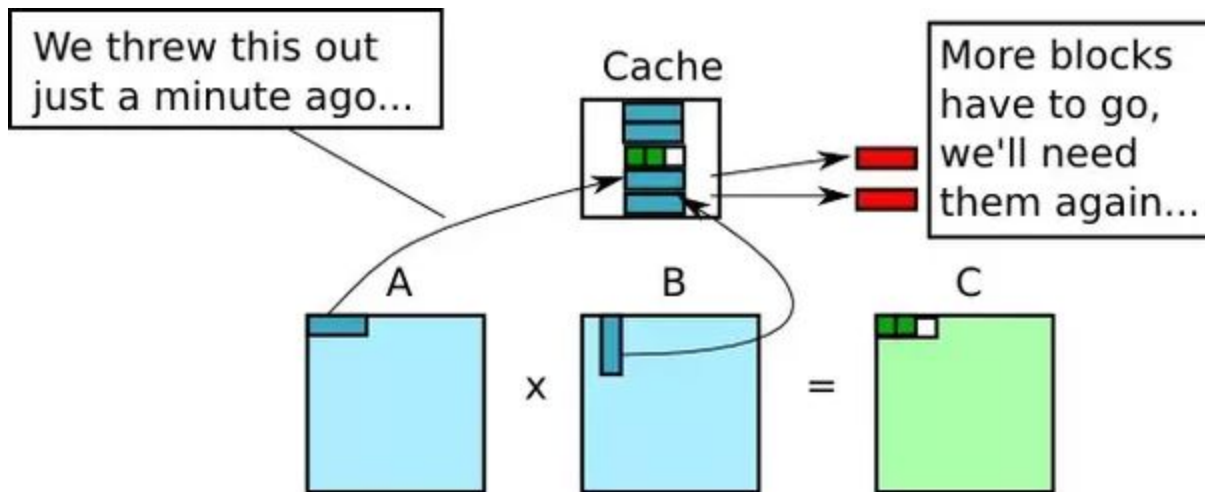
平行 baseline 矩陣乘法的 for 迴圈
openmp for
#pragma omp parallel for

```
Randy1005 > PL_hw4 gcc-6 -o strassen_para strassen_para.c -fopenmp
Randy1005 > PL_hw4 gcc-6 -o strassen_para strassen_para.c -fopenmp
Randy1005 > PL_hw4 ./strassen_para input/input16x16.txt
First Matrix:
Second Matrix:
Parallel Strassen's Algorithm (no optimization): 0.003864 sec.
Randy1005 > PL_hw4 ./strassen_para input/input256x256.txt
First Matrix:
Second Matrix:
Parallel Strassen's Algorithm (no optimization): 0.058375 sec.
Randy1005 > PL_hw4 ./strassen_para input/input1024x1024.txt
First Matrix:
Second Matrix:
Parallel Strassen's Algorithm (no optimization): 3.688001 sec.
Randy1005 > PL_hw4 ./strassen_para input/input2048x2048.txt
First Matrix:
Second Matrix:
Parallel Strassen's Algorithm (no optimization): 30.421891 sec.
Randy1005 > PL_hw4 ./strassen_para input/input4096x4096.txt
First Matrix:
Second Matrix:
```

其他加速方法: cache locality



其他加速方法



```
for (int i=0; i < row; i++){  
    for (int j=0; j < col; j++){  
        for(int k=0; k < tmp; k++){  
            ansMatrix[i][j] += fMatrix[i][k]*sMatrix[k][j];  
        }  
    }  
}
```

```
for (int i=0; i < row; i++){  
    for (int k=0; k < col; k++){  
        for(int j=0; j < tmp; j++){  
            ansMatrix[i][j] += fMatrix[i][k]*sMatrix[k][j];  
        }  
    }  
}
```




Strassen's ALGO (Parallel/optimization)

- cache locality + 只做2層recursion 就換baseline的乘法

```
if(size <= (row/4)){  
    mat_mul(size, fMat, sMat, ansMat);  
}
```

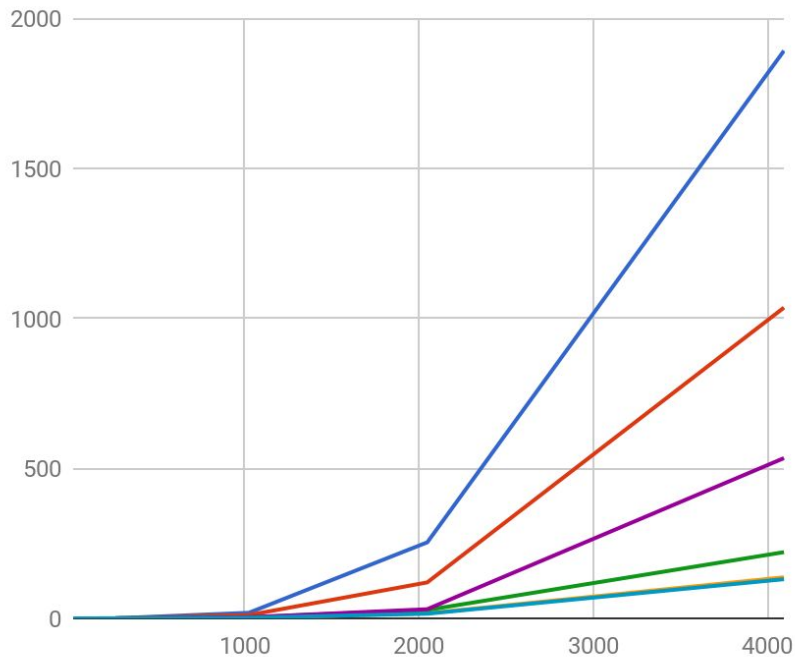
Strassen's ALGO (Parallel/optimization)

```
Wandy1005 PL_hw4 gcc-6 -o strassen_para strassen_para.c -fopenmp
Wandy1005 PL_hw4 gcc-6 -o strassen_para strassen_para.c -fopenmp
Wandy1005 PL_hw4 ./strassen_para input/input16x16.txt
First Matrix:
Second Matrix:
Parallel Strassen's Algorithm (cache - friendly): 0.002790 sec.
Wandy1005 PL_hw4 ./strassen_para input/input256x256.txt
First Matrix:
Second Matrix:
Parallel Strassen's Algorithm (cache - friendly): 0.017955 sec.
Wandy1005 PL_hw4 ./strassen_para input/input1024x1024.txt
First Matrix:
Second Matrix:
Parallel Strassen's Algorithm (cache - friendly): 1.983462 sec.
Wandy1005 PL_hw4 ./strassen_para input/input2048x2048.txt
First Matrix:
```

CC project set	00:00
compiler set	00:00



比較



size(n)	Naive	P Naive	Strassen	P Strassen	Optimize
16	0.000076	0.000598	0.000388	0.003864	0.000598
256	0.132198	0.042979	0.076512	0.058375	0.042979
1024	18.512002	2.005506	3.590768	3.688001	2.005506
2048	253.222766	16.767355	27.802743	30.4211891	16.767355
4096	1891.756694	136.73551	220.610812	534.010275	136.73551

- Naive
- Parallel Naive
- Parallel Naive(Cache)
- Strassen's ALGO
- Parallel Strassen's
- Parallel Strassen's (Cache)



分工

張哲: strassen

孫上智: strassen加速

郭彥松: 傳統

許雁婷: 傳統加速