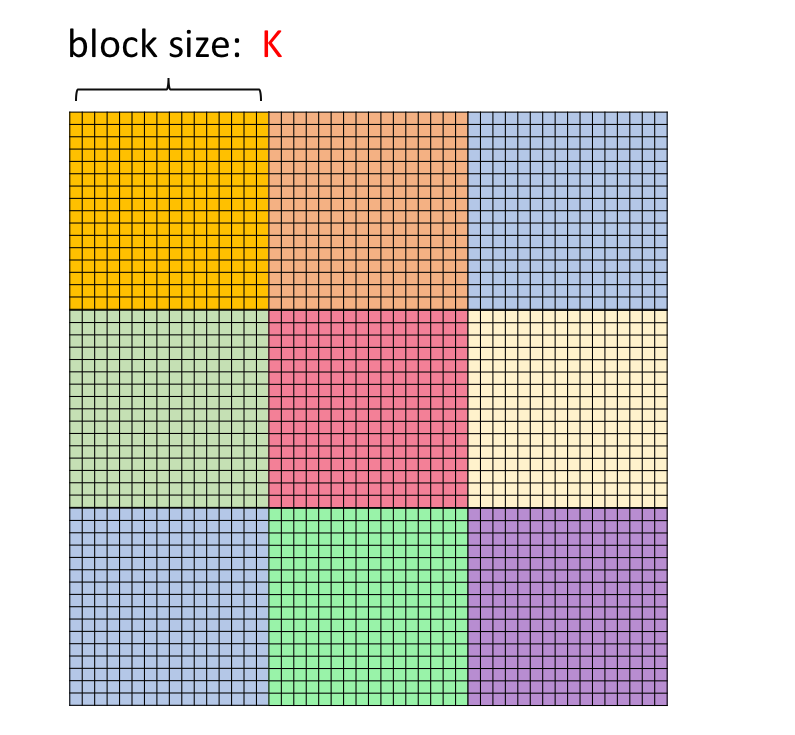
Parallel Programming hw4-1 ----Blocked Folyd-Warshall algo.

學號：109062639 姓名：葉哲欣

**Implementation:**

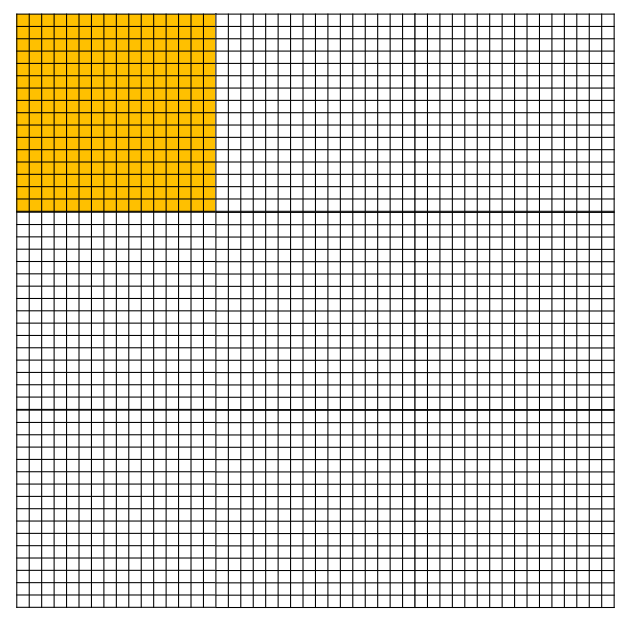
K:32

將distance matrix(data)依照每個block size（K\*K）來分割data。並依照演算法3個kernel function(phase )用到的data dependency block 來宣告block數以及share memory 數。

共需執行Round = ceil(V/k) 次，Round\*Round個block

Phase1

phase1每個round只需運算一個block(黃色)的大小data (block(k,k))

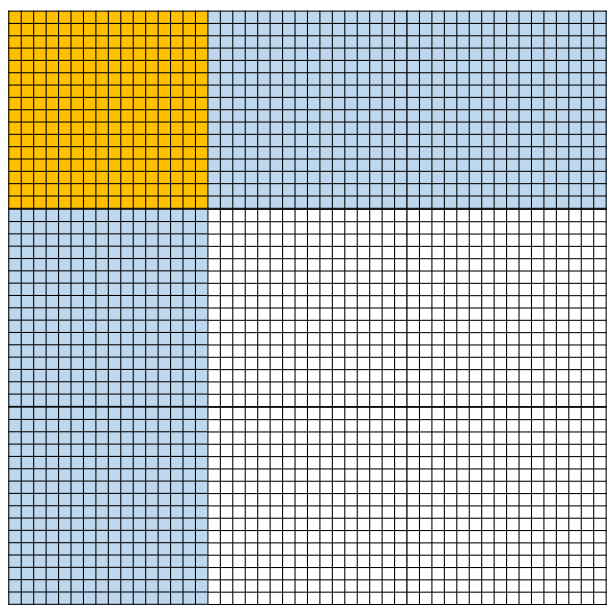
故phase1所需thread及block：

dim3 ThreadPerBlock(k,k)

dim3 BlockPerGrid(1,1)

並且每個block所使用到的share memory size = K\*K

Phase2



Phase2每個round需運算一個2\*Round個block(藍色+黃色)

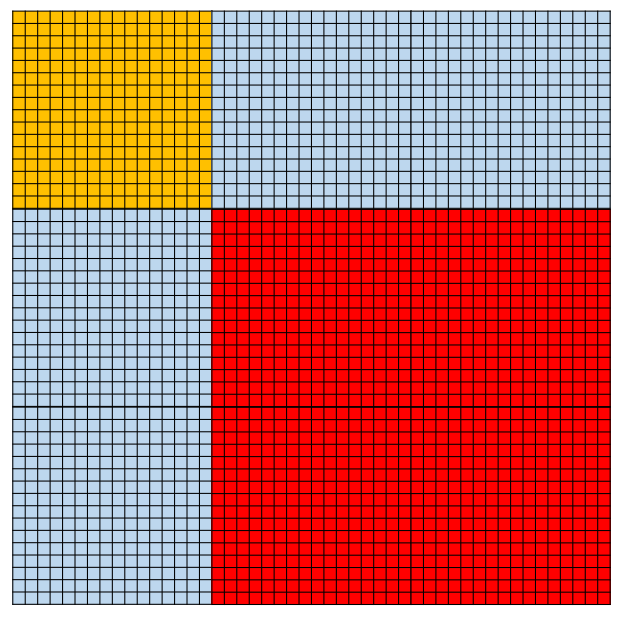
故phase2所需thread及block：

dim3 ThreadPerBlock(k,k)

dim3 BlockPerGrid(round,2)

而每個block會需要用到自己及phase1的data

故每個block所使用到的share memory size = 2\*K\*K

Phase3

Phase3每個round需運算一個(round-1)\*(round-1)個block(藍色+紅色)

故phase3所需thread及block：

dim3 ThreadPerBlock(k,k)

dim3 BlockPerGrid(round,round)

而每個block會需要用到自己及phase2的row 及col data

故每個block所使用到的share memory size = 3\*K\*K

**Profiling Results**

Occupancy

|  |  |
| --- | --- |
| **kernel** | **Achieved occupancy(avg.)** |
| Phase1 | 0.496326 |
| Phase2 | 0.942633 |
| **Phase3** | **0.896393** |

sm efficiency

|  |  |
| --- | --- |
| **kernel** | **sm efficiency** |
| Phase1 | 3.66% |
| Phase2 | 98.39% |
| **Phase3** | **99.97%** |

Share memory load/store throughput

|  |  |  |
| --- | --- | --- |
| **kernel** | **Load throughput** | **Store throughput** |
| Phase1 | 57.631GB/s | 19.755GB/s |
| Phase2 | 2429.2GB/s | 1249.9GB/s |
| **Phase3** | **2970.5GB/s** | **92.716GB/s** |

Global load/store throughput

|  |  |  |
| --- | --- | --- |
| **kernel** | **Load throughput** | **Store throughput** |
| Phase1 | 364.46MB/s | 741.18MB/s |
| Phase2 | 16.554GB/s | 44.787GB/s |
| **Phase3** | **19.902GB/s** | **56.468GB/s** |

**Experiment and Analysis**

**System Spec:** hades server

**Time distribution:**

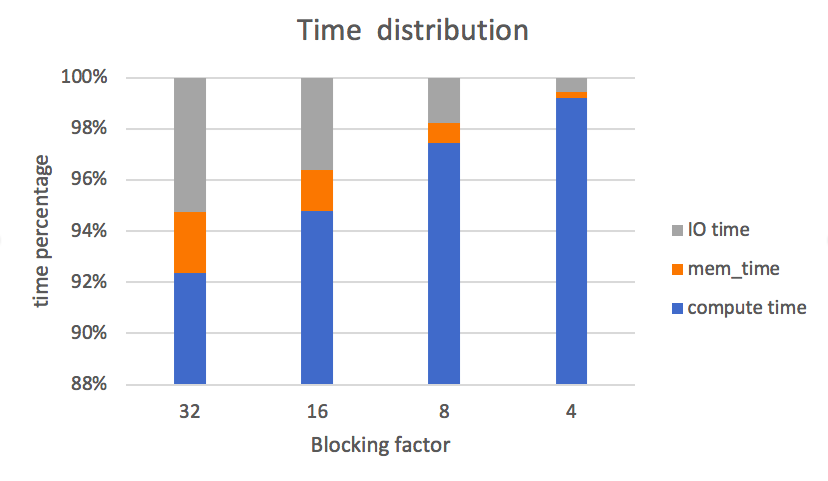
採用 nvprof來查詢 kernel compute 以及memry htd,dth time

使用timespec來紀錄I/O time

(a.)

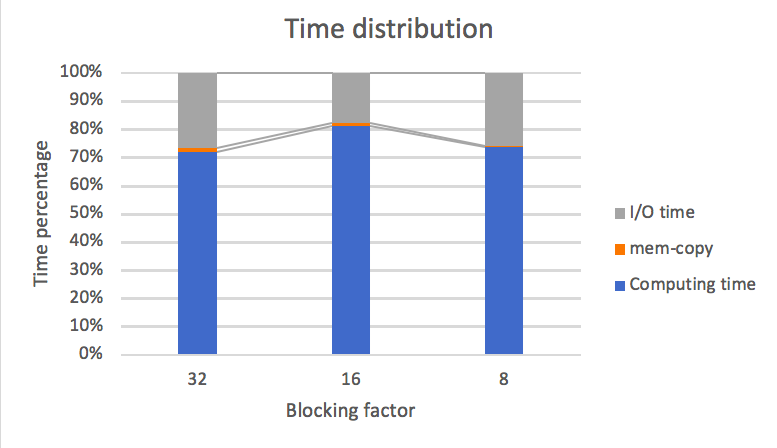
Test case:p21k

|  |  |  |  |
| --- | --- | --- | --- |
| **Block-factor** | **Computing time** | **copy time** | **I/O time** |
| **Block =32** | 20.813 s | 0.53733 s | 1.185324 s |
| **Block =16** | 33.0087 s | 0.54889 s | 1.199476 s |
| **sBlock =8** | 68.9008 s | 0.55201 s | 1.256658 s |
| **Block=4** | 227.112 s | 0.54833 s | 1.29825 s |



(b.)

Test case:p31k



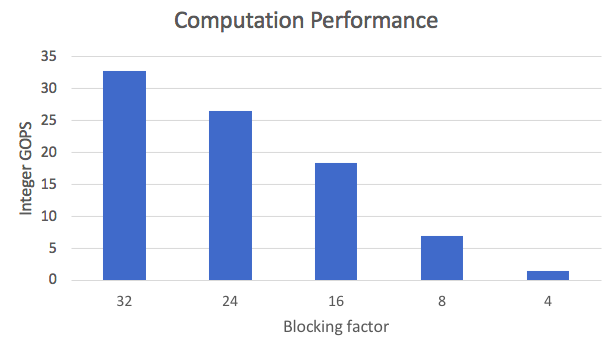
|  |  |  |  |
| --- | --- | --- | --- |
| **Block-factor** | **Computing time** | **copy time** | **I/O time** |
| **Block =32** | 64.9908 | 1.21186 | 23.973353 |
| **Block =16** | 100.281 | 1.20125 | 22.019499 |
| **Block =8** | 203.924 | 1.20078 | 71.302255 |

**Blocking factor:**

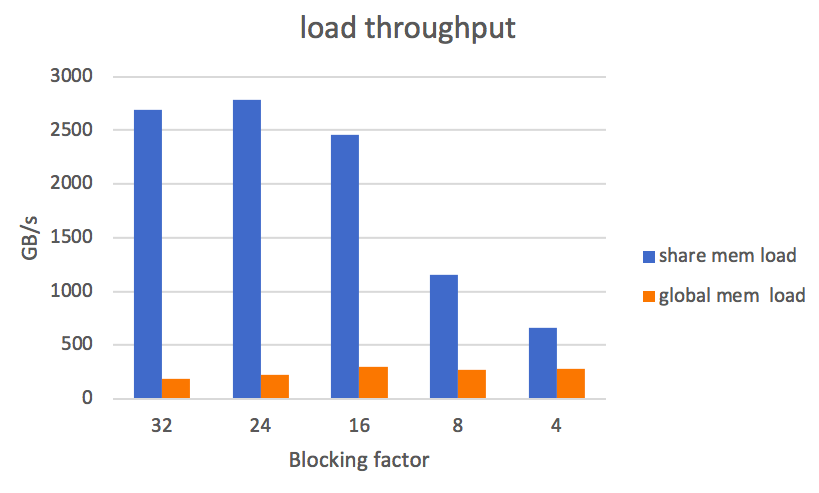
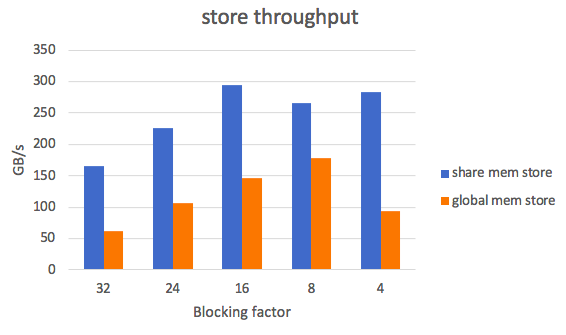
採用 phase3作為compute time標準

Computation Performance:

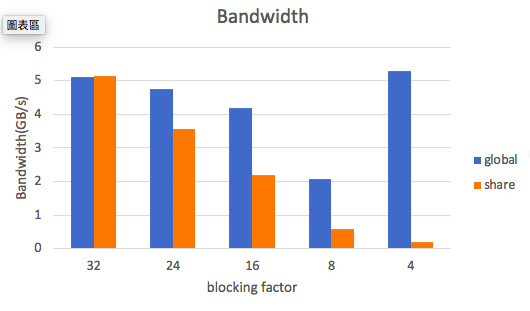
GOPS:



Global memory memory / shared memory load/store throughput :

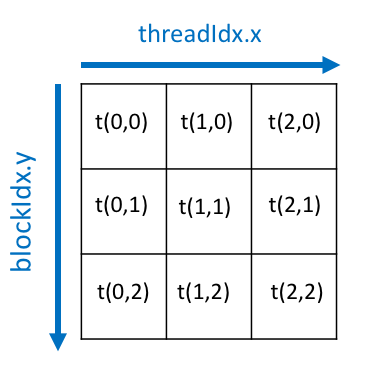
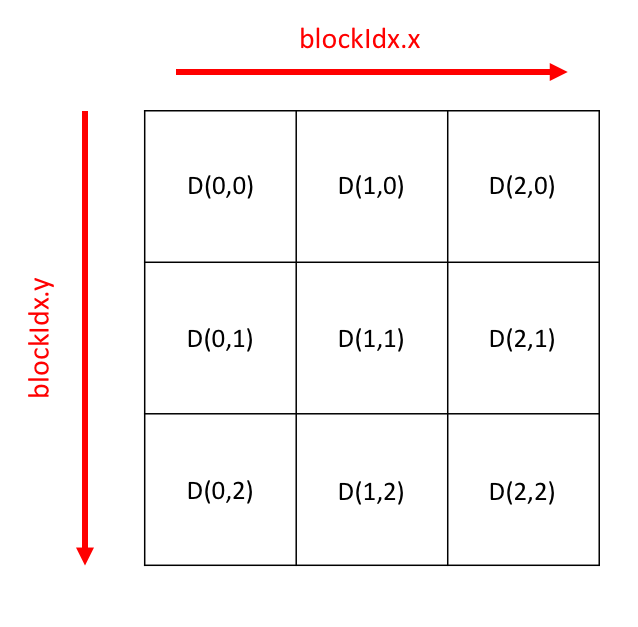
 

Global memory memory / shared memory load/store bandwidth :



**Optimization :**

1. using 2 dim block
2. memory coalesced

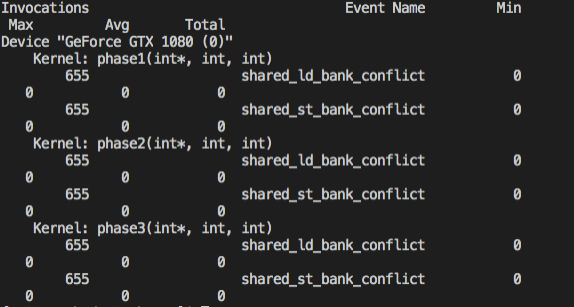
memory access 為 column major,而threadId 編號是以x 為major,若直接以threadIdx.x,threadId.y轉換為x,y座標，則會發生不連續memory access 

1. shared memory

使用share memory 來提升存取速度

4.avoid bank conflict

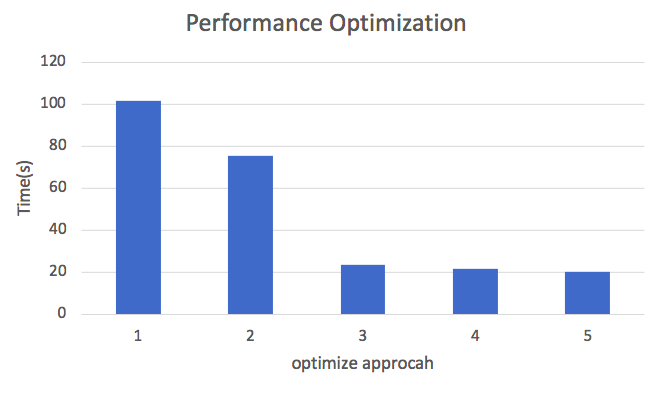
用 memory padding(share memory:block\_size\*(block\_size+1)) 方式來避免



5.unrolling

使用pragma unroll 來展開迴圈

|  |  |  |
| --- | --- | --- |
| 1 | 2D | 101.599 |
| 2 | 2D + share\_memory | 75.4432 |
| 3 | 2D + share\_memory +memoy coalsed | 23.3685 |
| 4 | 2D + share\_memory + bankconflict+memoy coalsed | 21.477 |
| 5 | 2D + share\_memory + bankconflict+memoy coalsed +unroll | 20.3403 |



**Experiment and Analysis**

從這次作業我學到如何運用cuda來做gpu的運算，來加速程式運算。

透過存取Share memory方式凸顯global memory存取速度上的差異性，並且可以透過coalesced memory ,memory padding等等各式各樣的優化方式讓效能最佳化。本次作業困難點就是block 與block 彼此之間資料的傳輸，data該如何分割，如何透過blockIdx,以及thredIdx mapping到data distance上 ，並存入shared memory 。另外發現若使用過多的 share memory效能不一定會越好，過多的thread 在syncthreads 會嚴重影響效能。 因此我覺得還可以改進的部分就是減少 syncthreads 時間。