CS3210 Parallel Computing

NUS AY 2022-23 Sem 1

Assignment 2 Report

Matric Number	Name
A0223111E	Anderson Leong Ke Sheng
A0220597B	Eric Bryan

Terminologies.
BLOCK_SIZE = number of threads per block = 128
GRID_SIZE = number of blocks per grid = 96

Section 1. Strategies

1.1 Parallelisation strategy

Our implementation consists of **GRID_SIZE** blocks per grid, with **BLOCK_SIZE** threads per block. Both grid and threads per block have a dimension of one. We use a logical dimension of 1 as we do not utilise the 2D or 3D topology in our implementation.

Each file is split into **BLOCK_SIZE** sections where each section is handled (checked for a matching signature) by each of the **BLOCK_SIZE** threads per block in parallel. Each block checks for a match of 1 signature per file (which as mentioned before, is split into **BLOCK_SIZE** sections running in parallel). There are **GRID_SIZE** numbers of such blocks running in parallel, checking for **GRID_SIZE** different signatures per file at the same time. We also maintain an array of counters that correspond to the number of signature-file matches for every signature-file pair. Whenever there's a file that contains a signature, we would increase the counter corresponding to this signature-file pair.

Block shared memory is not used. We used a unified memory, which has common memory addressing space that allows both the GPU and CPU to access them, to allow the GPU to write the number of times each signature appears in a file, and allow the CPU to read the values to print the output statements.

1.2 Methods used to increase Fb for program

We use the O(mn) brute-force approach where each index in the file is checked against a possible signature match. We do this for every signature-file pair. This method ensures a 100% True positive hit rate with no false positives or false negatives. Thus no additional method is used to increase the F-beta of the program which is always 1.

Section 2. Which aspect of the input size affects speedup the most for your implementation

Taking various measurements from different inputs where number of input files, number of signatures, size of input files, length of signatures are all varied. We discover that the number of input files, number of signatures, and size of input files all similarly affect speedup, while the length of signatures had less impact on speedup.

Comparing the results in appendix 2.1.a and appendix 2.1.b, we see that the speedup increased from 209.565x when the number of signatures were around 10000 to 237.607x at 20000 signatures. To reduce runtime, we compare the other aspects of the input using around 2000 signatures.

Using a database containing 2195 signatures of average length of 47 (`2k-sigs-short.txt`) and 80 input files of file sizes of 512kb as a basis of comparison, We achieve a speedup of 196.193x (appendix 2.1.c).
We observe a speedup increase to 251.965x is achieved when the file sizes all double to 1Mb (appendix 2.1.d); a speedup increase to 236.350x is achieved when the number of input files doubles to 160 (appendix 2.1.e); and a speedup increase to 204.698x is achieved when the average length of the signatures roughly doubles to 104 in the database (`2k-sigs-long.txt`) from 47 (appendix 2.1.f).

Thus we conclude that the size of the input files affects speedup the most.

2.1 Why do you think this is the case?

We observe that both the number of input files and the file sizes affect the speedup drastically. However, the sizes of the input files seem to matter more compared to the number of input files even though in both tests, we double the value compared to the basis of comparison. There is more speedup for when the file sizes are larger despite both cases having the same increase in input size. In our case, our algorithm splits the files that it intakes into **BLOCK_SIZE** number of sections to concurrently check for the signature results in larger file sizes bringing more speedup compared to more number of files.

To illustrate, in case one where have a total of **BLOCK_SIZE** chunks of size **1024Kb/BLOCK_SIZE** when a 1Mb file is split, whereas in a second case where 2 files of 512Kb as an equivalent to make 1Mb results in a total of **2 * BLOCK_SIZE** chunks of the size **2 * 512Kb/BLOCK_SIZE**.

The first case would have to take **BLOCK_SIZE/32** warps to complete whereas the second case takes **2 * BLOCK_SIZE/32** warps to complete, twice that of the first case. Even though it would take less time for the second case to complete each warp due to it's smaller size, there are some overhead cost incurred when each block is initialised with the memory required, these additional time incurred to read and write more memory and taking more warps leads to **file size having the most impact on the speedup** as opposed to number of files.

Section 3. Description of ONE performance optimisation attempted

With analysis and supporting performance measurements. Refer to an optimisation related to your parallelization efforts. Include at least one summary graph in your report and link to your supporting measurements.

For optimisation, we sought to optimise execution configuration to strike a balance between occupancy and resource utilization by reducing the threads per block, which is BLOCK_SIZE to an ideal value by measuring the speedups on the same runs. A balanced value of threads per block ensures that the blocks are always kept busy by having the number of warps stay larger than the number of multiprocessors which is around 108 for node xgpg7 to improve occupancy while **BLOCK_SIZE** is also a multiple of the warp size of 32 as well, being **128** threads to utilize resources in the best way. Refer to Appendix 3 for measurements.

BONUS

- **1 Bonus mark for:**
  - 100% accurate CUDA implementation faster than the multithreaded sequential implementation: Yes
  - Parallel speedup: 9.829x (xgpd7) ; 4.355x (xgpg7)
  - F-beta: 1.0000

To replicate run this line on the nodes specified: `./check.py signatures/10k_sigs.txt mytest/10k_try_128/*`

run on node xgpd7	**run on node xgpg7**

`loaded 11205 signature(s), 128 input file(s)`
`cuda stats:`
`  # of SMs: 80`
`  global memory: 12066.69 MB`
`  shared mem per block: 49152 bytes`
`  constant memory: 65536 bytes`
`-------------------------`
`false positives:`
`none`
`-------------------------`
`false negatives:`
`none`
`-------------------------`
 	
`summary:`
`  true pos:  355`
`  false pos:  0`
`  false neg:  0`
 	
`time taken: 13.934s vs 4340.316s`
`speedup: 311.495x`
`F-beta = 1.0000`
 	
`parallel speedup (for bonus): 9.829x`
	`loaded 11205 signature(s), 128 input file(s)`
	`cuda stats:`
	`  # of SMs: 108`
	`  global memory: 40536.19 MB`
	`  shared mem per block: 49152 bytes`
	`  constant memory: 65536 bytes`
	`-------------------------`
	`false positives:`
	`none`
	`-------------------------`
	`false negatives:`
	`none`
	`-------------------------`
	
	`summary:`
	`  true pos:  355`
	`  false pos:  0`
	`  false neg:  0`
	
	`time taken: 7.619s vs 3094.793s`
	`speedup: 406.196x`
	`F-beta = 1.0000`
	
	`parallel speedup (for bonus): 4.355x`

Appendix 2 Replicate Results in Section 2

Replicating results:
Nodes used:
xgpg7.comp.nus.edu.sg

Appendix:	Command
2.1.	`./check.py signatures/sigs-both.txt tests/*`
2.1.a	`./check.py signatures/new-10k-sigs.txt mytest/new10k-512-40/*`
2.1.b	`./check.py signatures/sigs-20k.txt mytest/20k-512-40/*`
2.1.c	`./check.py signatures/2k-sigs-short.txt mytest/2k_s512/*`
2.1.d	`./check.py signatures/2k-sigs-short.txt mytest/2k_s1mb/*`
2.1.e	`./check.py signatures/2k-sigs-short.txt mytest/2k_s512_160/*`
2.1.f	`./check.py signatures/2k-sigs-long.txt mytest/2k_s512_long/*`
2.1.g	`./check.py signatures/10k_sigs.txt mytest/10k_try_128/*`

Performance measurements:

2.1

```
loaded 2000 signature(s), 21 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    27
  false pos:   0
  false neg:   0

time taken: 0.5ms vs 59.932s
speedup: 123.065x
F-beta = 1.0000

parallel speedup (for bonus): 1.437x
```

2.1.a Testing on ~10k signatures

```
loaded 10150 signature(s), 40 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    131
  false pos:   0
  false neg:   0

time taken: 2.251s vs 471.651s
speedup: 209.565x
F-beta = 1.0000

parallel speedup (for bonus): 2.313x
```

2.1.b Testing on ~20k signatures

```
loaded 20055 signature(s), 40 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    141
  false pos:   0
  false neg:   0

time taken: 3.909s vs 928.910s
speedup: 237.607x
F-beta = 1.0000

parallel speedup (for bonus): 2.613x
```

2.1.c Basis of comparison

```
loaded 2195 signature(s), 80 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    255
  false pos:   0
  false neg:   0

time taken: 1.067s vs 209.295s
speedup: 196.193x
F-beta = 1.0000

parallel speedup (for bonus): 2.190x
```

2.1.d Double to file size

```
loaded 2195 signature(s), 80 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    244
  false pos:   0
  false neg:   0

time taken: 1.789s vs 450.782s
speedup: 251.965x
F-beta = 1.0000

parallel speedup (for bonus): 2.759x
```

2.1.e Double the number of input files

```
loaded 2195 signature(s), 160 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    497
  false pos:   0
  false neg:   0

time taken: 1.822s vs 430.737s
speedup: 236.350x
F-beta = 1.0000

parallel speedup (for bonus): 2.652x
```

2.1.f Longer signature

```
loaded 2195 signature(s), 80 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    281
  false pos:   0
  false neg:   0

time taken: 1.022s vs 209.100s
speedup: 204.698x
F-beta = 1.0000

parallel speedup (for bonus): 2.342x
```

2.1.g Minimal Criteria of at least 10k signatures and at least 10 input files of at least size 512

```
loaded 11205 signature(s), 128 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    355
  false pos:   0
  false neg:   0

time taken: 7.648s vs 3094.793s
speedup: 404.669x
F-beta = 1.0000

parallel speedup (for bonus): 4.339x
```

Appendix 3

SUPPORTING MEASUREMENTS for PERFORMANCE OPTIMISATION

To replicate, run on node **xgpg7**: `./check signatures/10k_sigs.txt mytest/10k_try_128/*`

And vary the **BLOCK_SIZE** according to the table.

BLOCK_SIZE = 512
Speedup: 297.815x

```
loaded 11205 signature(s), 128 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    355
  false pos:   0
  false neg:   0

time taken: 10.392s vs 3094.793s
speedup: 297.815x
F-beta = 1.0000

parallel speedup (for bonus): 3.193x
```

BLOCK_SIZE = 256
Speedup: 376.477x

```
loaded 11205 signature(s), 128 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    355
  false pos:   0
  false neg:   0

time taken: 8.220s vs 3094.793s
speedup: 376.477x
F-beta = 1.0000

parallel speedup (for bonus): 4.037x
```

BLOCK_SIZE = 128
Speedup: 404.669x **(BEST)**

```
loaded 11205 signature(s), 128 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    355
  false pos:   0
  false neg:   0

time taken: 7.648s vs 3094.793s
speedup: 404.669x
F-beta = 1.0000

parallel speedup (for bonus): 4.339x
```

BLOCK_SIZE = 64
Speedup: 268.255x

```
loaded 11205 signature(s), 128 input file(s)
cuda stats:
  # of SMs: 108
  global memory: 40536.19 MB
  shared mem per block: 49152 bytes
  constant memory: 65536 bytes
-------------------------
false positives:
none
-------------------------
false negatives:
none
-------------------------

summary:
  true pos:    355
  false pos:   0
  false neg:   0

time taken: 11.537s vs 3094.793s
speedup: 268.255x
F-beta = 1.0000

parallel speedup (for bonus): 2.876x
```