ECE284 - Assignment 2

Ishan Bansal
A59025917
WI24

5.a.

a. kmerPosConcat:

blocks \ size	**32**	**128**	**512**
64	1ms	845.34us	5.75ms
256	844.7us	4.73ms	4.53ms
1024	5.18ms	3.85ms	4.17ms

b. kmerOffsetFill:

blocks \ size	**32**	**128**	**512**
64	98.02ms	141.94ms	507.47ms
256	116.37ms	386.44ms	1.15s
1024	346.08ms	971.9ms	3.51s

c. kmerPosMask:

blocks \ size	**32**	**128**	**512**
64	1.562ms	1.15ms	4.94ms
256	1.17ms	4.61ms	9.20ms
1024	5.20ms	9.35ms	8.05ms

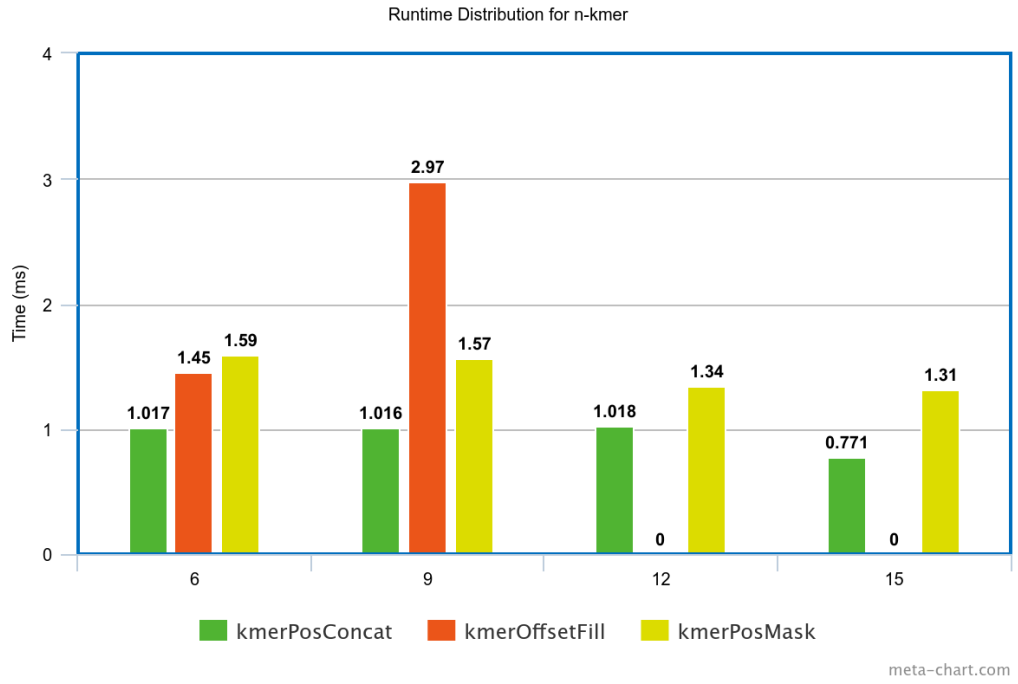
b. The best overall performance is seen when the GPU parameters are:
numBlocks: 64
blockSize: 32
Total time taken: ~109ms

c. Speedup vs Sequential implementation:
i. kmerPosConcat:
    Tseq = 764.34ms
    Tpar = 1ms
    Speedup = 764
ii. kmerOffsetFill:
    Tseq = 1.68s
    Tpar = 98.02ms
    Speedup = 17.13
iii. kmerPosMask:
    Tseq = 1.01s
    Tpar = 1.56ms
    Speedup = 647

d. Runtimes of the GPU activities:
    i. cudaMalloc: Total: 129.27ms for 4 calls at an avg of 32.31ms per call
    ii. CUDA memcpy HtoD: 615.74us
    iii. CUDA memcpy DtoH: Total: 3.48us for 2 calls at an avg of 1.74us per call
    iv. cudaMemcpy: Total: 855.65us for 3 calls at an avg of 285.22us per call
    v. cudaFree: Total: 8.5ms for 4 calls at an avg of 2.12ms per call

GPU activities also take up time which adds to the temporal overhead of our execution. Commands such as cudaMalloc and cudaFree, which are used for allocating and freeing up memory respectively, are the most time consuming calls. Out of these, cudaMalloc is especially taxing and takes approx 20% more time than the three functions combined.

e.



PS: kmerOffsetFill for 12-kmer and 15-kmer has been left empty in order to let the rest of the chart give context for a clean comparison in the above chart.
For: 12-kmer, kmerOffsetFill takes 99ms which is ~33x the time taken for 9-kmer
For: 15-kmer, kmerOffsetFill takes 4.32s which is ~43x the time taken for 12-kmer