CSci 5451: Parallel Computing–Spring 2011

Homework 4: CUDA Longest Common Subsequence

Harlan Iverson

April 13, 2011

The solution is not working code. The idea was to create a grid with blocks and have them computed in a diagonal order, from top left to bottom right. Each block depends on the ones directly above and to the left being computed. In order to preserve spatial/temporal locality, a mapping that stores rows of the grid in a diagonal-major order was used.

The longer string should be along the y axis, and the shorter string along the x axis. This would allow a maximum number of concurrent blocks for the most steps, and help spatial locality. The first and last steps would be a single block; and each step from the beginning would have incrementally more until a maximum is reached, and then each step toward the end would have incrementally less.

The code is semi-complete and the algorithm works in serial using a for loop that 'simulates' the device by looping over x values (and uses a 1x1 block configuration, since a thread would be in charge of a single column). The working implementation is in lcs_serial.cu. The non-working partial CUDA port is in lcs_cuda.cu. Some of the calculation code is in calc.c.

The procedure of the calculation is as follows:

- Create a grid of blocks(maxb, maxt, width, height)
  - ensure that maxt*maxb <= width
  - block width = maxt
  - block height = some constant
  - # y blocks = ceil(h / block height)
  - # x blocks = ceil(w / block width)
  - # of diagonals = # x blocks + # y blocks - block width
- for each diagonal (d = 0 to # of diagonals - 1)
  - start block = max(d - # x blocks - 1, 0)

- end block = min(d, # x blocks - b)
- # blocks = end block - start block
- run CUDA with <<<# blocks, maxt>>>(grid, start block, d, block height)
    * block x = (start block + cuda.blockIdx)*cuda.blockDim [* block width is covered by blockDim]
    * block y = ((d+1) * block height) - block x
    * block x += 1, block y += 1
    * for y = block y to block y + block height
        · do LCS cell procedure on (x,y)

Trace back is identical to the serial counter-part.