GPU-Accelerated 3-Dimensional Reaction Diffusion Systems

John K. Holmen

Department of Electrical & Computer Engineering

Kettering University

Advised By: David L. Foster

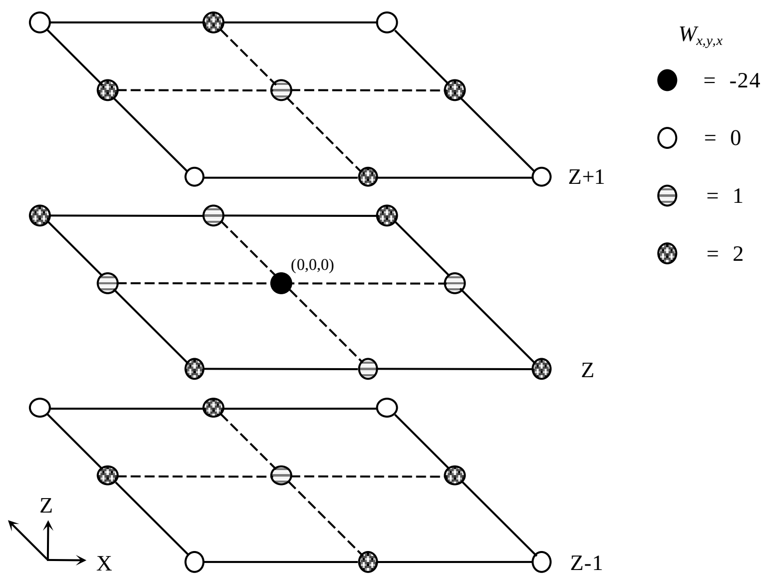
Reaction-Diffusion Systems Overview

- Model diffusion and/or chemical reactions occurring within a concentration of one or more chemical species over a given time period
  - Useful for describing heat diffusion, climate science, etc…
- Focus placed on solving for the diffusion of a chemically inert compound
- Success allows results to be obtained quicker and/or larger simulated spaces, additional time steps, and/or higher resolutions to be achieved

Diffusion Algorithm

$$c_{i,j,k}(t+\partial t) = c_{i,j,k}(t) + \left[D\left(\left(\frac{1}{6h^2}\right)\sum_{x=-1,y=-1,z=-1}^{x=1,y=1,z=1}(W_{x,y,z}c_{i+x,j+y,k+z})\right)\right]\partial t$$

Symmetric, Nearest Neighbor 19-Point Stencil



Serial Pseudocode

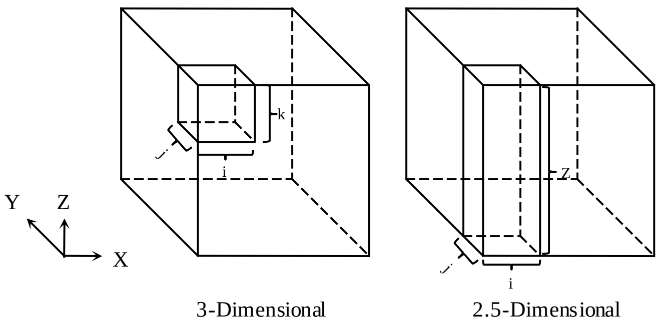
```
For all z from halo to zmax-halo
    For all y from halo to ymax-halo
        For all x from halo to xmax-halo
            Laplace =  old[z+1][y-1][x]+             //Edges
                       old[z+1][y][x-1]+
                       old[z+1][y][x+1]+
                       old[z+1][y+1][x]+
                       old[z][y-1][x-1]+
                       old[z][y-1][x+1]+
                       old[z][y+1][x-1]+
                       old[z][y+1][x+1]+
                       old[z-1][y-1][x]+
                       old[z-1][y][x-1]+
                       old[z-1][y][x+1]+
                       old[z-1][y+1][x]+
                       2*(old[z+1][y][x]+            //Faces
                       old[z][y-1][x]+
                       old[z][y][x-1]+
                       old[z][y][x+1]+
                       old[z][y+1][x])-
                       24*old[z][y][x])              //Center
            new[z][y][x] = old[z][y][x]+(0.02/6)*Laplace
```

Stencil computation results in a memory bound algorithm (0.29 flops/byte)

Explored Optimization Techniques

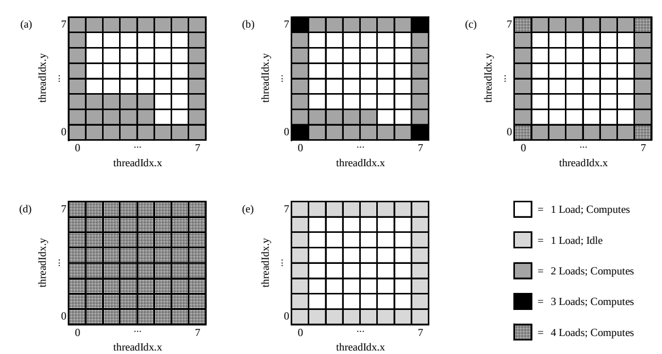
Spatial Blocking Techniques

MOTIVATION: To reduce memory bandwidth requirements associated with loading of the *halo region* (the layer of data points immediately surrounding a data point that computes)



Halo Region Loading Techniques

MOTIVATION: To determine the ideal balance between memory accesses, divergence, and computing threads

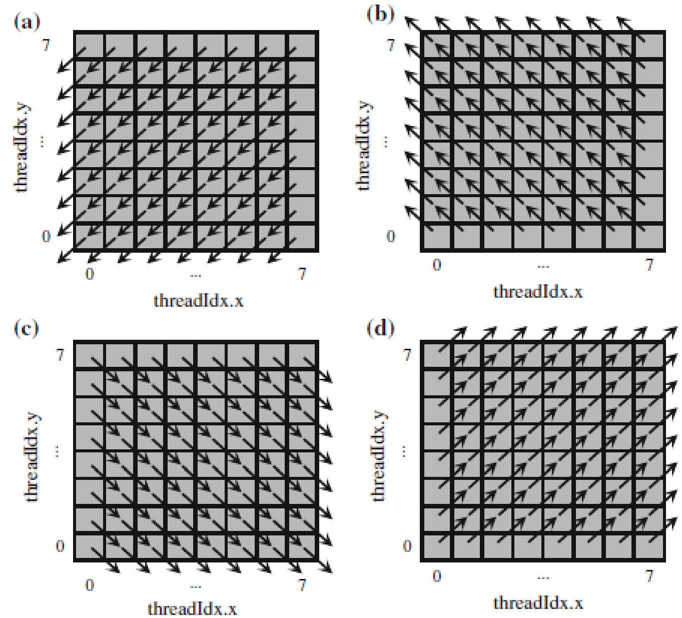


(a)	Multi-Load (2-Load Max)	(d)	Quadrant-Load
(a)	Multi-Load (3-Load Max)	(e)	Single-Load
(b)	Multi-Load (4-Load Max)		

Significant Outcomes

- 2.5-dimensional blocking with the "Quadrant-Load" halo region loading technique proved most advantageous
- Up to 8.69x speedup (3995.05 Mpoints/s) in reference to a multithreaded CPU implementation for simulations conducted on GeForce GTX 260, Tesla C1060, and GeForce GTX 560 Ti GPUs
  - 10,000 time steps
  - 256^3 simulated spaced
  - Single-precision floating point numbers

- Full discussion available in:
  - Holmen, J.K. and Foster, D.L. 2013. Accelerating Single Iteration Performance of CUDA-Based 3D Reaction-Diffusion Simulations, In *International Journal of Parallel Programming,* 10.1007/s10766-013-0251-z



"Quadrant-Load" Halo-Region Loading Technique