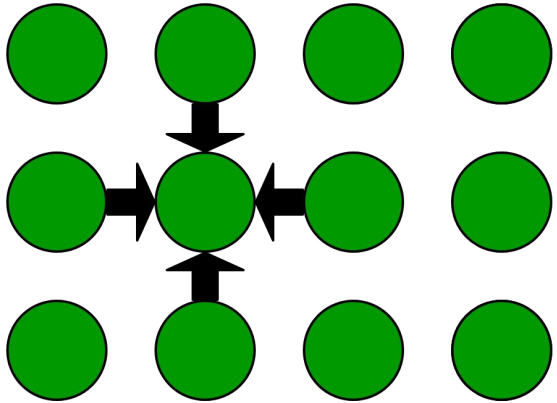
Live Demo

Solving Poisson's equation

SURF Open Innovation Lab and University of Amsterdam
In partnership with NIKHEF

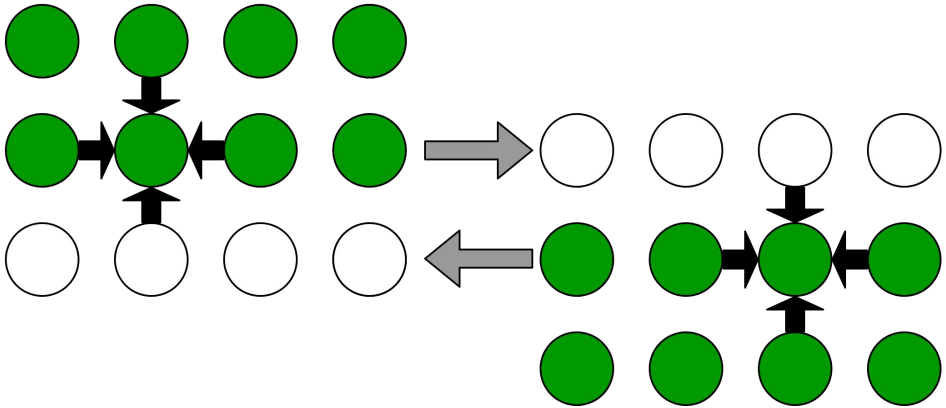
Jacobi solver

- Solve Poisson equation.
- Fixed boundary condition.
- Average of direct neighbors.



The code

- CUDA code written by Nvidia[1].
- 2D domain decomposition.
- MPI for communication between GPU's.
- Halo rows exchanged after each iteration



[1] https://devblogs.nvidia.com/benchmarking-cuda-aware-mpi/

The code: Jacobi kernel

```
int memIdx = (idx.y + 1) * (stride + 2) + idx.x + 1;

// Each thread computes its new value based on the neighbors' old values
real newVal = ((real)0.25) * (oldBlock[memIdx-1] +
                              oldBlock[memIdx+1] +
                              oldBlock[memIdx-stride-2] +
                              oldBlock[memIdx+stride+2]);
newBlock[memIdx] = newVal;

// The global maximum residue must be updated
AtomicMax<real>(devResidue, rabs(newVal - oldBlock[memIdx]));
```

The code: Kernel launch

```
// Launch the kernel for the Jacobi iteration
cudaMemcpy(devResidue, &residue, sizeof(real), cudaMemcpyHostToDevice);

JacobiComputeKernel<<<gridSize, blockSize>>>(devBlocks[0],
                      devBlocks[1], *bounds, size->x, devResidue);

cudaMemcpy(&residue, devResidue, sizeof(real), cudaMemcpyDeviceToHost)
```

Demo

- Explore CUDA code
- Hipify code
  - Hipconvertinplace-perl
  - Hipconvertinplace Clang
  - Compare difference
- Explore HIP code
- Run code
- Briefly look at performance

Some numbers

- 1000 iterations
- Each GPU processes 4096 x 4096 cells

	# GPU's	GFLOPS / GPU
TitanX	1	66.47
MI50	1	42.70
MI50 same island	2	39.70
MI50 different island	2	27.60
MI50 same island	4	38.44

Conclusion

- The tools are straightforward to use ...
- But, we can’t blindly trust the tools.
- Understanding performance might require more in depth analysis.