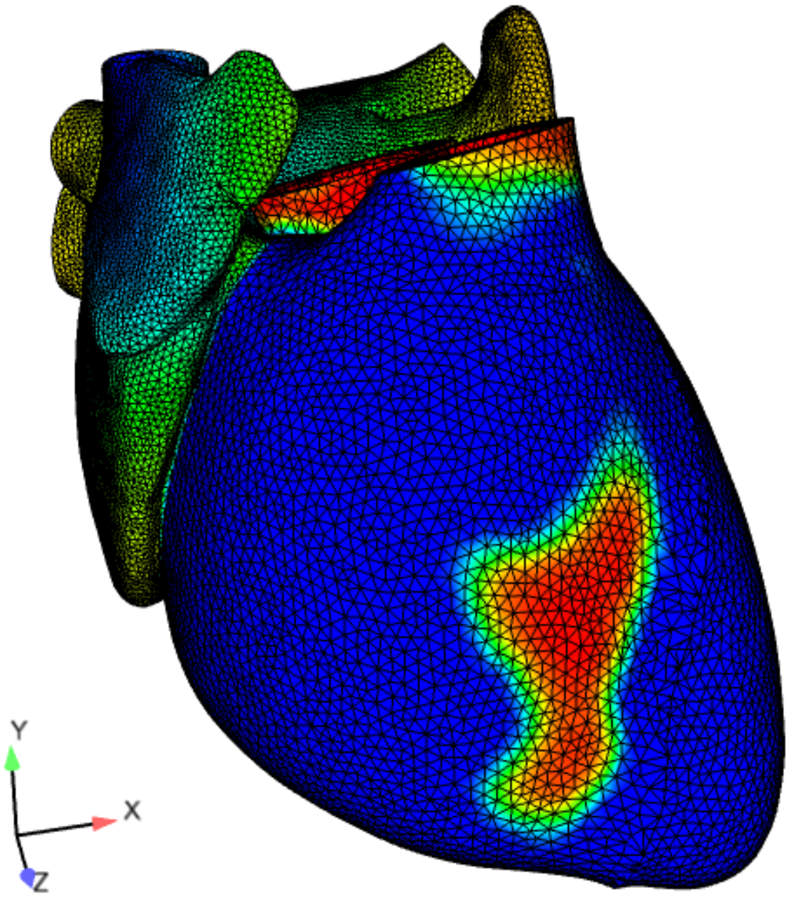
Porting Tissue-scale Cardiac Simulations to the Knights Landing Platform

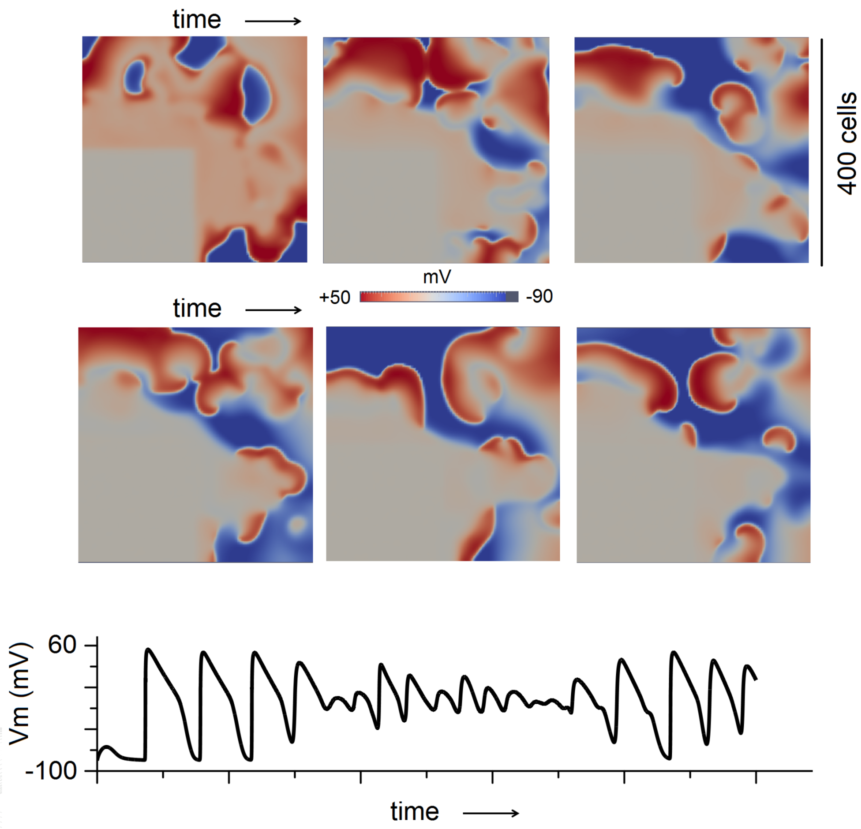
Johannes Langguth

Simula Research Laboratory, Oslo, Norway

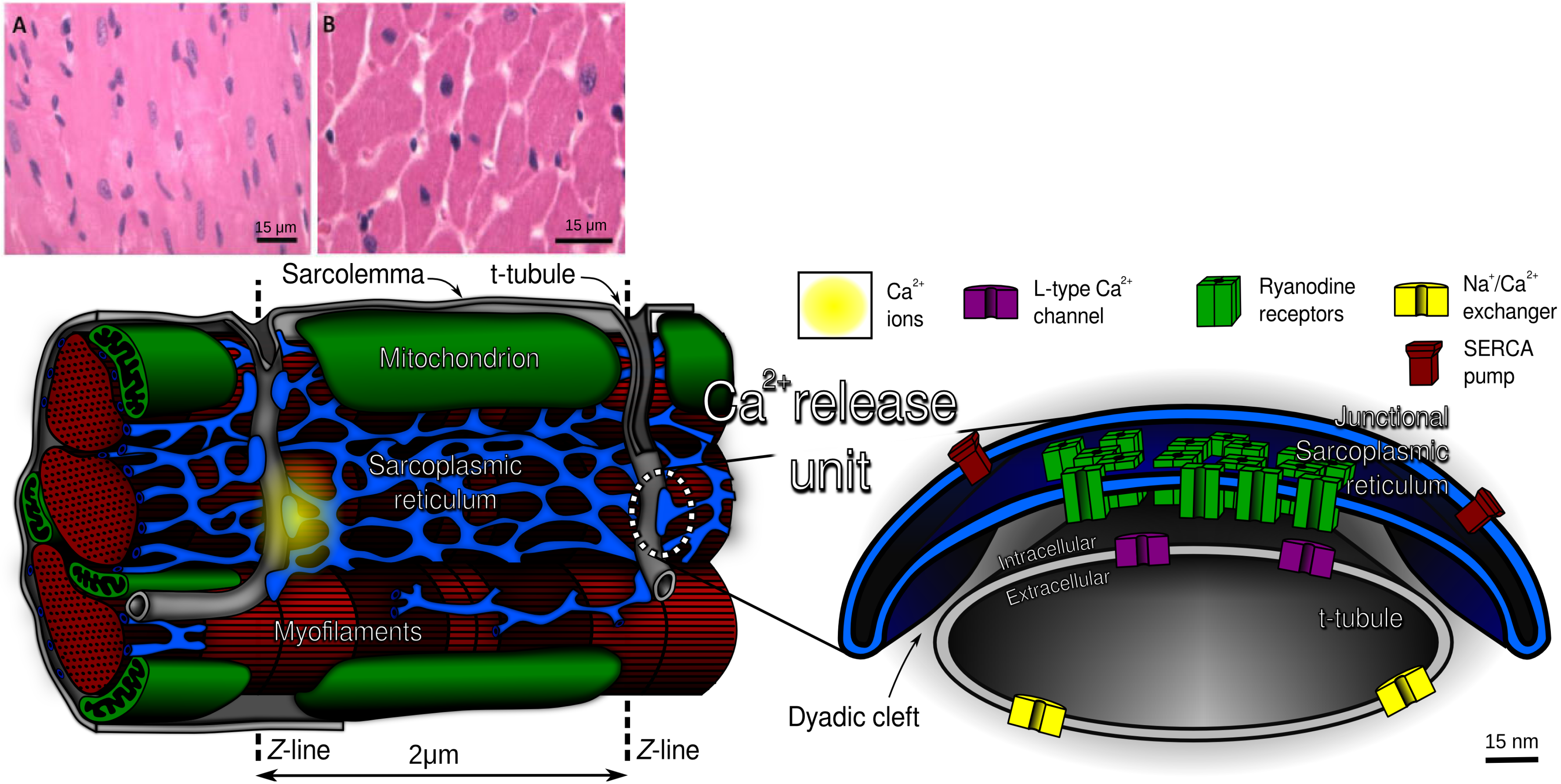
joint work with
Chad Jarvis and Xing Cai

Cardiac Electrophysiology

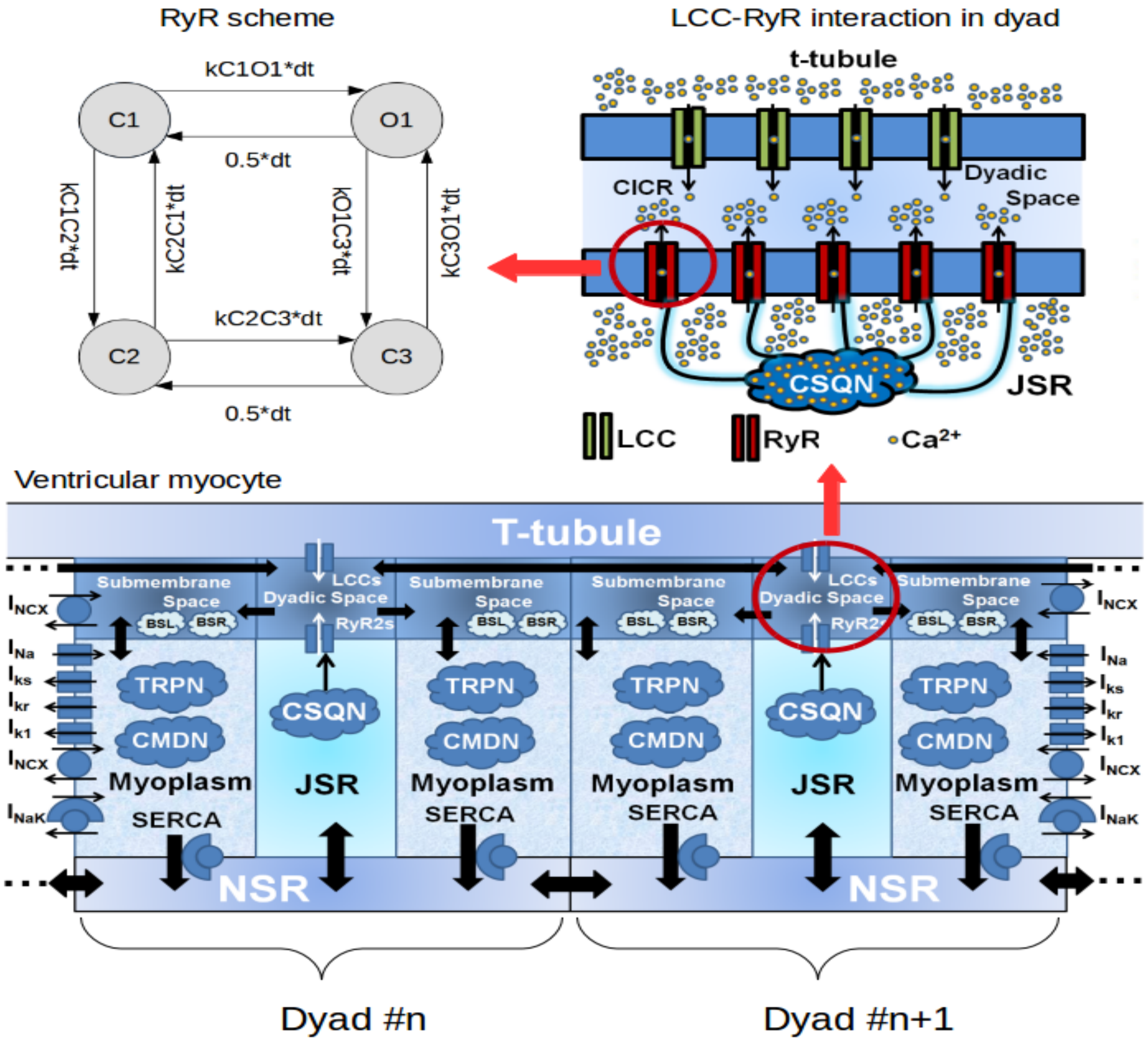




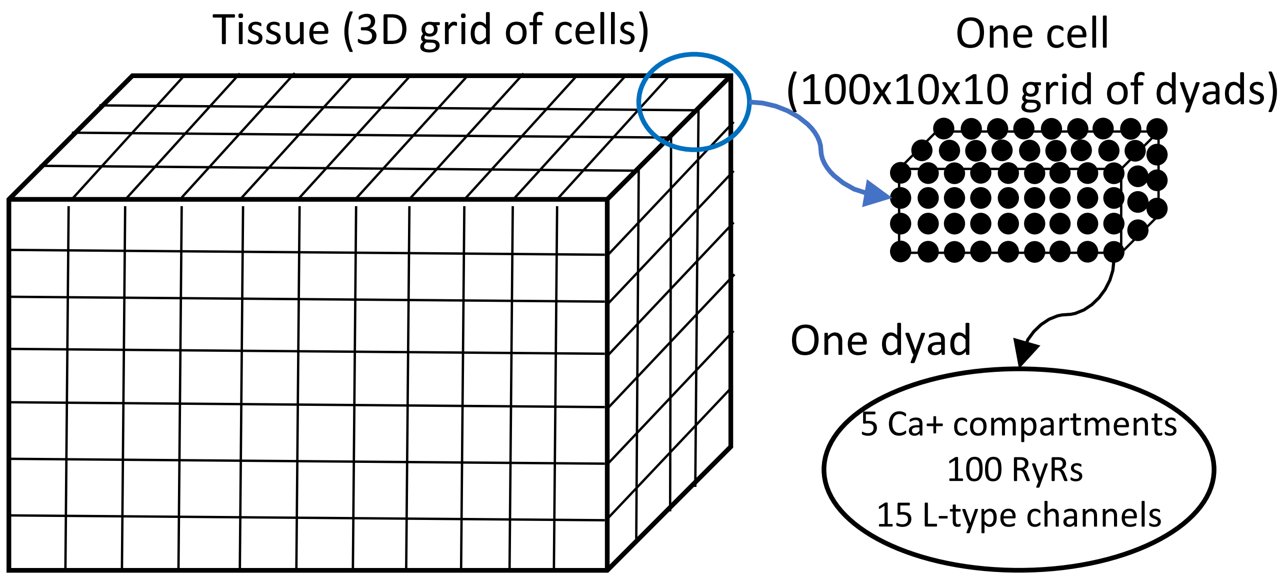
Calcium Handling in the Human Cardiac Ventricle



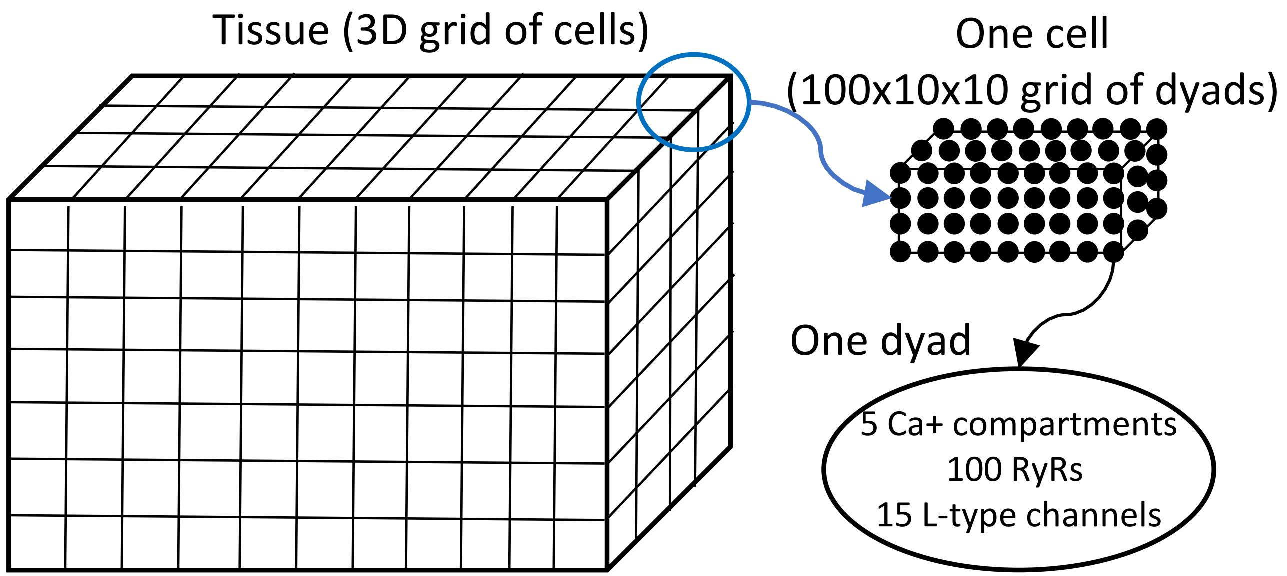
Ryanodine Receptors



Computational Scope



Computational Scope



- $2 * 10^9$ Cells in the heart
- 10^4 Dyads per cell
- 10^2 Ryanodine Receptors (RyRs) per dyad
- 10^4 Time steps per heartbeat

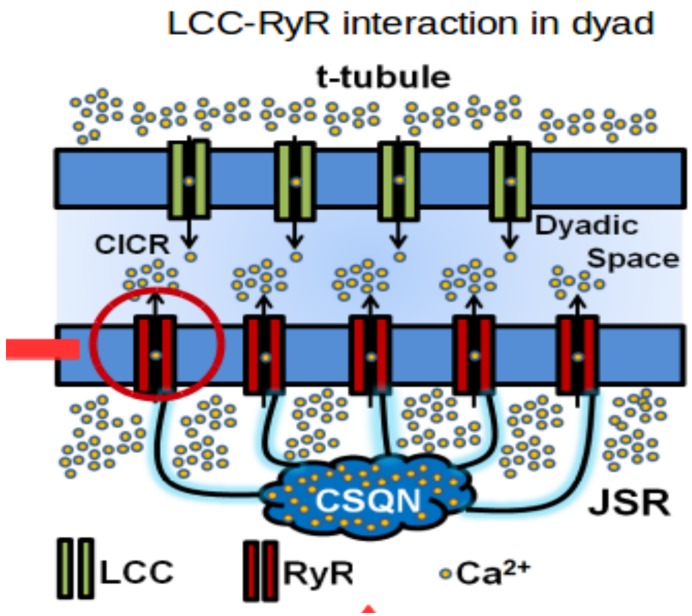
10^{19} possible state transitions

Computation Timestep

For each Cell do (OpenMP parallel for)

1. Compute L-type opening probabilities
2. Simulate L-type opening
3. Compute RyR opening probabilities
4. Simulate RyR opening
5. Compute calcium concentrations
6. Dyad diffusion

Cell diffusion (Local & MPI)



Computation Timestep

For each Cell do (OpenMP parallel for)

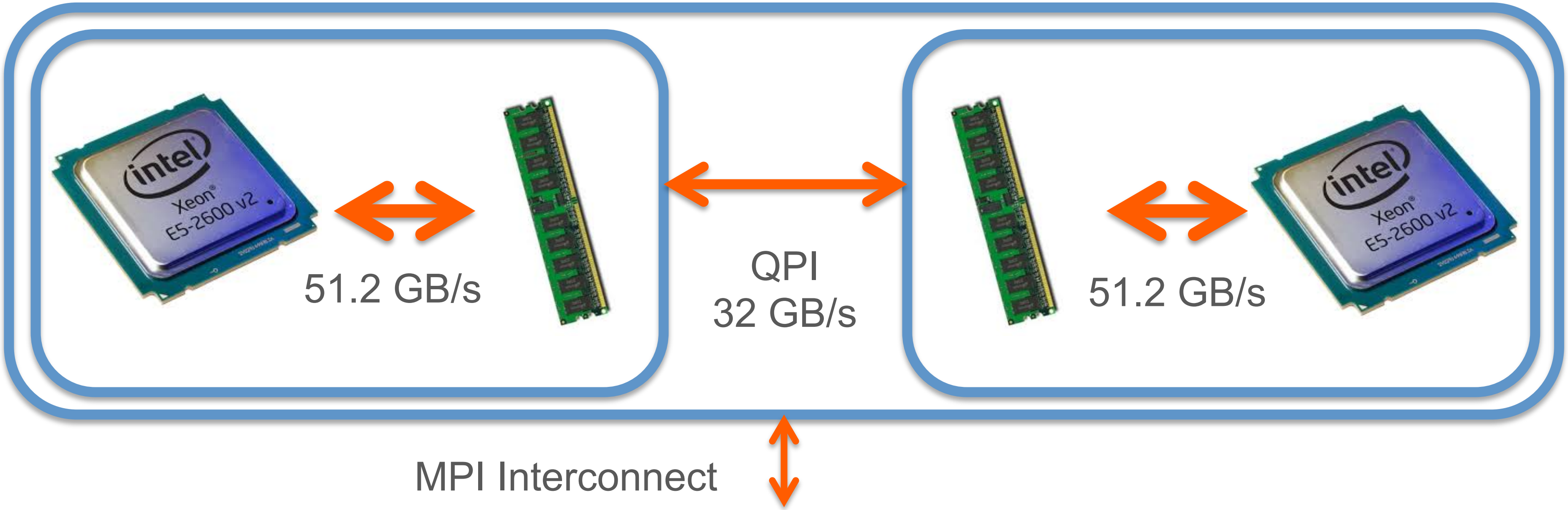
1. Compute L-type opening probabilities
2. Simulate L-type opening
3. Compute RyR opening probabilities
4. Simulate RyR opening
5. Compute calcium concentrations
6. Dyad diffusion

Cell diffusion (Local & MPI)

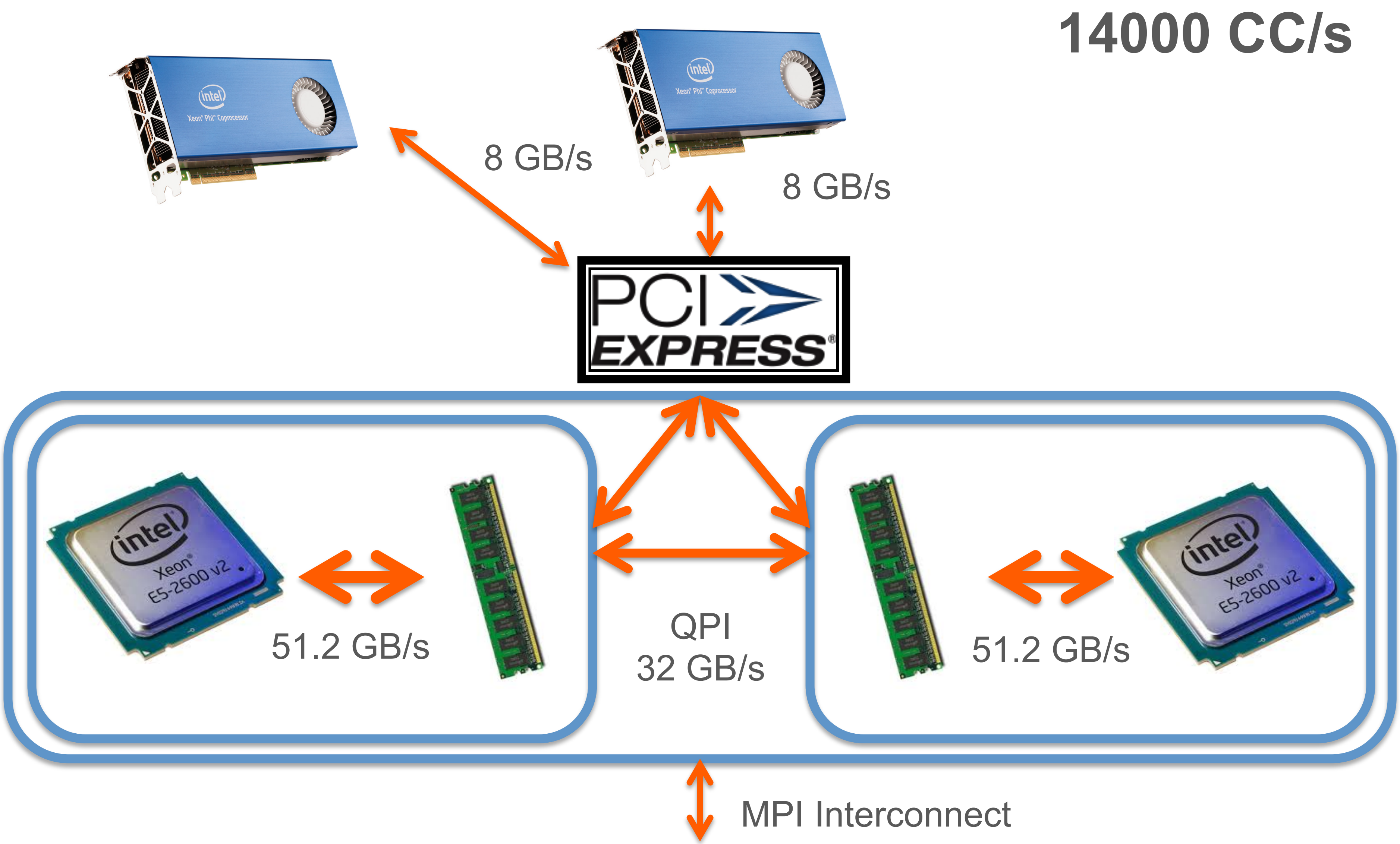
Relevant metric: Cell computations/s (CC/s)

Basic Configuration: Dual Sandy Bridge

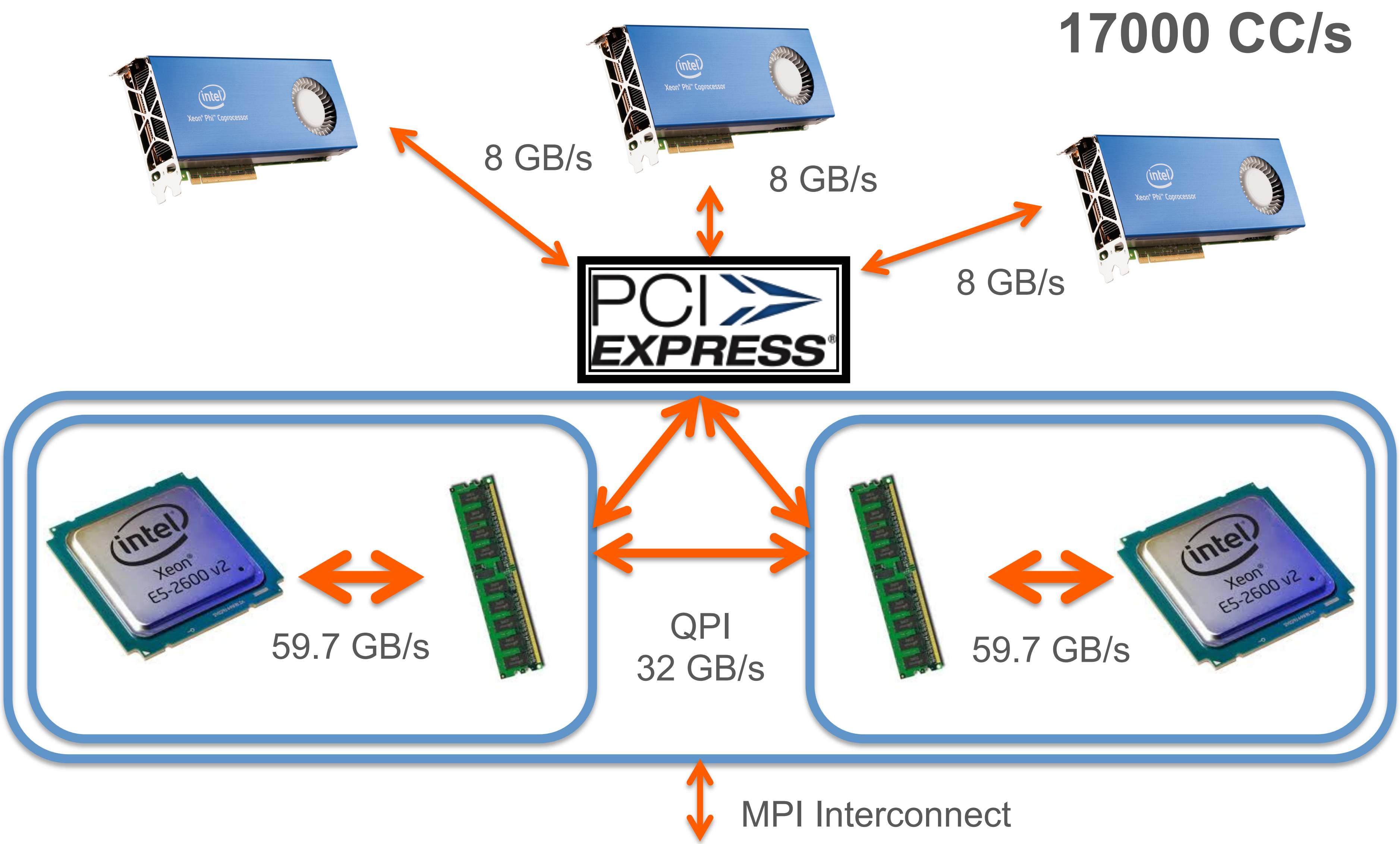
4000 CC/s
(Cell computations per second)



Node Structure on Abel



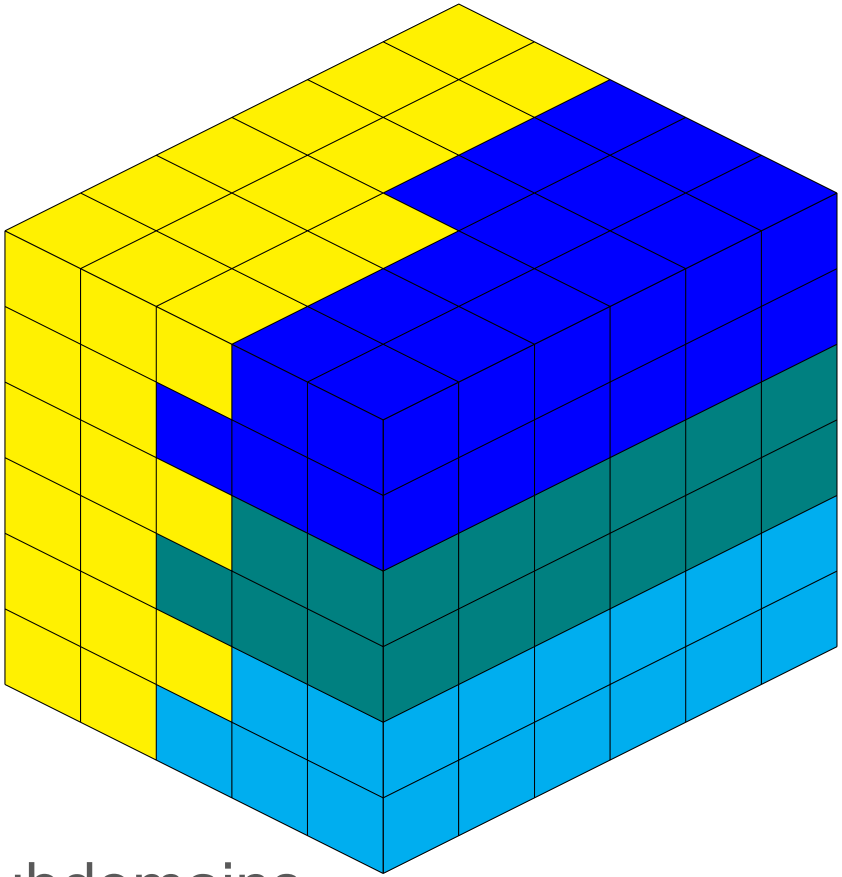
Node Structure on TianHe-2



Computation on TianHe-2

- Largest grid so far: 3.2 million (400 nodes)
- Problem: heterogeneous load balancing and memory
- Need about 3 MB per cell -> about 2000 cells per Phi max
- Find feasible and balanced allocation to avoid idling

Domain Decomposition



- Cuboid node subdomains
- Unstructured intra-node subdomains
- Use 20 x 20 x 20 node subdomains
- 93% CPU load

Xeon Phi 7250 Server

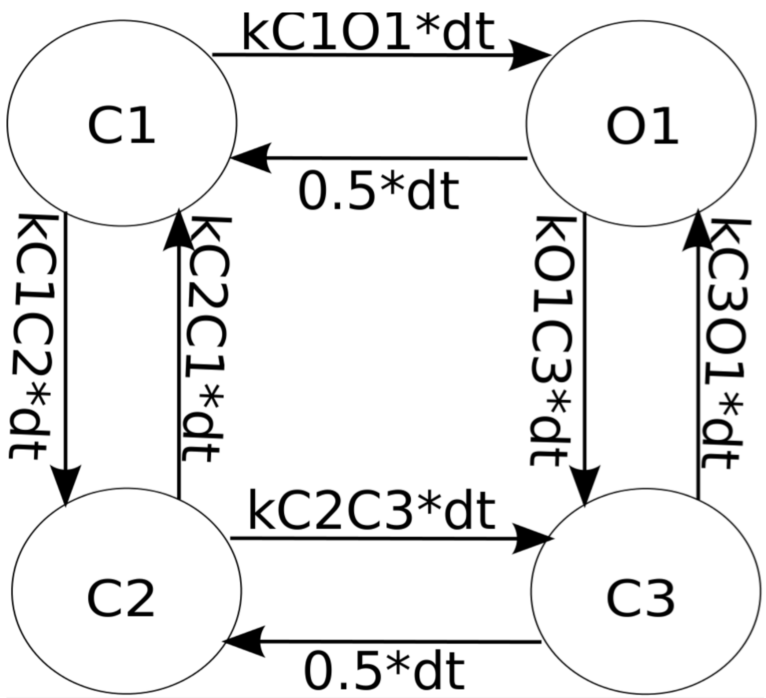
14000 CC/s
"Out of the box"

- 16 GB MCDRAM
- 96 GB DDR4
- Homogeneous

- 5000   Cells in HBM
- 34000 Cells in DDR4

State Transitions

- Stochastic state transitions for each ryanodine receptor
- 10^6 possible state transitions per cell and time step
- Bernoulli trials (coin toss) cost too many random numbers
- Binomial distributions are efficient but irregular
- Difficult vectorization due to variable number of RyRs per state

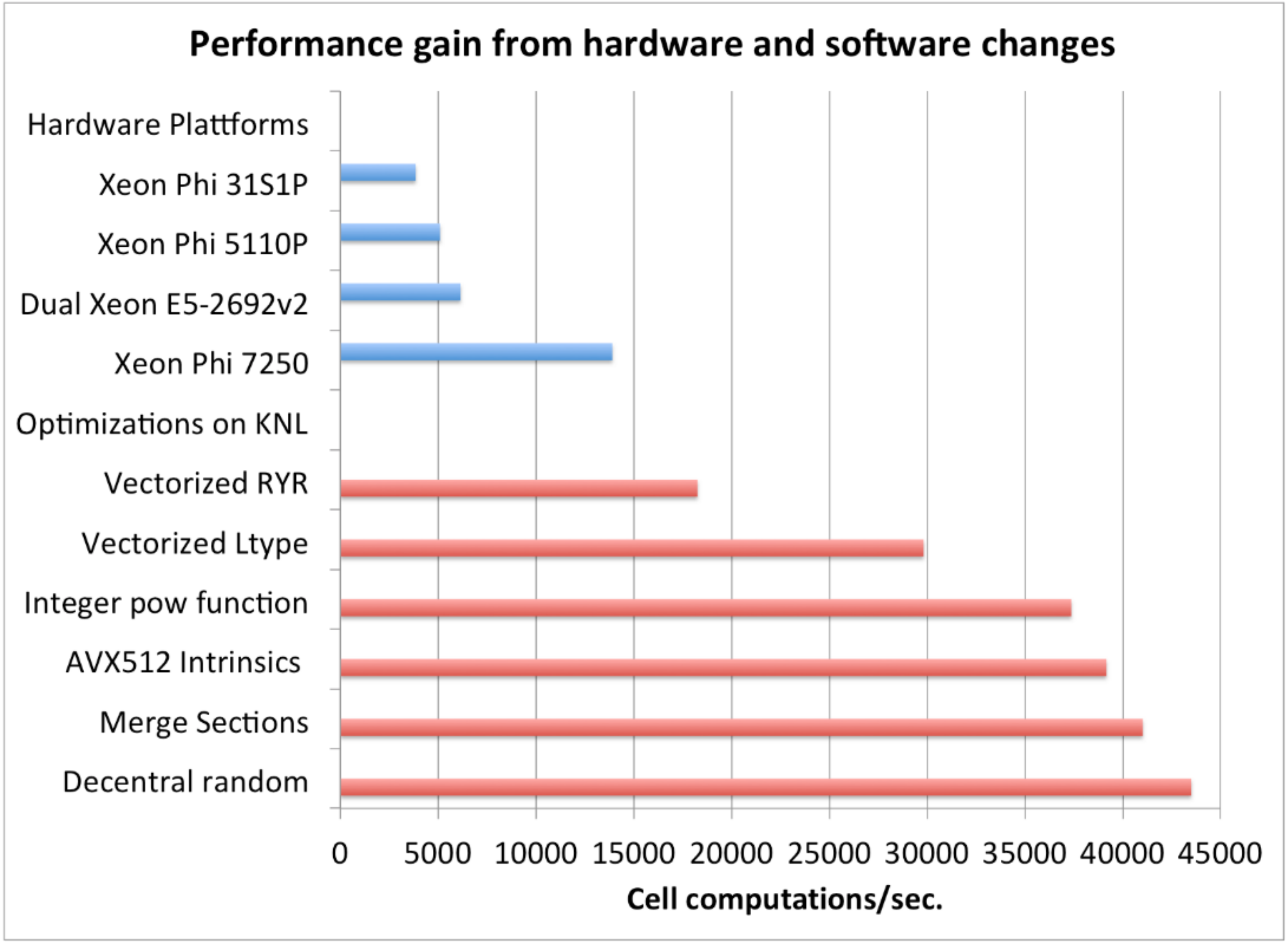


Vectorized Binomial Distribution Sampling

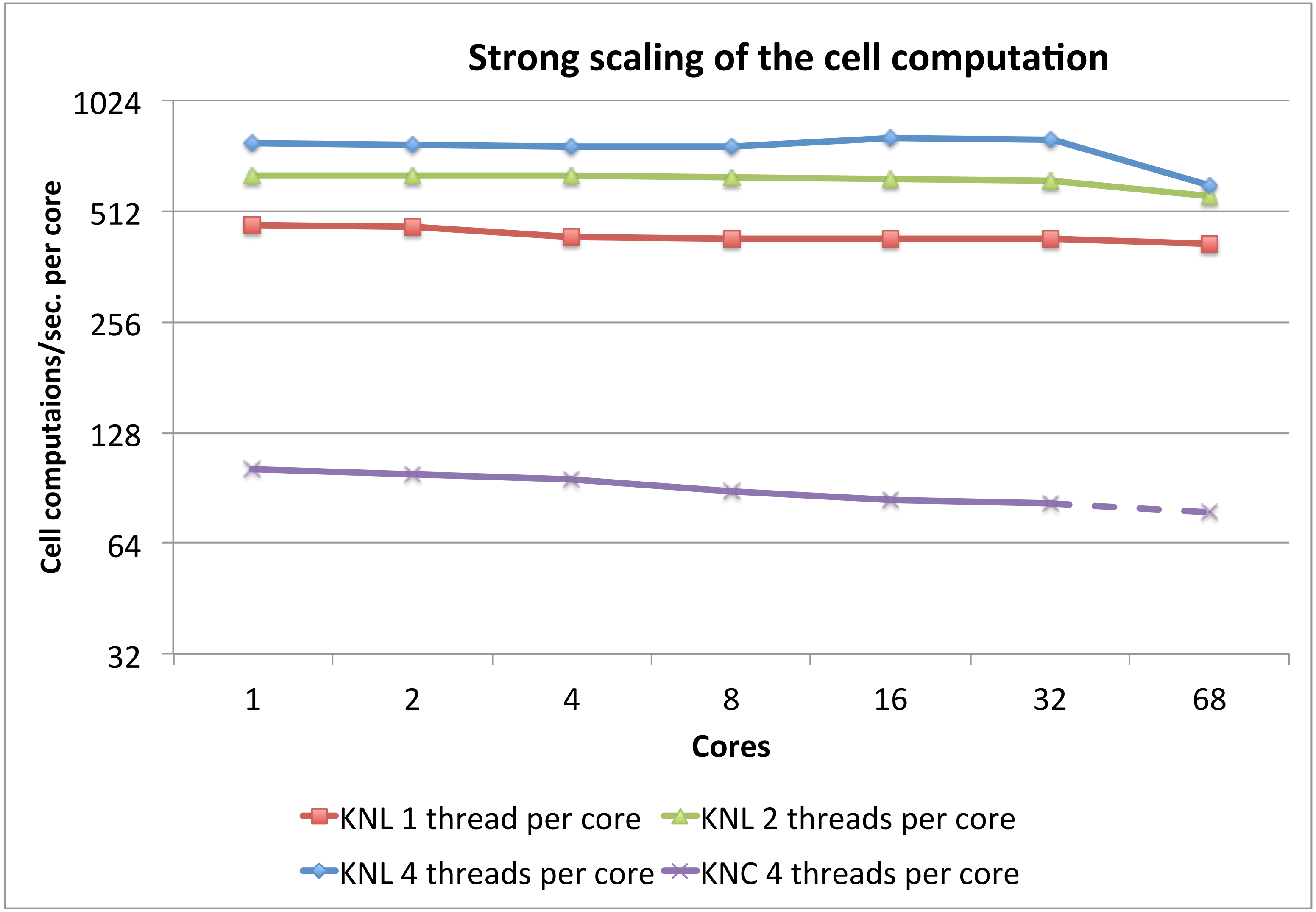
$$F(k;n,p) = \Pr(X \le k) = \sum_{i=0}^{\lfloor k \rfloor} \binom{n}{i} p^i (1-p)^{n-i}$$

```
function VectorizedBinomial {
    Input: Vectors N, P, RANDVAL
    Output: Vector K
    Initialize K = 0
    Initialize 1P = Vector_subtract(1,P);
    Initialize PKNK = Vector_power(1P,N);
    Initialize P1P = Vector_divide(P,1P);
    for (int i = 0; i < max(N); i++) {
        BC = Vector_gather(BC_table,N,K);
        SUB = Vector_multiply(BC,PKNK);
        RANDVAL = Vector_subtract(RANDVAL,SUB);
        PKNK = Vector_multiply(P1P,PKNK);
        MASK = Vector_mask_compare(MASK,RANDVAL > 0);
        K = Vector_mask_add(K,1,MASK);
    }
}
```

Performance Improvement



Strong Scaling



Hyperthreading matters, bandwidth limited

Memory Strategies

1. **Cache Mode**
   - Easiest to use
   - Loss of memory

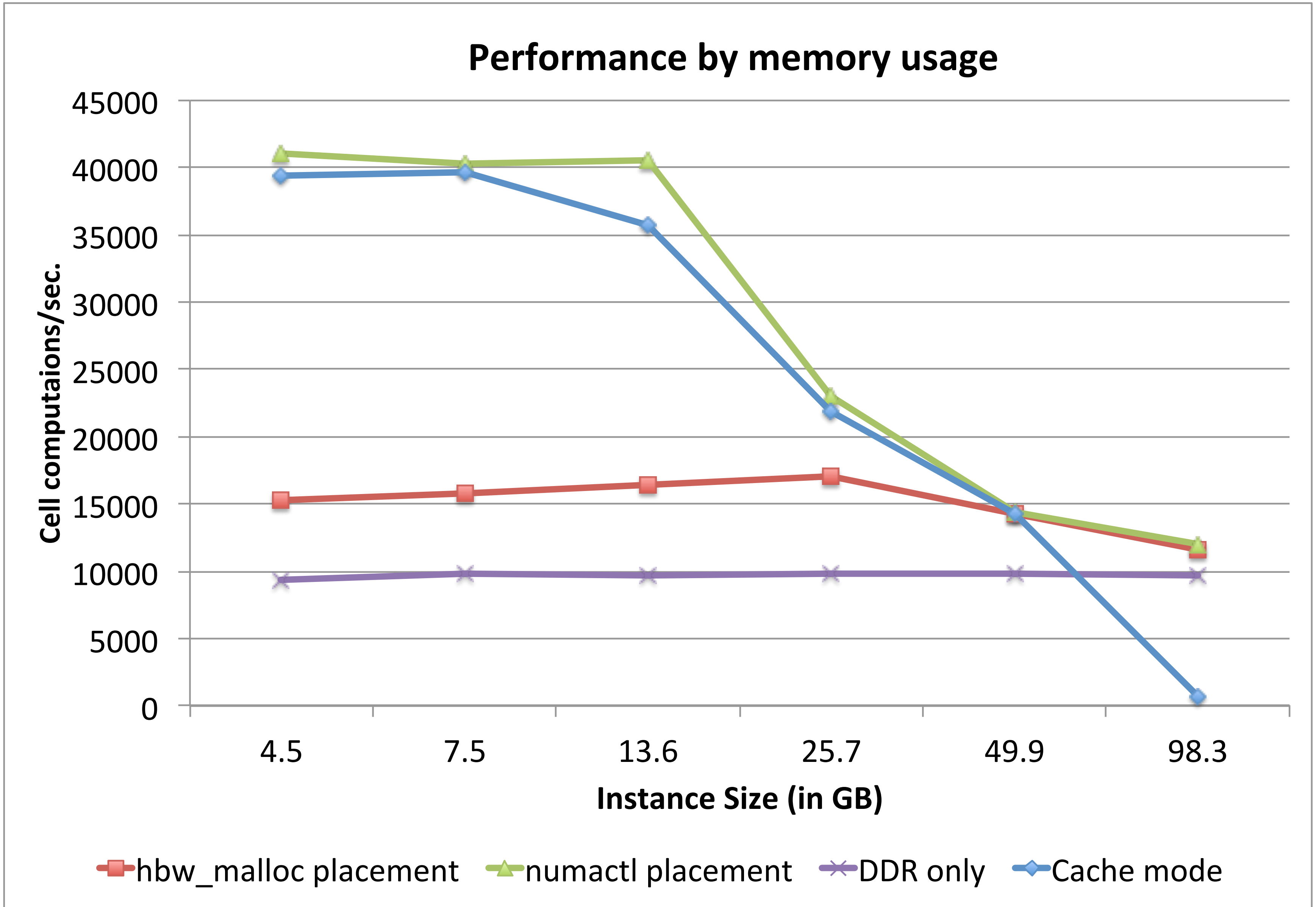
Memory Strategies

1. **Cache Mode**
   - Easiest to use
   - Loss of memory
2. **Flat mode with** *hbwalloc*
   - Maximum control
   - Size specific

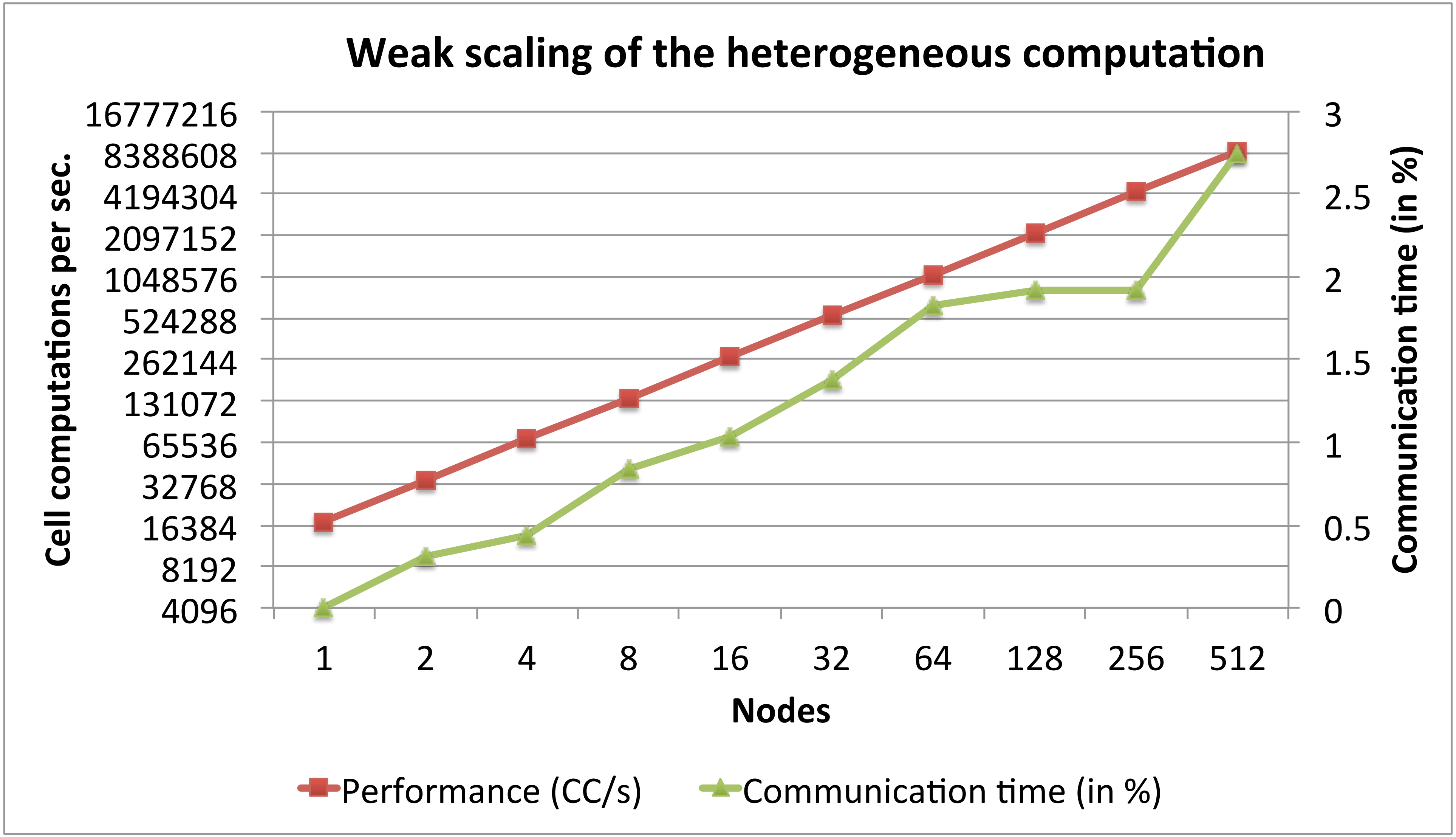
Memory Strategies

1. **Cache Mode**
   - Easiest to use
   - Loss of memory
2. **Flat mode with** *hbwalloc*
   - Maximum control
   - Size specific
3. **Flat mode with ordering and** *numactl –preferred=1*
   - Easy and portable
   - Software development hazard

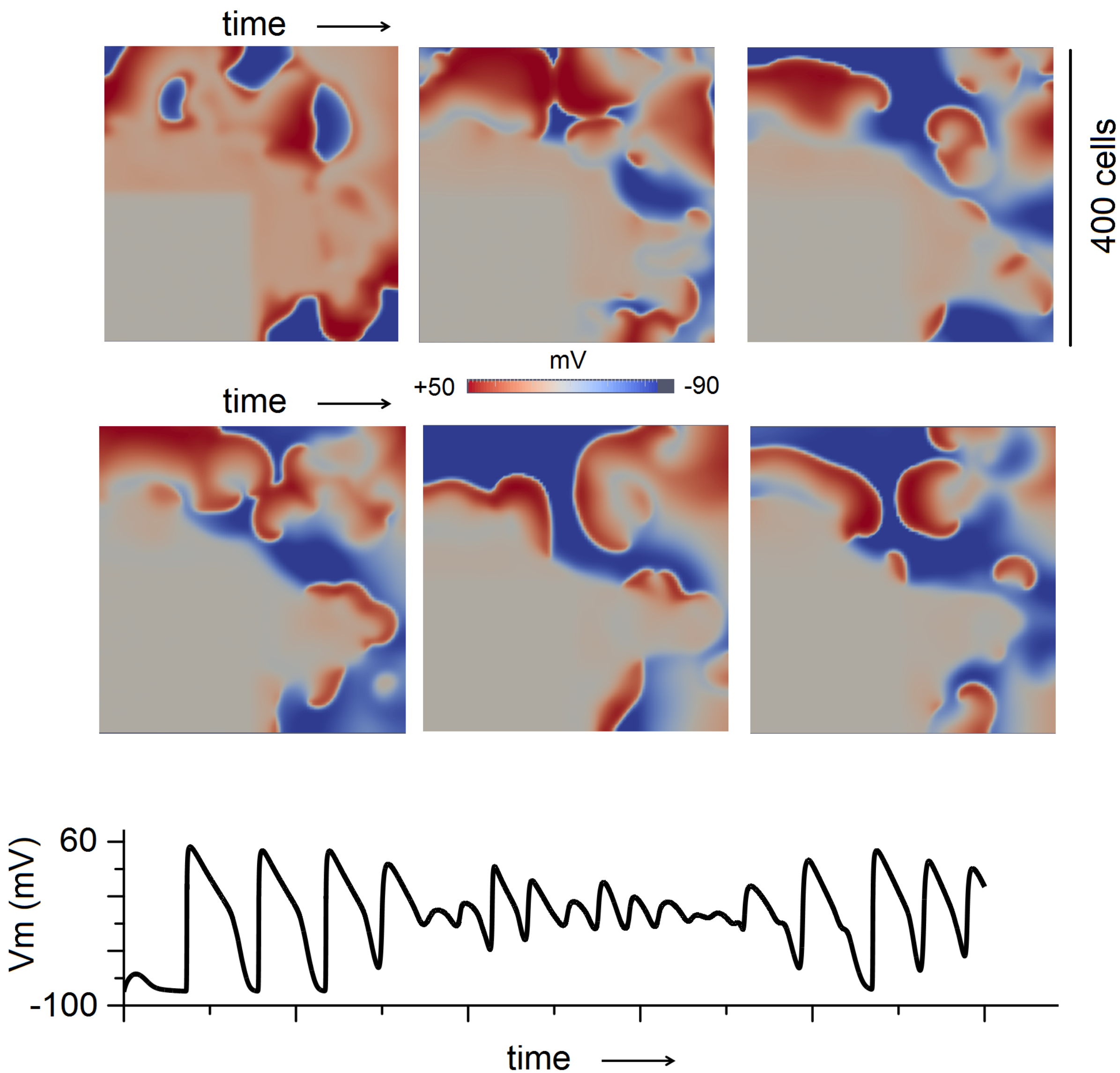
Performance is Memory Dependent



Scaling is not a Problem (on TH-2)



Simulation of Unhealthy Tissue



Observations

- Difficult computation
- Easy communication
- Organ scale requires very large machines

Observations

- Difficult computation
- Easy communication
- Organ scale requires very large machines

- KNL has decent performance “for free” (3x over KNC)
- Intrinsics enable full performance (another 3x)
- Memory limited and bandwidth bound

Observations

- Difficult computation
- Easy communication
- Organ scale requires very large machines

- KNL has decent performance “for free” (3x over KNC)
- Intrinsics enable full performance (another 3x)
- Memory limited and bandwidth bound

- K20X GPU delivers ½ KNL CC/s @ ½ KNL GB/s
- P100 and V100 could be much faster

PHI vs GPU

KNL advantages:

- No heterogeneous programming
- Smaller penalty for exceeding 16 GB
- Could use MCDRAM <-> DDR swapping
- Low reuse: requires temporal blocking to be effective

Questions ?