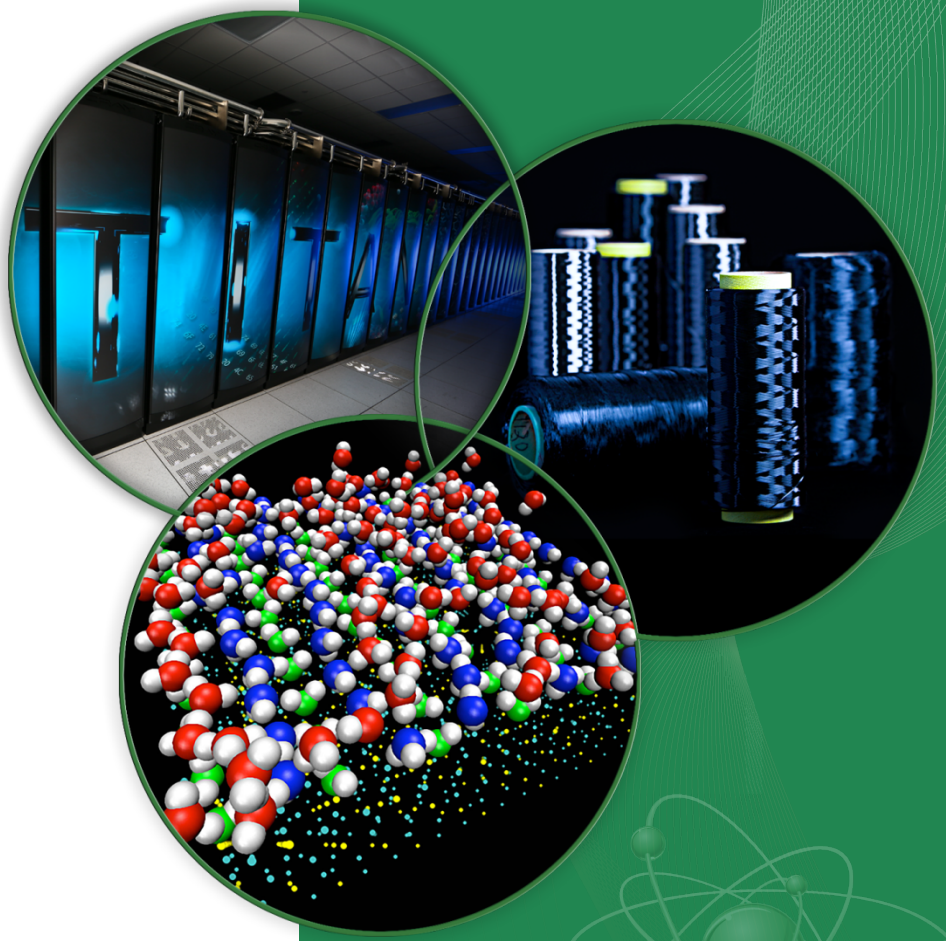
What is Parallelism?

Robert D. French

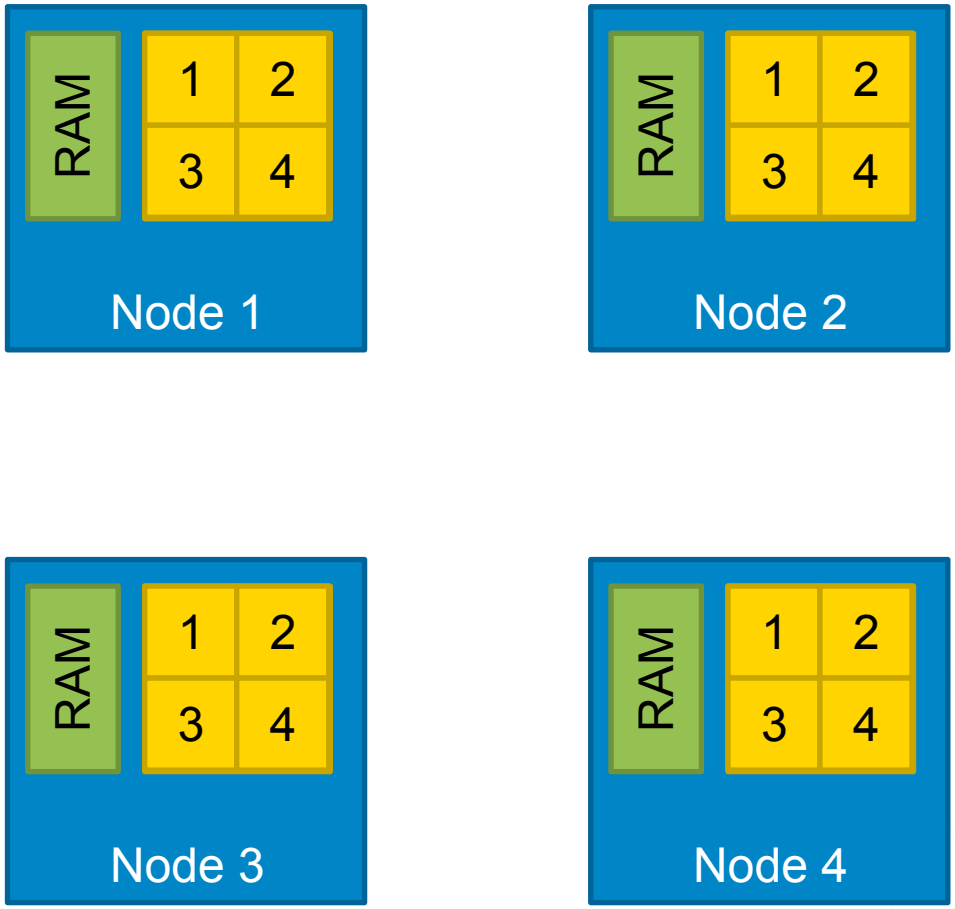
NCCS User Assistance



What is Parallelism

- Using multiple computers (or cores) to work on a single problem

What is Parallelism



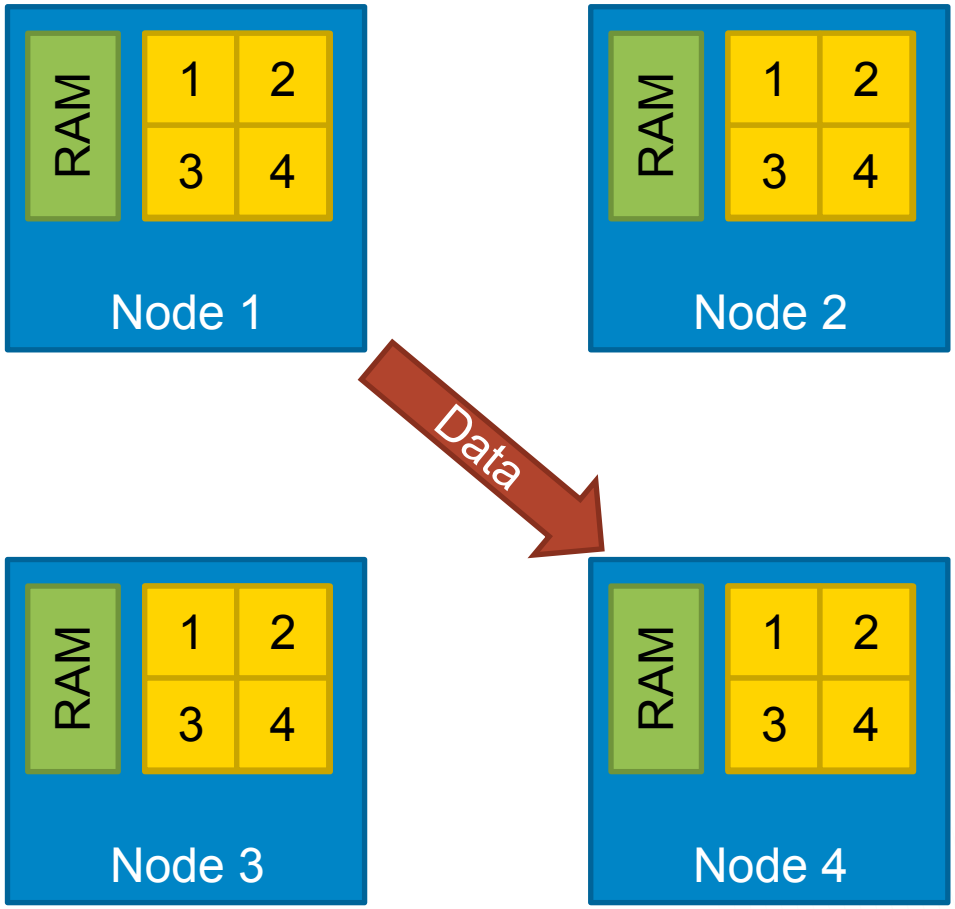
What is Parallelism

- Using multiple computers (or cores) to work on a single problem
- What kinds of problems actually need more than one computer?

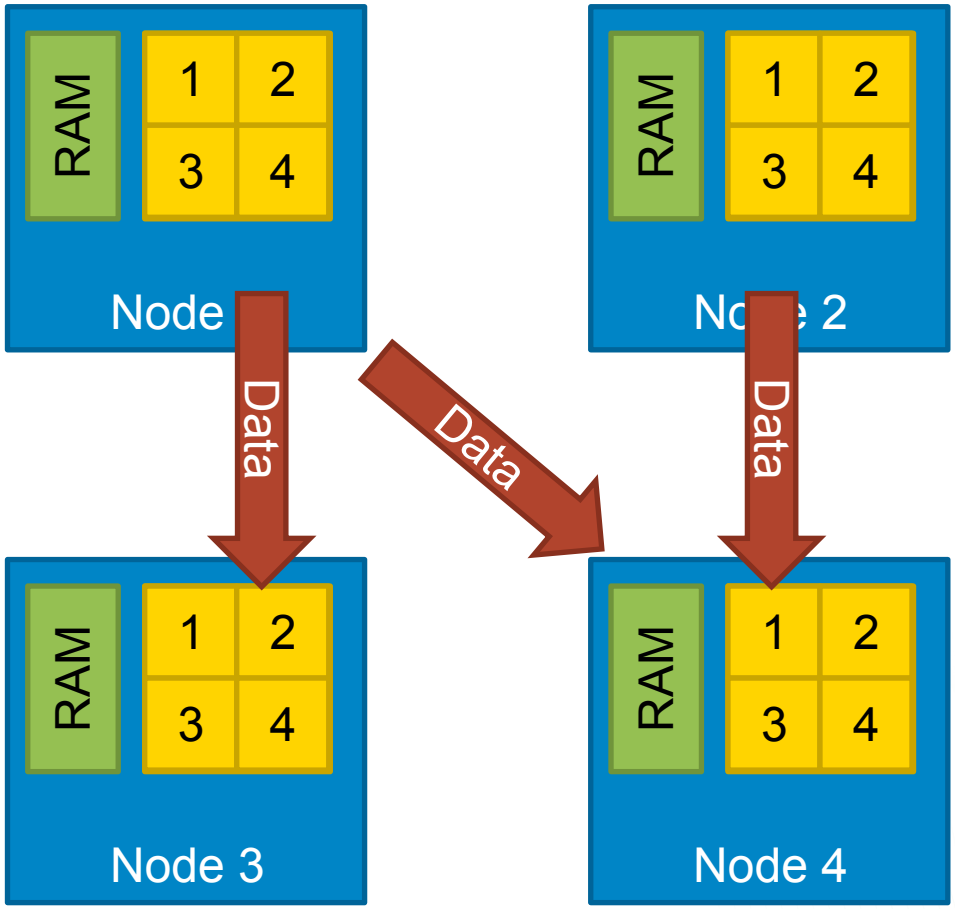
What is Parallelism

- Using multiple computers (or cores) to work on a single problem
- What kinds of problems actually need more than one computer?
- What does it mean for computers to collaborate on the same problem?

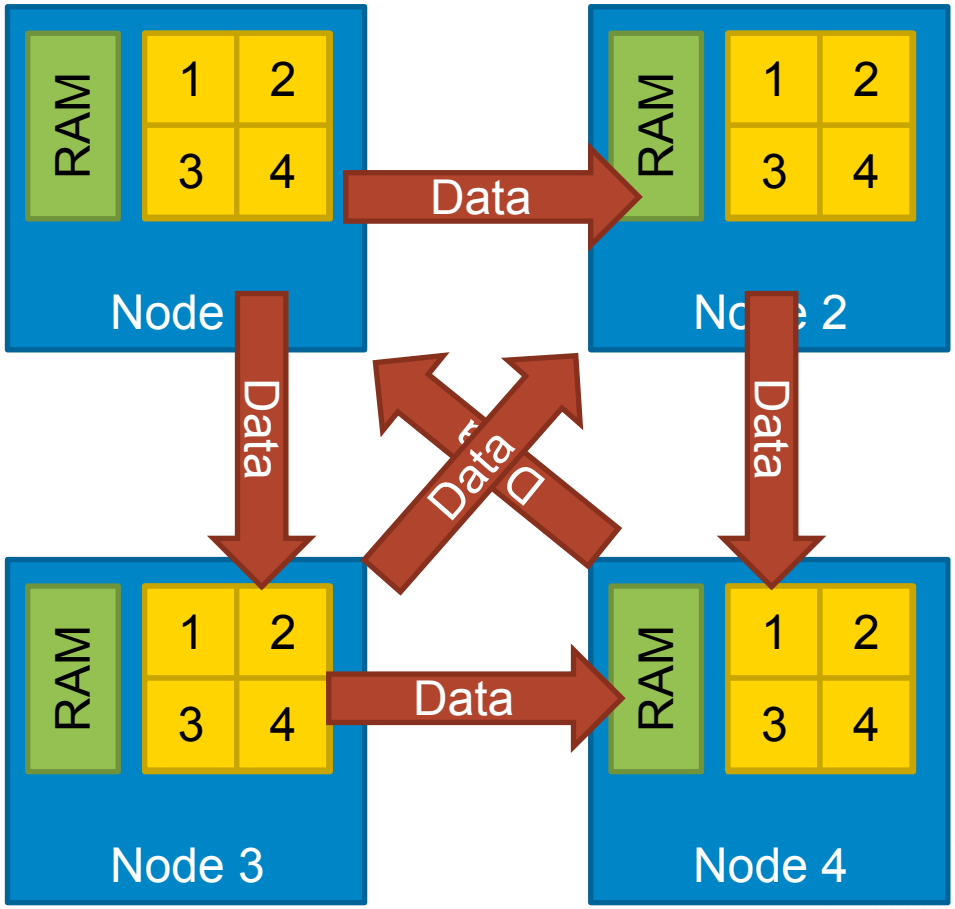
What is Parallelism?



What is Parallelism?



What is Parallelism?



Smoothed Particle Hydrodynamics

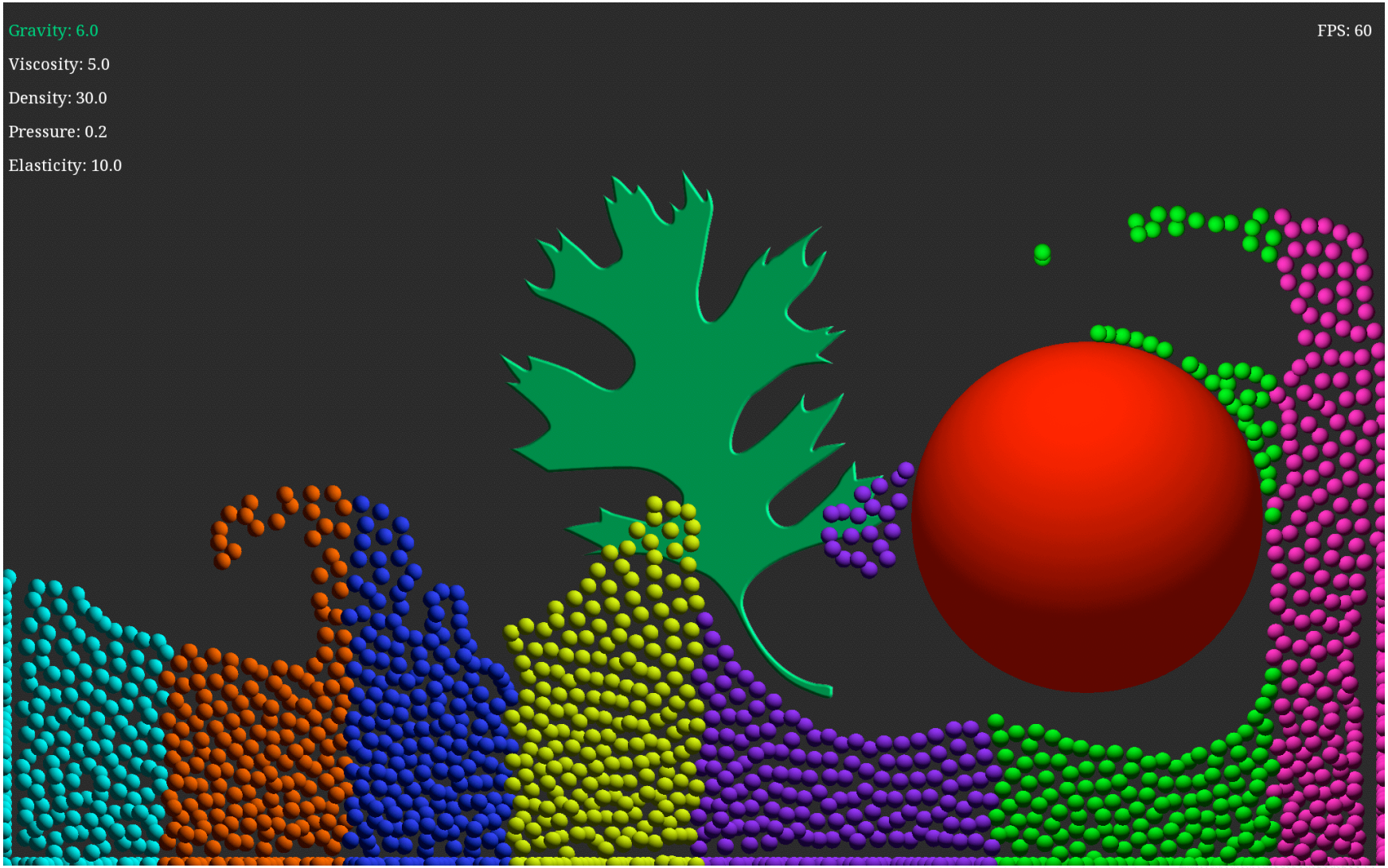
OAK RIDGE National Laboratory

Smoothed Particle Hydrodynamics

- Understanding the motion of liquids
- Can answer the question:

“What does it look like when water sloshes around?”

Smoothed Particle Hydrodynamics



Smoothed Particle Hydrodynamics

- How the simulation works (*The Algorithm*)
  1. Calculate the force between each pair of particles
  2. Total force on each particle determines its *acceleration*
  3. Update particle *position*
  4. Proceed to next *time step*
- Real easy, right?

Smoothed Particle Hydrodynamics

- Say we have One Million Particles

Smoothed Particle Hydrodynamics

- Say we have One Million Particles
- That's one trillion force pairs to calculate.
- *That's a lot of work for one computer*
- This is why we need parallelism

Threads

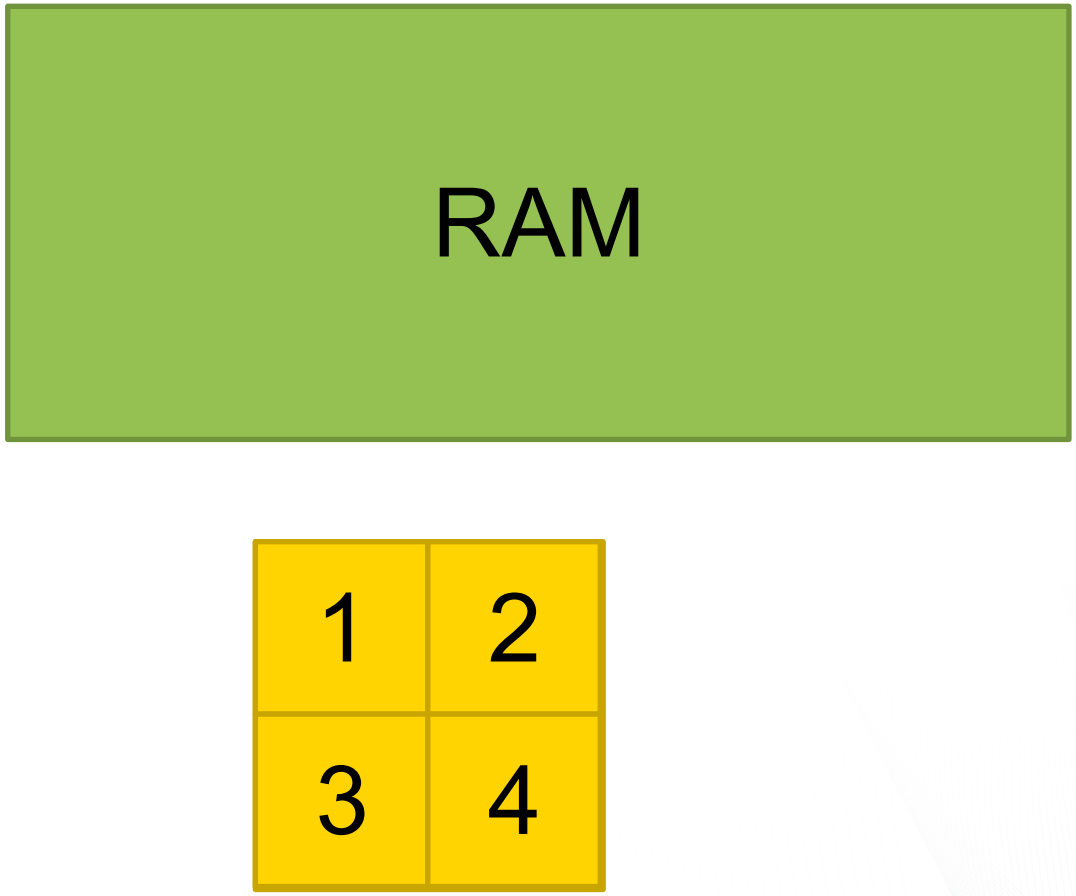
Threads

- Sometimes called “Shared Memory Parallelism”
- Parallelism using multi-core processors
- Can be used to divide work within a program
- Each thread can handle a subset of the force pair calculations

Threads

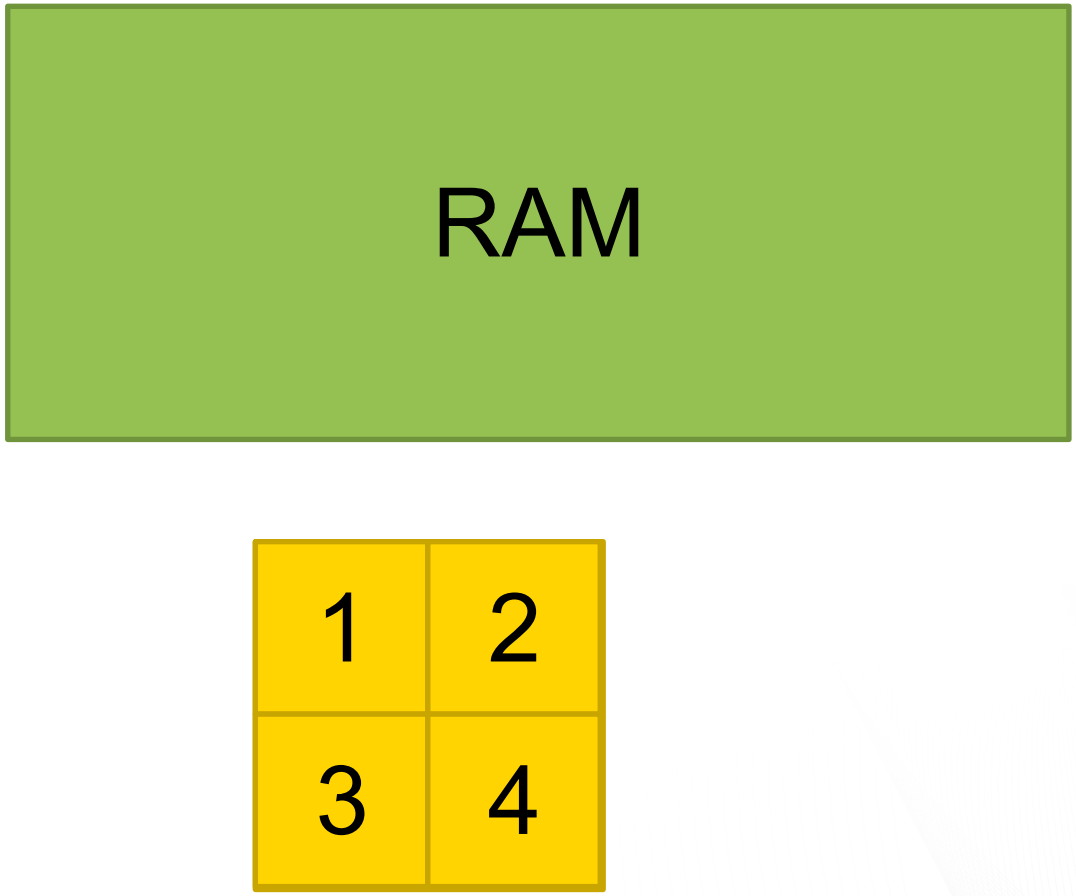
- In our SPH example, more cores means less work for each core
- A quad-core CPU can work about 4x than a single core CPU
- We need *threads* to manage this workload

Threads



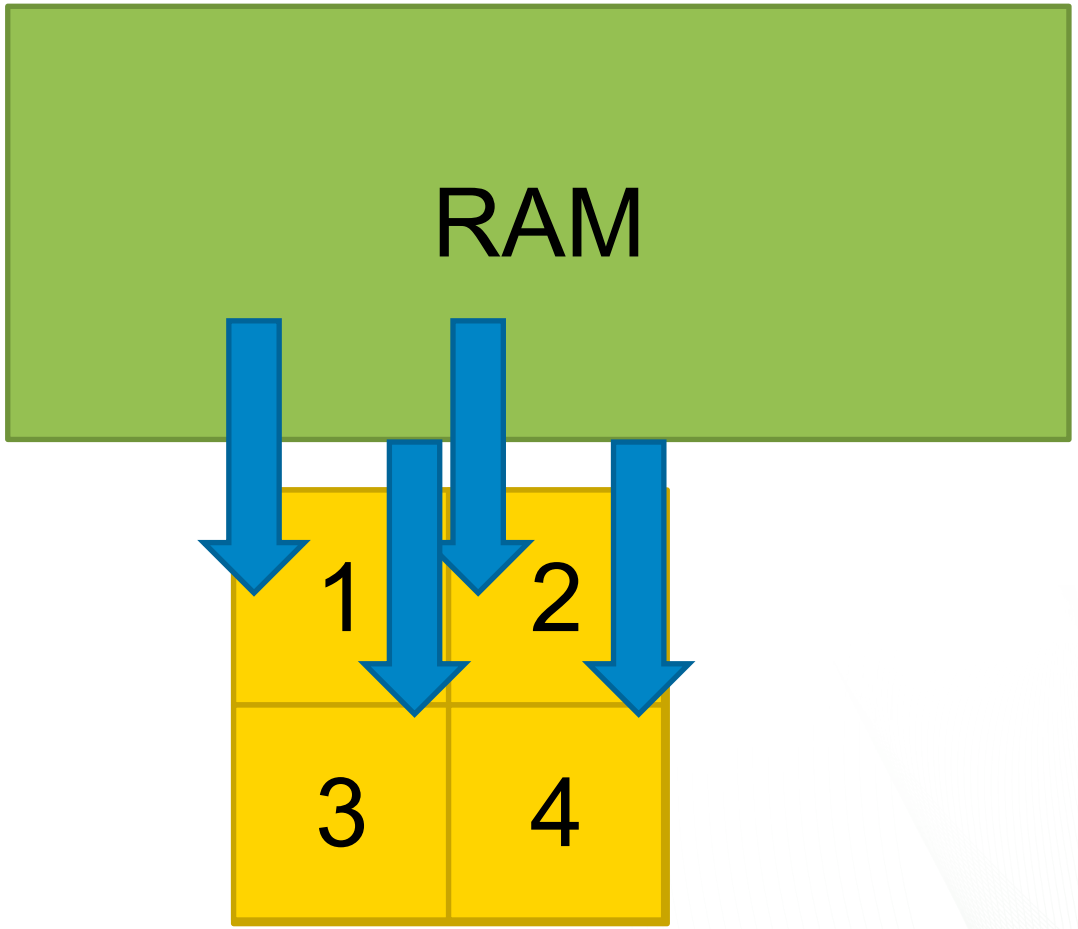
Threads

The particle information stays in RAM



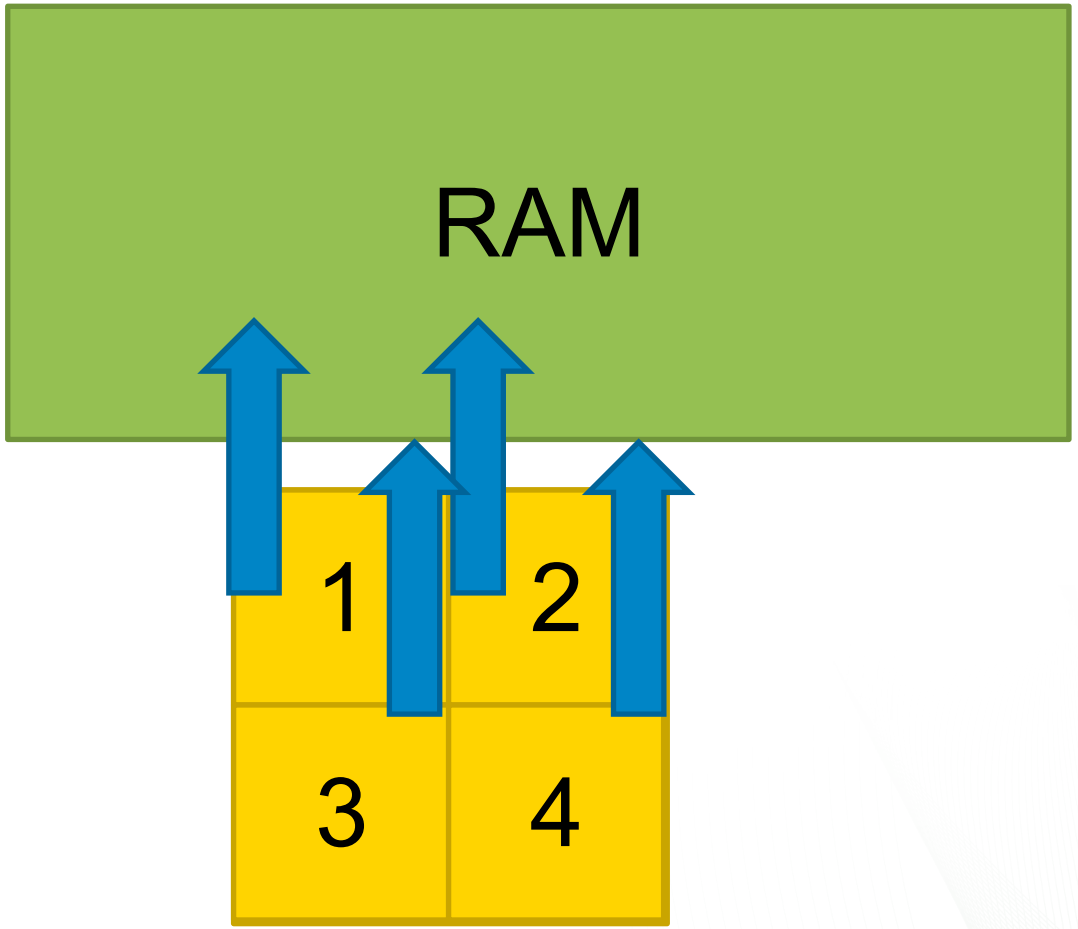
Threads

Particle Info moves to CPUs for Force Calculation



Threads

CPUs send updated positions back to RAM



Threads

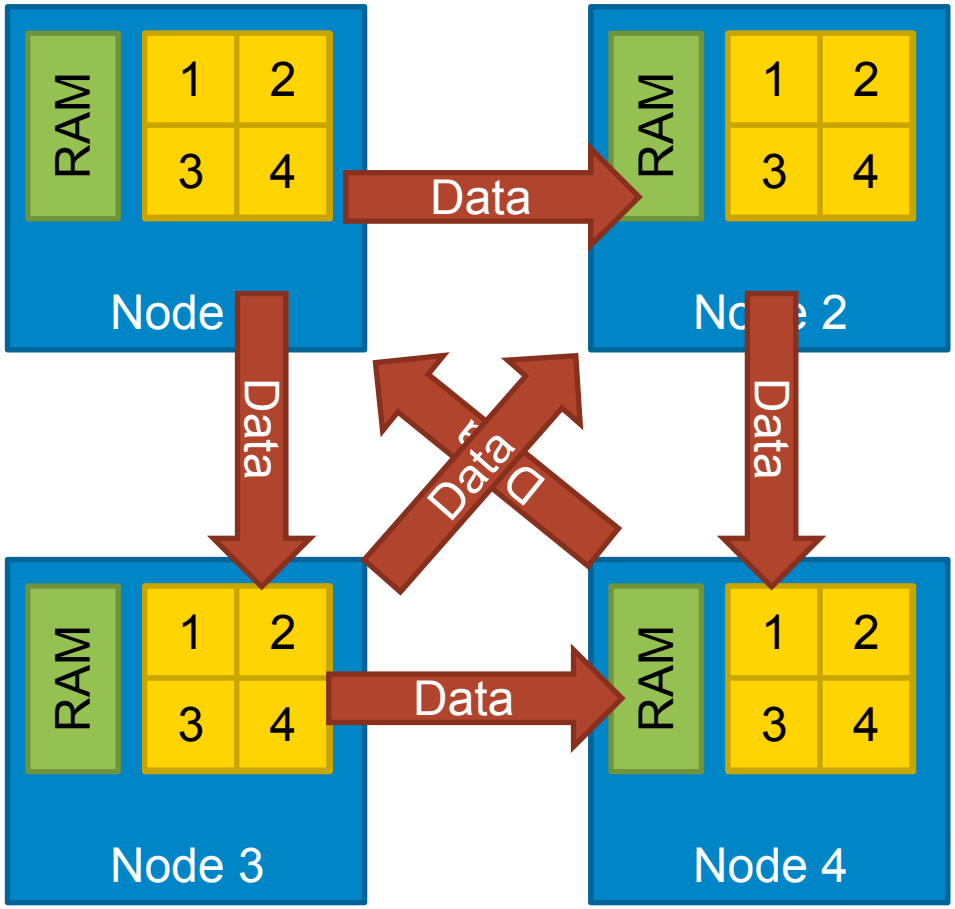
- Even still, 1 Trillion Force Pairs is a lot of work for 4 CPUs
- Is there any way to leverage the CPUs on other computers?

Ranks

Ranks

- Sometimes called “Distributed Memory Parallelism”
- Parallelism *between* nodes
- Makes use of multiple computers (Ranks)on a shared network
- Ranks communicate by *passing data* from one computer to another

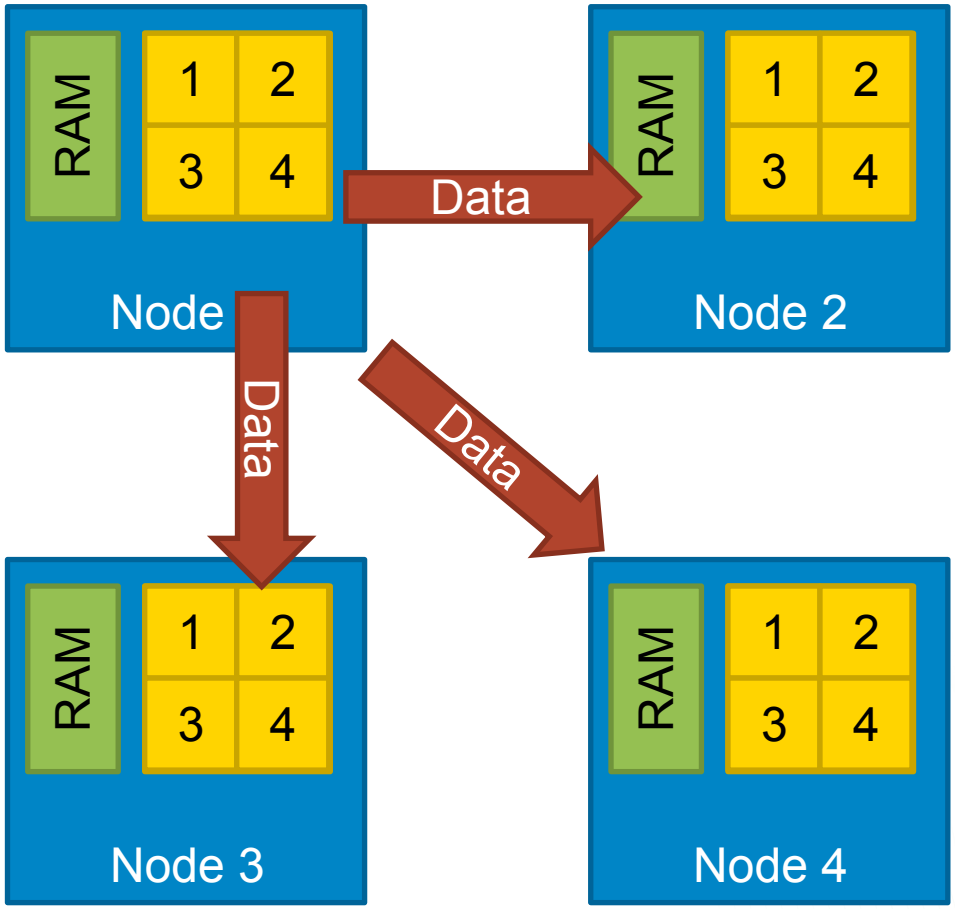
Ranks



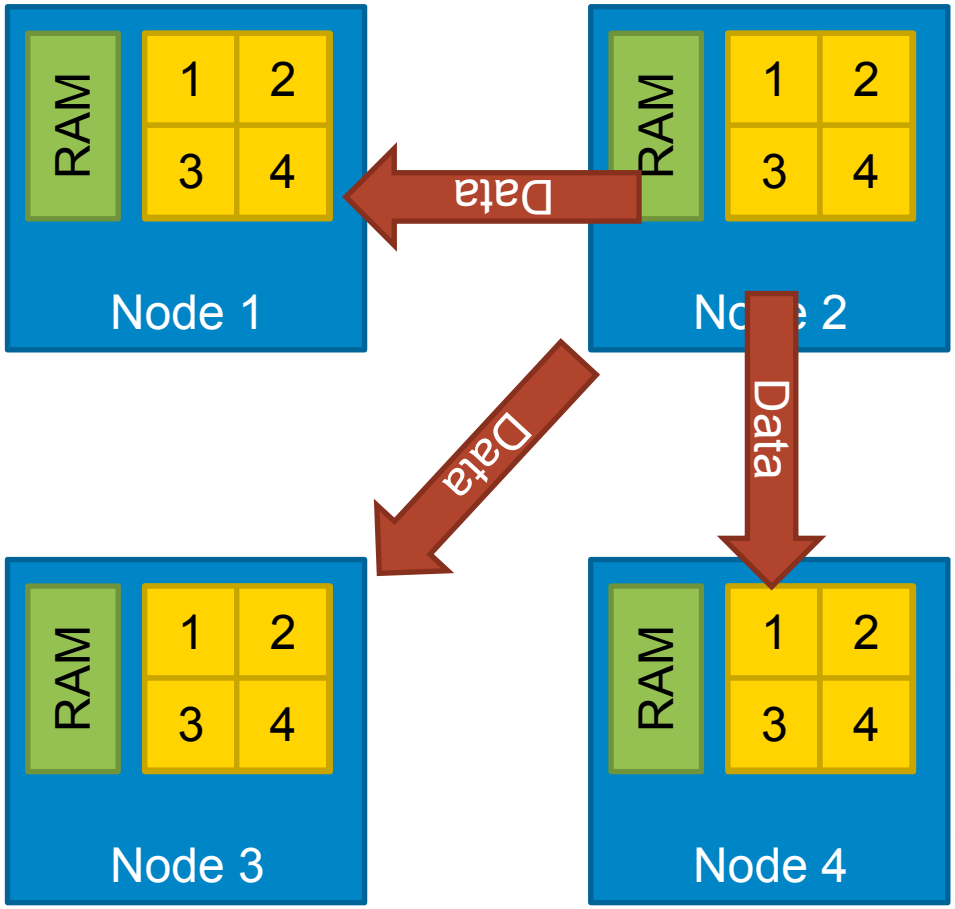
Ranks

- For SPH, this means sending particle information between ranks
- Each rank is responsible for computing the forces of one set of particles
- Once the positions are updated, this information must be shared with *every other rank*

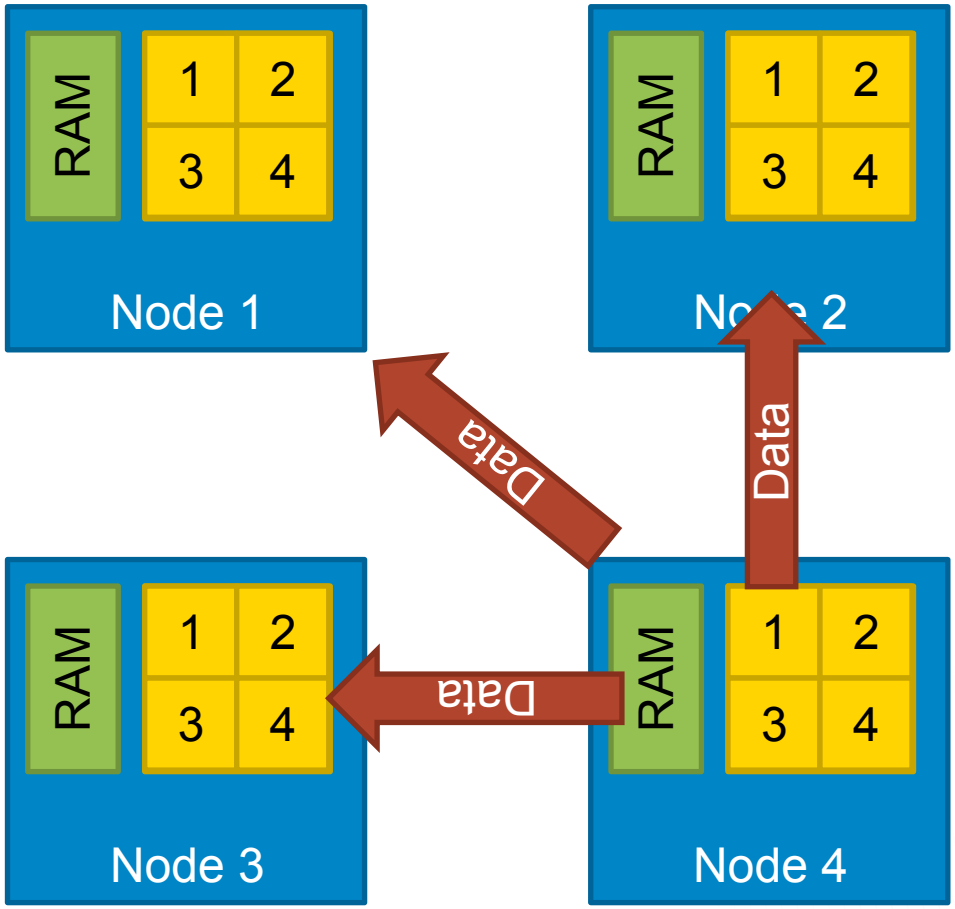
Ranks



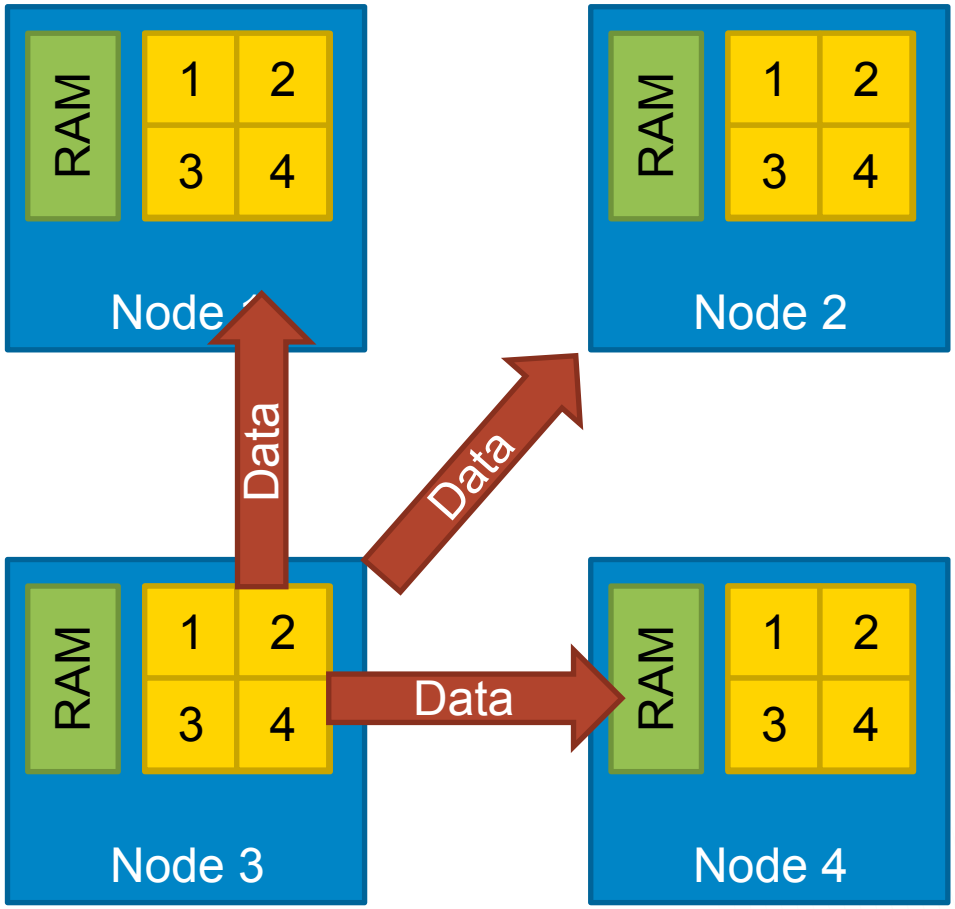
Ranks



Ranks



Ranks



Ranks

- Once every rank knows what every other rank has done, we can move to the next timestep

Quick Recap

Quick Recap

- We are simulating a liquid using 1 million particles
- We need to calculate the forces between *each pair* of particles
  - 1 million X 1 million = 1 trillion force pairs

Quick Recap

- Each of 4 *ranks* is responsible for 250,000 particles
  - That's 250 billion force pair calculations
- Each node has a quad-core CPU
- This means each rank can use 4 threads
  - That's 250 billion / 4 = 62.5 billion force pairs per core

Quick Recap

- Let's guesstimate that each force pair calculation requires 100 flops
  - A "flops" is a Floating Point Operation Per Second, or just an add or a multiply – basic arithmetic.
- So that's 100 basic arithmetic "moves" to calculate each force pair
- A top-dollar 3Ghz CPU can give us ~ 20 Gigaflops, or 20 billion arithmetic moves per second
- That's about 200 Million Force-pairs per core per second

Quick Recap

- I apologize for all the math, here's what I'm getting at:
  - At 200 Million force pairs per second, it will take each core more than 5 minutes to finish its 62.5 Billion force pairs
  - That's only for 1 timestep
  - If we need to simulate 10,000 timesteps, it will take more than a month

Accelerators

OAK RIDGE National Laboratory

Accelerators

- Normal CPUs are designed to do lots of different things:
  - PowerPoint
  - Email
  - Netflix
  - Etc…

Accelerators

- Graphics Processors (GPUs) are designed to be really awesome at just one thing:
  - Video Games

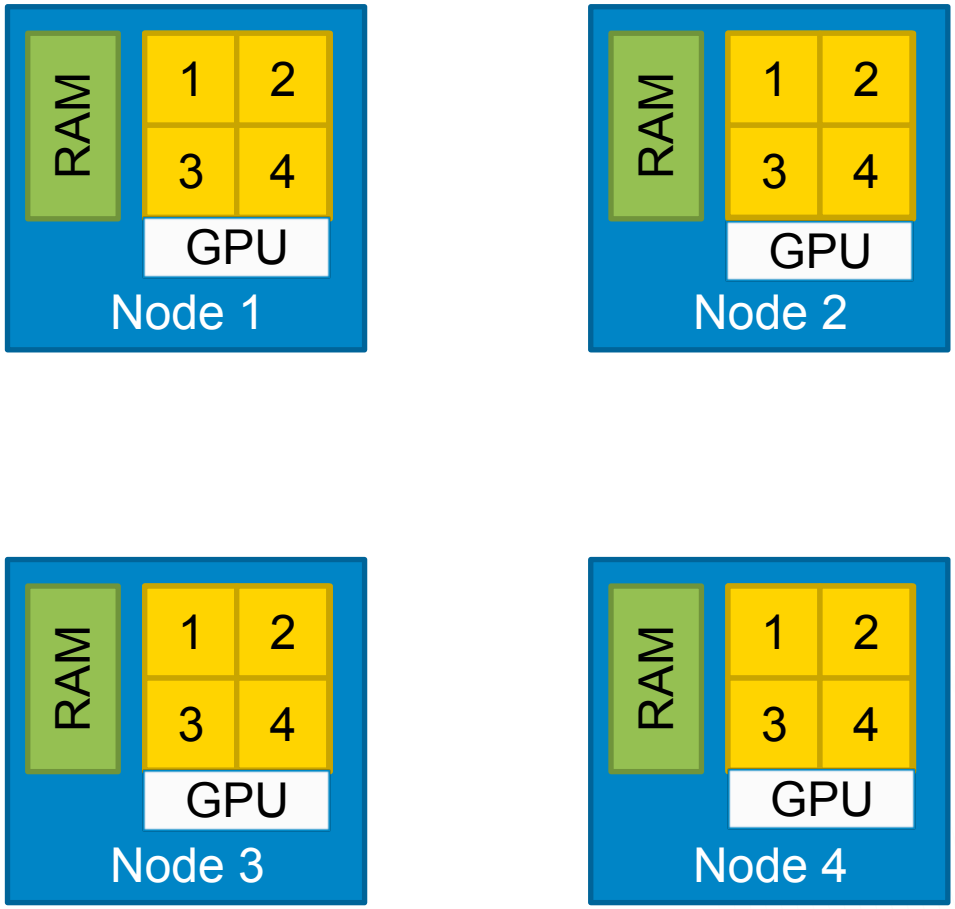
Accelerators

- Animating video games requires *tons and tons* of low-level arithmetic
  - Rotating landscapes, specular lighting, etc all make heavy use of trig functions
- Scientific simulations also require *tons and tons* of low-level arithmetic
  - Linear Algebra, Linear Algebra, Linear Algebra…

Accelerators

- Because GPUs were designed for video games, they offer heaping truckloads of *flops*
- The GPUs in Titan offer more than 1 Teraflop each
  - That's 1 Trillion arithmetic moves per second
- Let's put one of these in each of our 4 compute nodes

Accelerators



Accelerators

- Offloading the force-pair calculations to the GPU makes each node about 10x faster
- This means roughly 30s per timestep for 1 million particles
- Down to 3 days for a 10,000 timestep simulation.

Questions?

OAK RIDGE National Laboratory