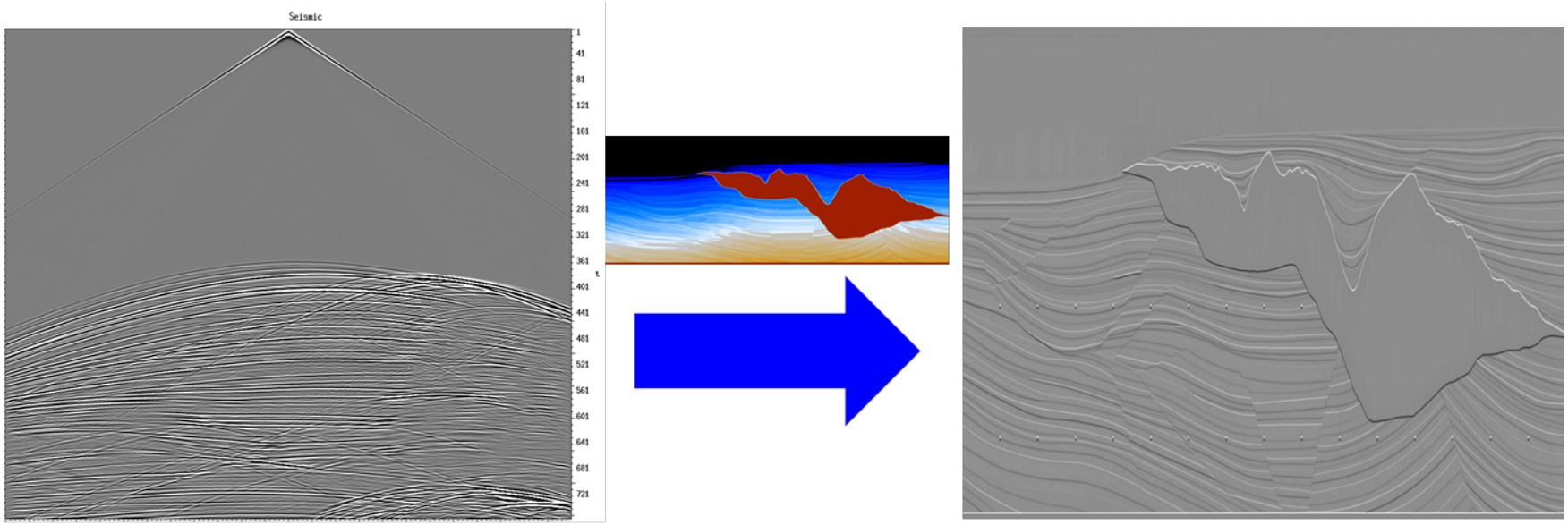
Code Porting to Cray XC40 Lesson Learned

By Jim McClean and Raj Gautam {jim.mcclean,raj.gautam}@pgs.com

Seismic imaging: Create an image of the subsurface from surface measurements

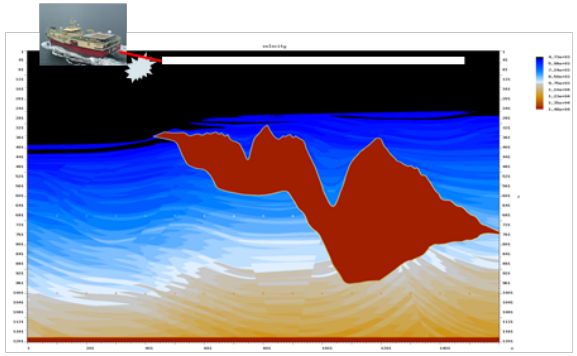


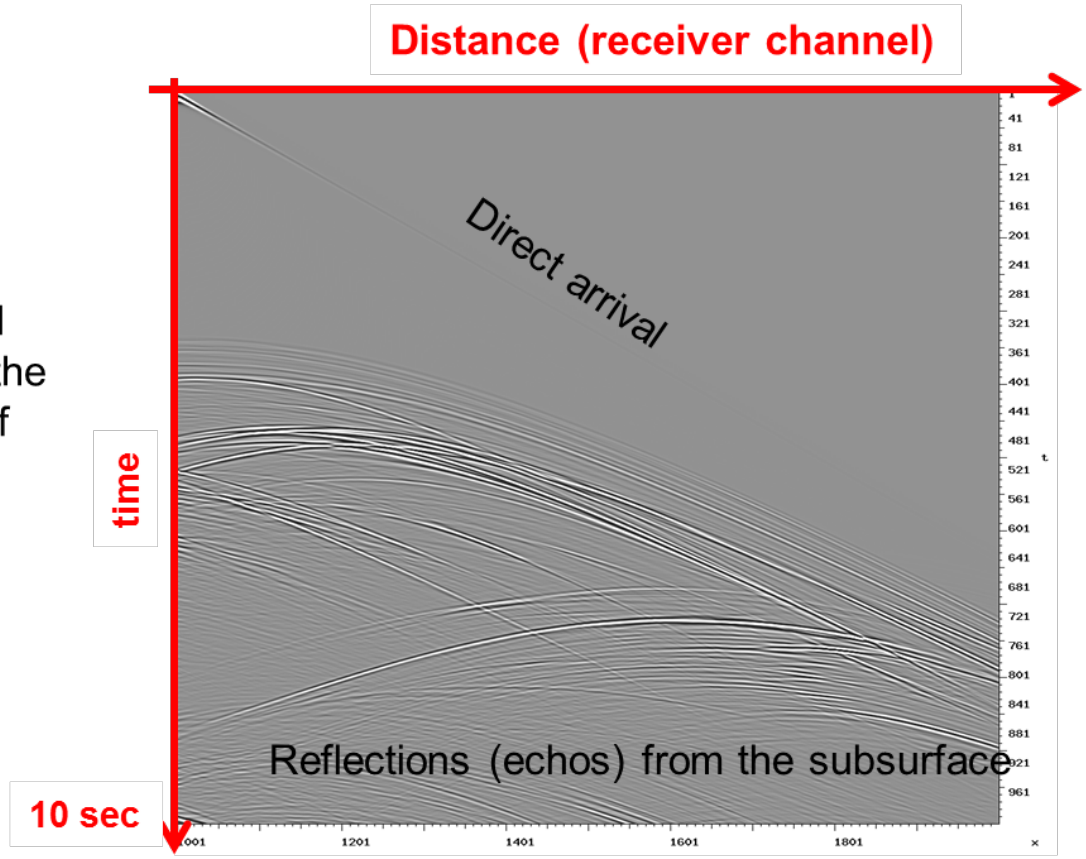
$$data(\mathbf{s},\mathbf{r},t) \xrightarrow[vel]{} image(x,y,z)$$

How exploration seismology has been carried out for decades…

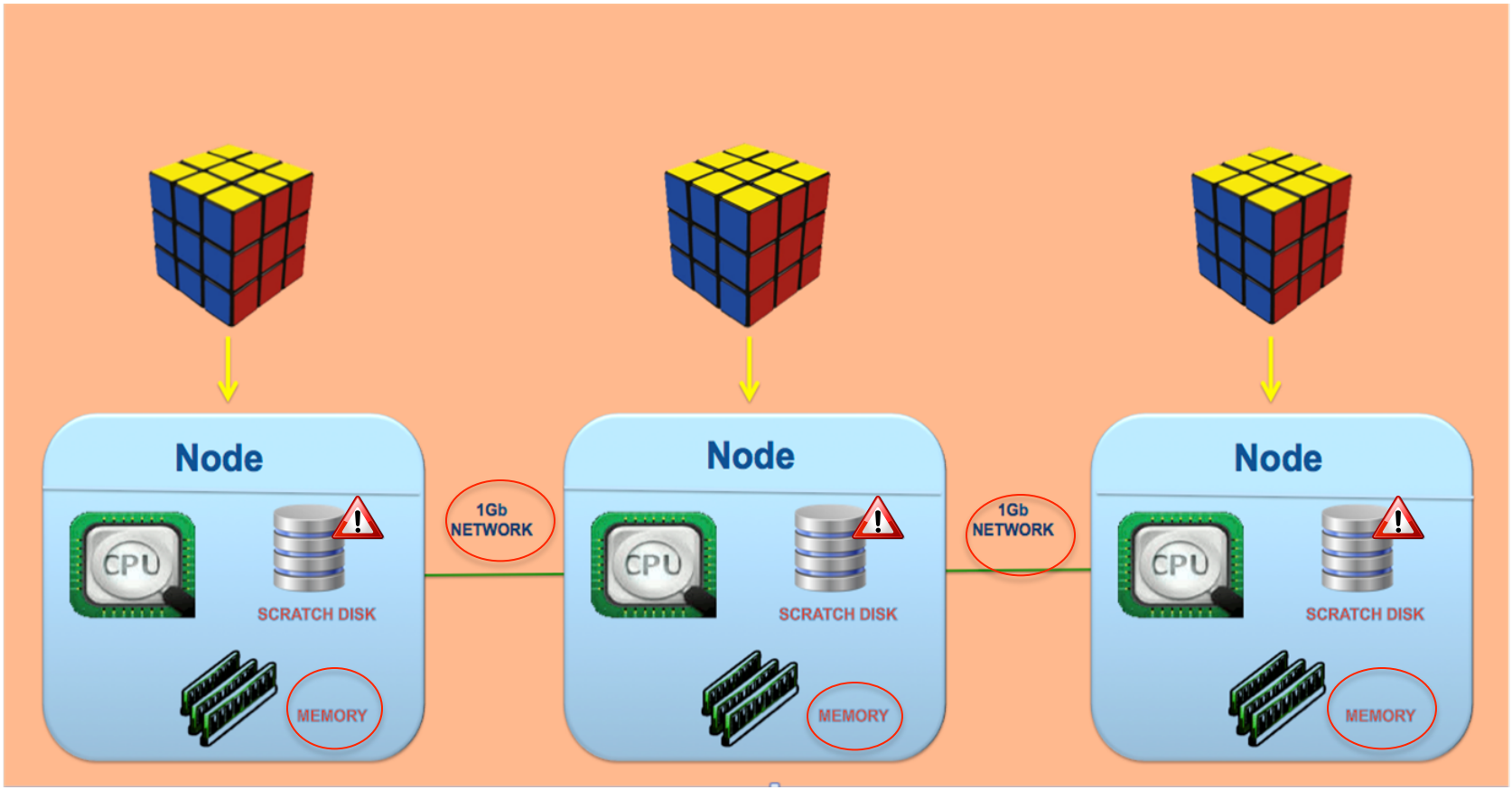
Firing a single shot and recording the reflection:

- This forms a seismic shot record
- Treating each shot as an individual wave equation realization started the “embarrassingly parallel regime” of seismic imaging





Beowulf Cluster:



Slow Network, scratch in node, one shot per node, limited memory, limited SIMD

Motivation for restructuring code

- Exponential growth in compute cost as function of frequency. Wave equation has 3 spatial dimensions and 1 temporal, so computational cost scale as frequency to the 4th power.

- Memory demand increases as the 3rd power of the frequency because
  Bin size ~= vmin / (Fmax * Points Per Wavelength)

- Programs affected are reverse time migration(RTM), Kirchhoff depth migration(KDMIG), and basically all our other imaging programs.

- Suppose a single shot imaging experiment took 128GB memory using RTM algorithm at Fmax of 20Hz.
  Then at 25Hz, it takes 1.95 times as much memory per node to solve this problem

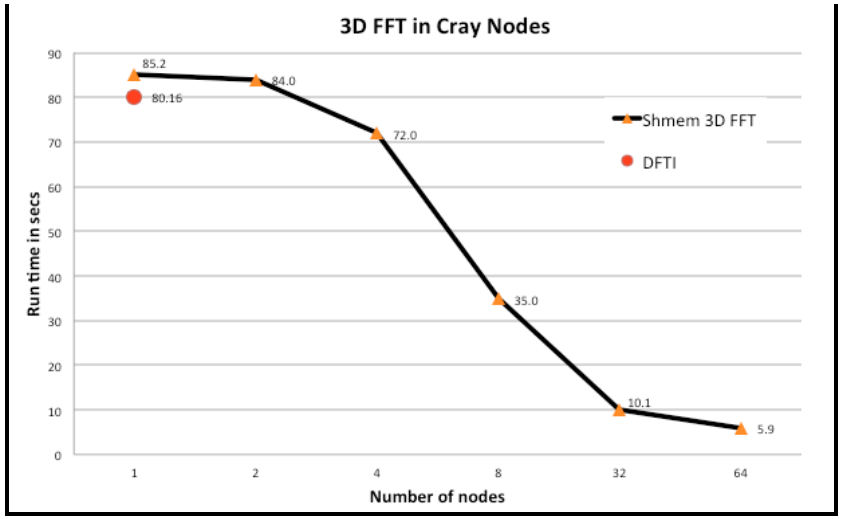
- Solution either expand memory in a node (~256GB+) or go distributed and use fast network to simulate shared memory.

- Why go distributed?
  At 25 Hertz we would need 256GB memory nodes but at 30 Hertz we would need 1 ¼ TB memory per node. Not very practical..

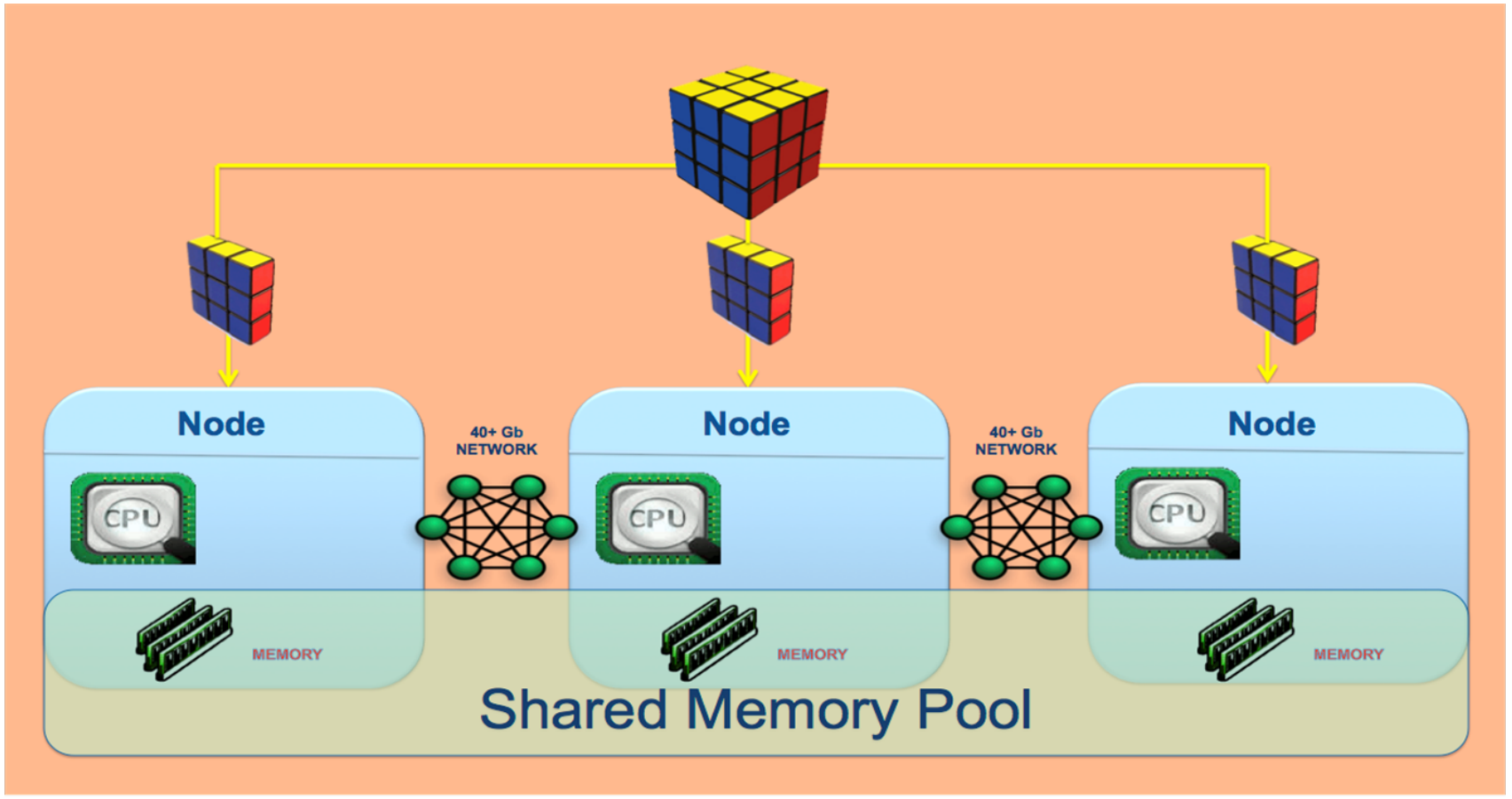
We chose distributed memory ..

Solution

- Go Distributed memory use fast network to simulate shared memory.
- Eliminate scratch disk. Store forward model data in memory.
- TTI RTM acoustic equations are mostly 4th derivatives in 2 directions only. Only a few 3D FFT's required for mixed XYZ derivatives.
- Use SHMEM implementation of 3D FFT. One sided communication twice as fast as two sided MPI.
- Use multiple nodes to store and compute results for one shot image. Typically 20 nodes for 20 to 30 Hertz. This is about 2.5 Terabytes of DDR4 memory to solve one shot experiment.
- Stripe earth model on Lustre disk for fast load time into memory. Use MPI IO instead of serial IO as before.



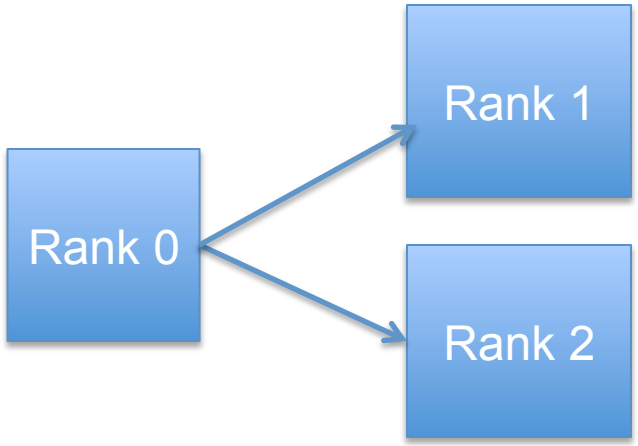
XC40 Implementation:



Faster network, decompose a shot into slabs, distributed memory, more SIMD units, no scratch disk in node

Kirchhoff Depth Imaging

- Ran better on the XC 40 compared to Beowulf clusters out of the box.
- Able to run whole offset classes instead of just tiles. This was due to improved networking on XC 40
- However we still observed that broadcasting travel times to the nodes was a limiting factor.

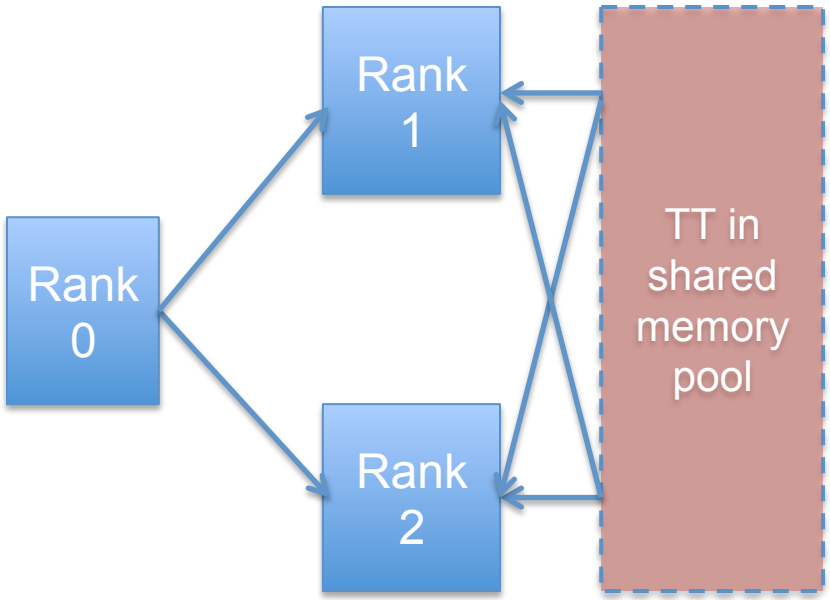


Rank 0 reads the data and travel time tables from disk and broadcasts it to other ranks.

Reading Travel times is the limiting factor.

Kirchhoff Depth Imaging (Solution)

- MPI IO to read Travel Time tables into each node
- Travel Times tables stored in shared memory pool

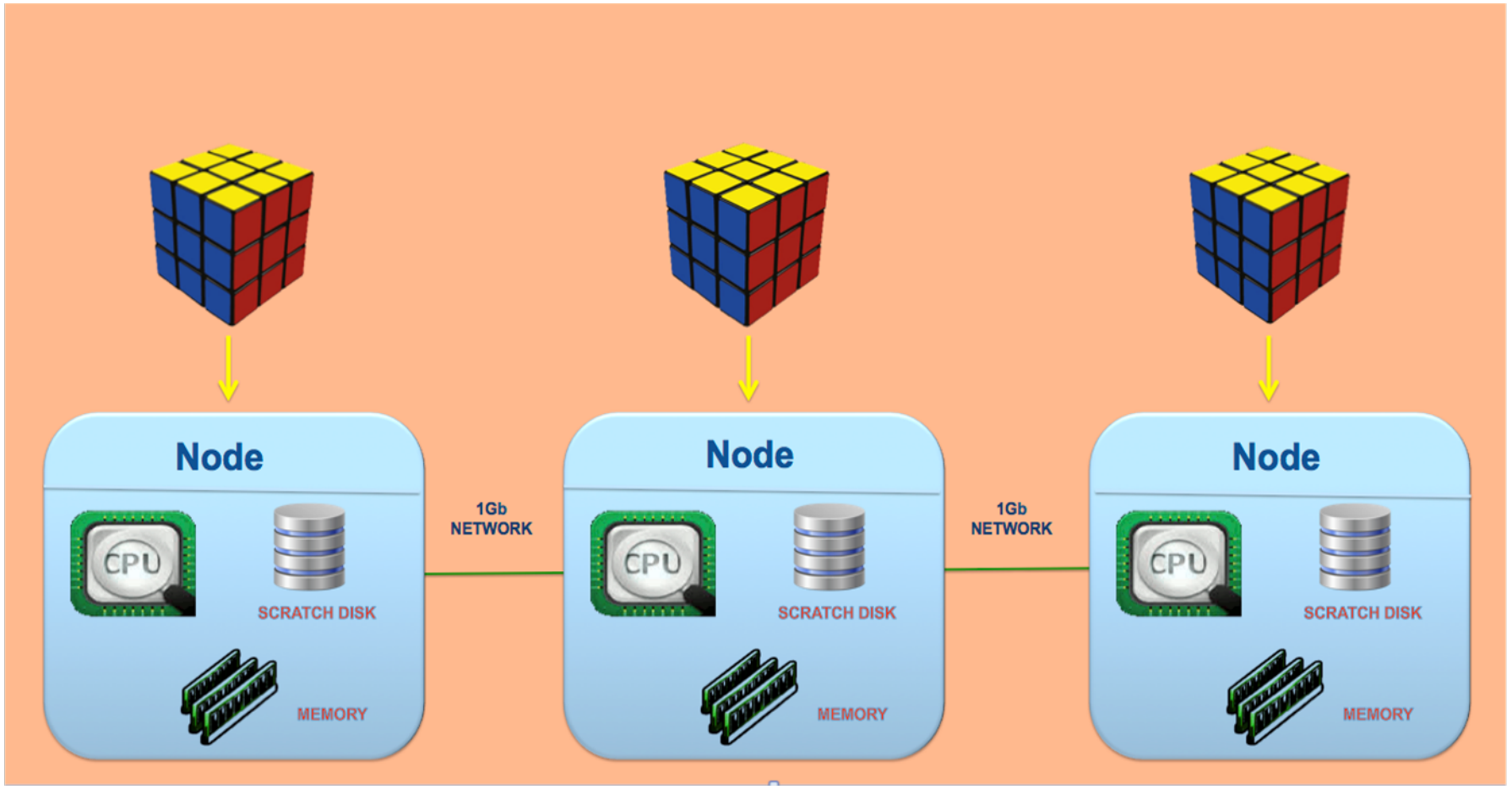


Rank 0 still broadcasts bulk data.
But worker nodes get TT information from shared memory pool!

Wave equation migration(WEM)

- WEM originally used paradigm similar to RTM
- One shot per node, intermediate results and earth model kept on scratch disk, or everything has to fit in memory.
- Scratch disk IO is limiting factor as maximum frequency is increased. Increasing inefficiency with frequency.
- Memory also increase with 3rd power as in RTM case, also computer required increases with 4th power as well.

Wave equation migration(WEM) Before

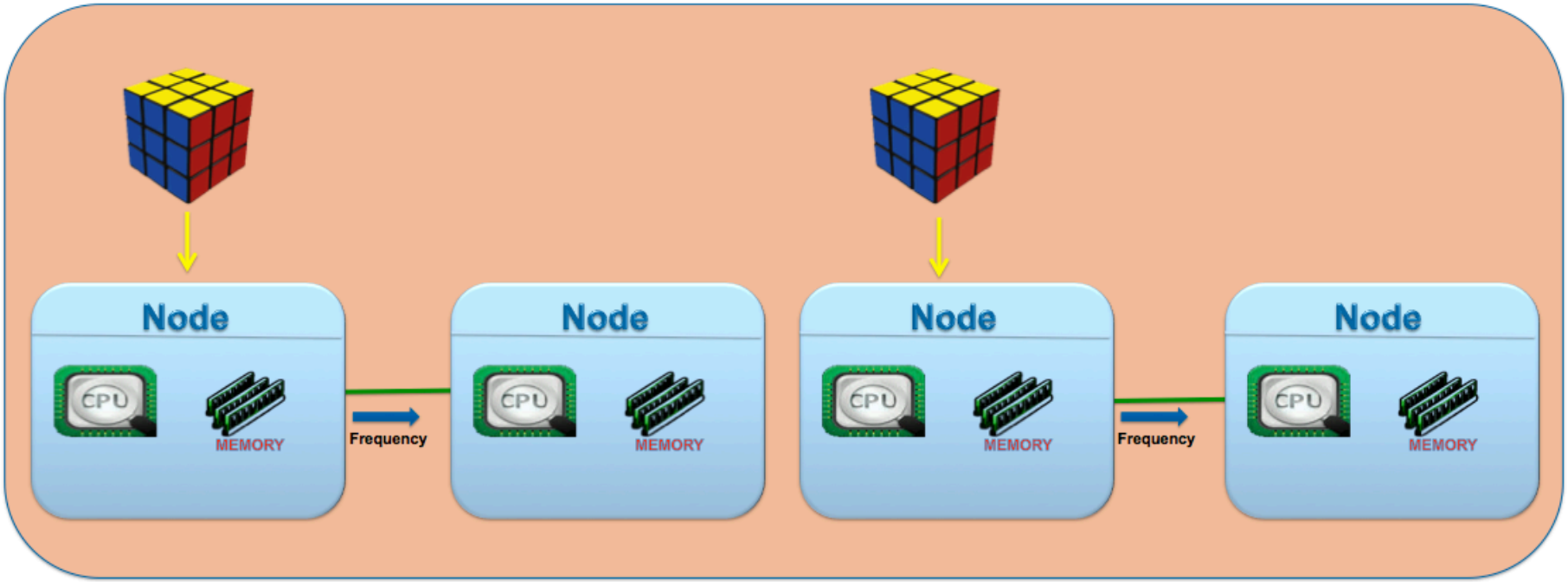


WEM(Solution)

- Go distributed use more than one node to process a shot. Divide responsibility for output depth ranges amongst two or more nodes processing a shot. Split earth model parameters amongst the two or more nodes as well.

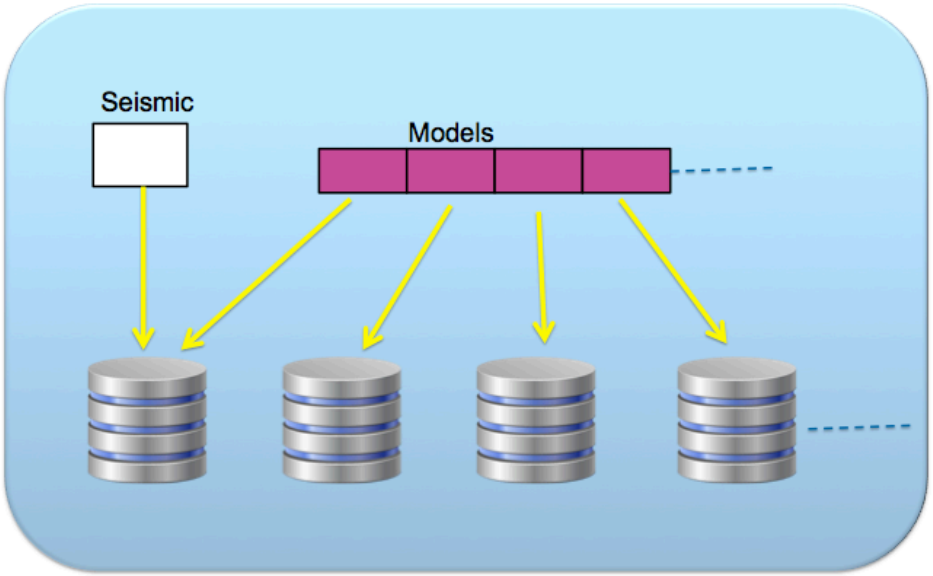
- Pass frequency packets amongst the nodes processing a shot. From lower ranked to higher ranked.

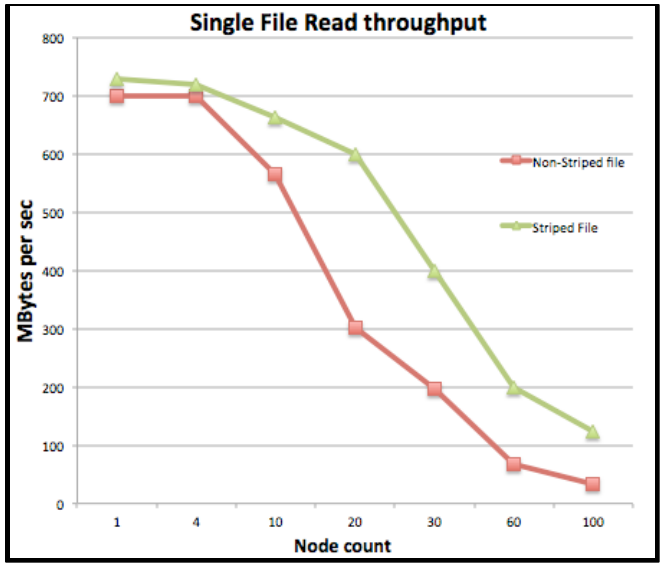
- Scratch disk is largely eliminated. Everything is contained in memory. Add more nodes as maximum frequency is increased.



Lustre File System IO

- Files read by all the nodes are striped across all the OSTs. These are earth model data and Travel Times tables.
- Files that are read by one process at a time are stored in one OST (not striped). It tends to distribute evenly when the system is full.
- Two Lustre FS:
  - Isolate the IO among different algorithms to overcome noisy neighbor problems.
  - Isolate IO – read from one and write to other
  - Staging FS and Active FS





Conclusions

- Going distributed computing, getting rid of scratch disk per node.
- Allowed more reliable computing experience
- Allowed capabilities we did not have before
- Lustre file system along with striping and MPI IO allowed effective use of parallel IO.