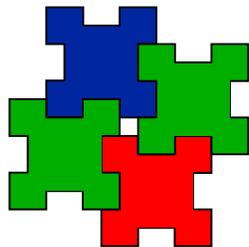




Petascale Virtual Machine: Computing on 100,000 Processors

AI Geist
Oak Ridge National Laboratory
www.csm.ornl.gov/~geist



EuroPVM-MPI 2002
Linz, Austria
October 1, 2002





Petascale Virtual Machine



Another kind of “PVM”

This talk will describe an exciting new project at ORNL to study scalability to 100,000 processors.

IBM Blue Gene System

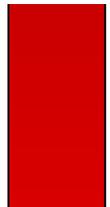
Computing on 100,000 processors, critical issues of scalability and fault tolerance of applications.

Superscalable algorithms with natural fault tolerance for petascale environments.

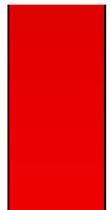
Demo of ORNL’s 100,000 processor simulator



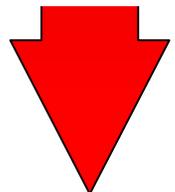
Terascale



Petascale



Beyond



Collaboration with IBM Blue Gene Team

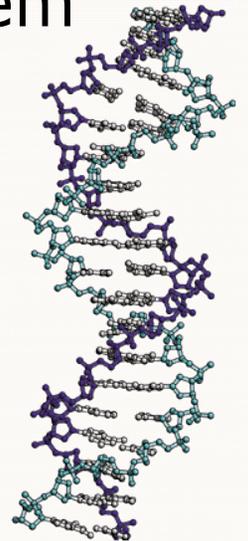
- ORNL/IBM Blue Gene CRADA to develop computational biology applications for Blue Gene.
- ORNL Research collaboration on the design of Blue Gene/D

Develop a theory of super-scalable algorithms

- Scale invariance
- Natural fault tolerance

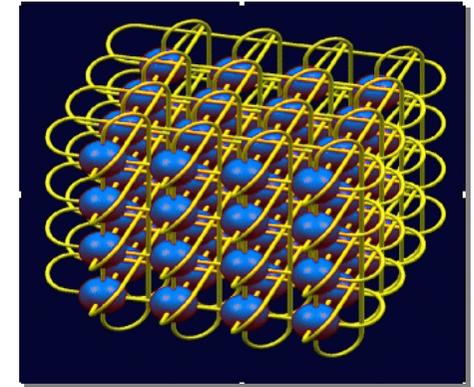
Develop Simulator for 100,000 processor system

- Focus on building tool for apps to think super scale
- And get them thinking about failure**
- Adjustable topology not Blue Gene specific



IBM Blue Gene/L System

Research Project inside IBM to investigate using Power 4 system-on-a-chip to create a low price/performance system.

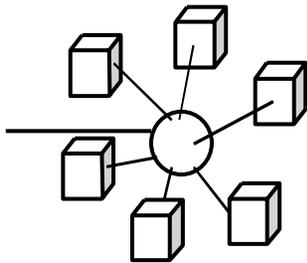


Designed by IBM Research – not a product

- IBM CMOS Cu11 Technology
- 65,536 processor chips
- 4MB Embedded DRAM + External Commodity DDR SDRAM
- (2) Processing cores (440) + FPUs (2.8 GF each)
(one is communication co-processor)
- 180 TF peak

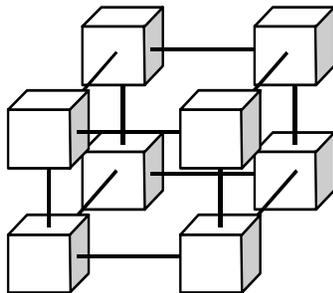
IBM Blue Gene Interconnects

65536 nodes interconnected with three integrated networks



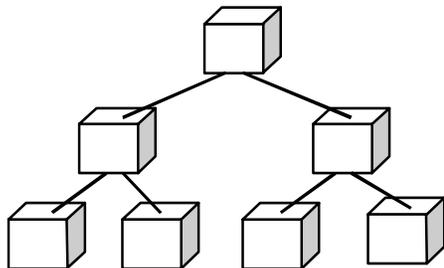
Ethernet

- Incorporated into every node ASIC
- Disk I/O
- Host control, booting and diagnostics



3 Dimensional Torus

- 2.8 Gb/s on all 12 node links (total of 4.2GB/s for each node)
- Communication backbone
- 134 TB/s total torus interconnect bandwidth
- 2.8 TB/s bisectional bandwidth



Global Tree

- One-to-all or all-all broadcast functionality
- Arithmetic operations implemented in tree
- 1.4 GB/s of bandwidth from any node to all other nodes
- Latency of tree less than 1usec
- 90TB/s total binary tree bandwidth (64k machine)

IBM Blue Gene 82,000 processor System

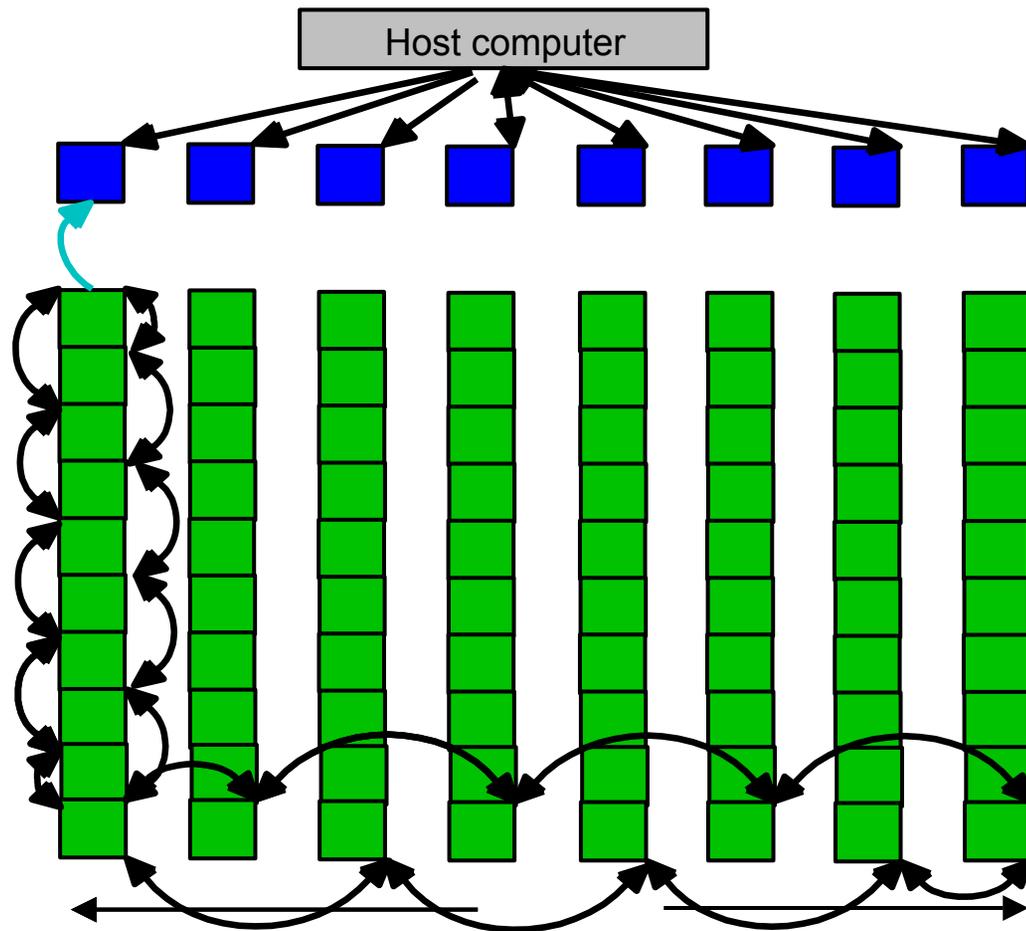
LLNL Configuration 180/360 Tflops peak

BlueGene/L Processing Nodes

- 81920 Nodes
 - Two major partitions
 - 65536 Nodes Production
 - 16384 Nodes code development

Host System:

- Diagnostics, Booting, Archive
- Application dependent requirements



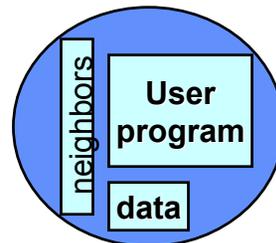
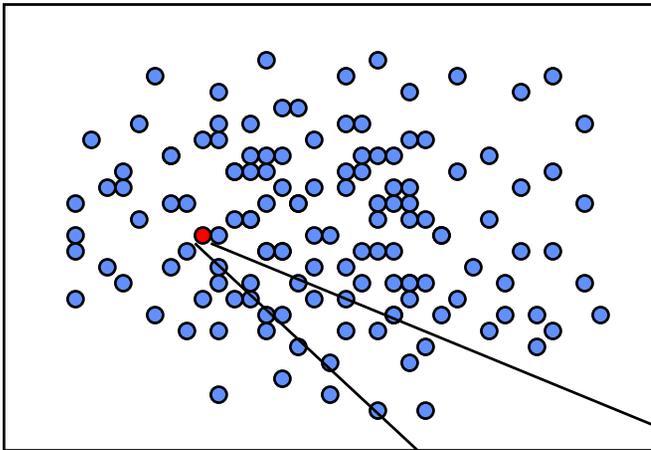
Observations

- The next generation of peta-scale computers are being designed with 50,000-100,000 processors.
- Amdal's Law will kill you at such scale. ASCI has seen real apps get 1%-10% efficiency on 10,000.
- The mean time to failure for such a system is likely to be just a few minutes.
- Application checkpoint/restart is today's typical fault tolerance method.
 - Not efficient use of resources to restart 99,000 nodes because one failed.
 - Not even possible when MTBF becomes less than the time to restart.

Super-scalable Algorithms

Establish a theoretical foundation for a whole new class of algorithms called Super-scalable algorithms

Goal: 100,000 dependent tasks

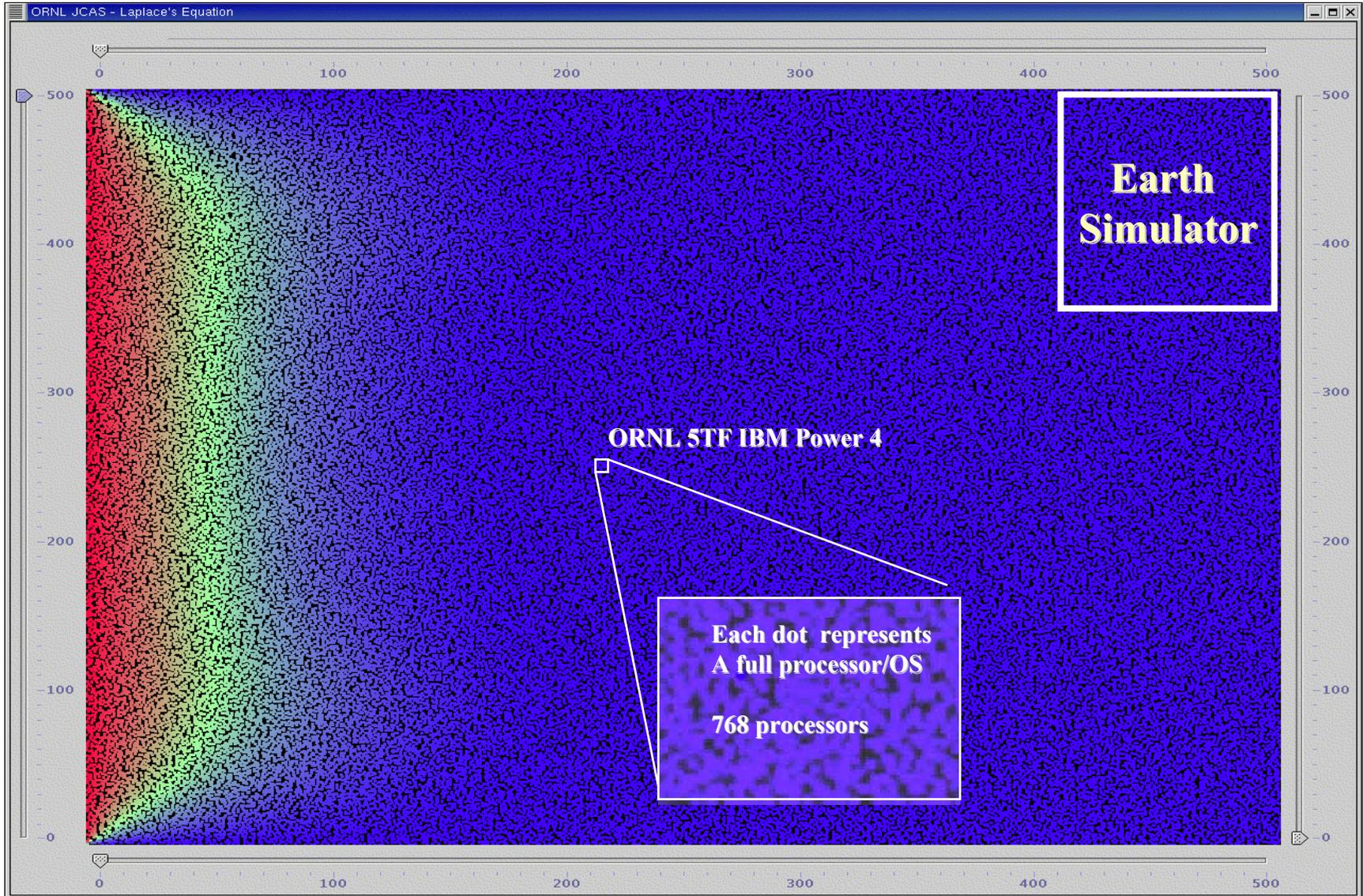


Theoretical Characteristics:

- Scale Invariance
- Natural fault tolerance
- Self Healing

Each dot represents a processor running a task that has a fixed set of neighbors

100,000 Processor Simulator for Applications



Large-Scale Fault Tolerance

Taking fault tolerance beyond checkpoint/restart.

Developing fault tolerant algorithms is not trivial. Anything beyond simple checkpoint/restart is beyond most scientists. Many recovery issues must be addressed

Doing a restart of 90,000 tasks because of the failure of 1 task, may be very inefficient use of resources.

When and what are the recovery options for large-scale simulations?

Fault Tolerance – a new perspective

Future systems are being designed with 100,000 processors. The time before some failure will be measured in minutes. Checkpointing and restarting this large a system could take longer than the time to the next failure!

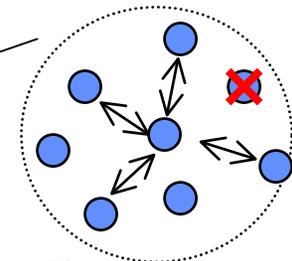
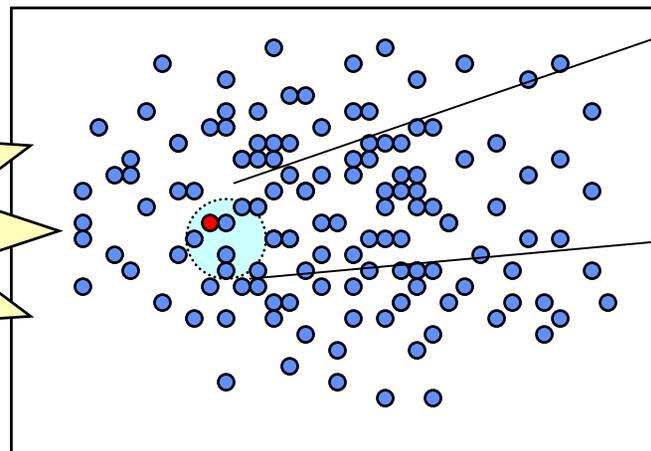
A new perspective on fault tolerance is needed.

What to do?

Development of algorithms that can be **naturally fault tolerant** I.e. failure anywhere can be ignored?



The search is on to find what classes of problems can be made naturally FT



Which Classes of Problems can be solved?



Explore a range of problem classes that span independent, local and global information needs.

Independent tasks

- Example class Monte Carlo integration
- Methods used by many biology app.

**Done
SETI@HOME**

Local exchange

- Example classes finite difference, finite element
- Requires the development of meshless methods
- What attributes allow natural fault tolerance

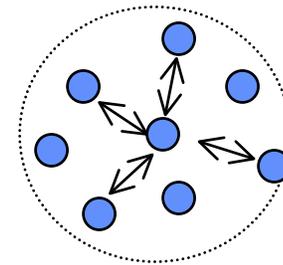
Global information

- Example classes gravity, radiative transport
- Regular globally distributed neighbor set
- Neighbor set based on problem physics such as $1/r$
- Neighbor set selected randomly

Demonstrated that the scale invariance and natural fault tolerance can exist for local and global algorithms

Local exchange (Christian Engelman)

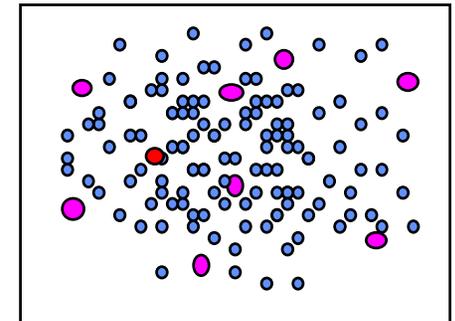
- Demonstrated natural fault tolerance w/ chaotic relaxation, meshless, finite difference solution of Laplace and Poisson problems



local

Global information (Kasidit Chancio)

- Demonstrated natural fault tolerance in global max problem w/random, directed graphs

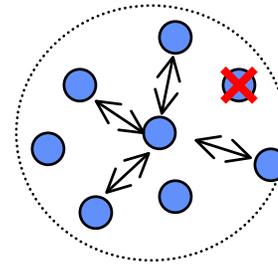


global

Approach for finite difference and finite element algorithms - in general $f(\text{local-region})$

General Approach to local problem

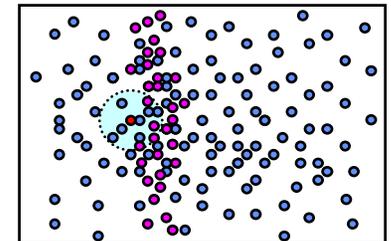
- Mesh free formulation
- Iterative solution
- Chaotic relaxation
- Apply $f()$ to all incoming streams
- Ignore failed neighbors



local

Adaptive Refinement

- Exploit the ability to add and delete processes
imagine pouring in more tasks around shock front, deleting them behind

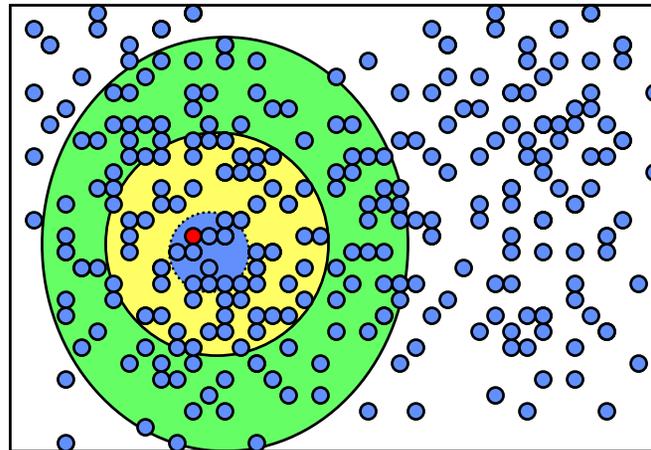


adaptive

Goal: Combine the fast convergence of multigrid with the natural fault tolerance property

Gridless Multigrid

- Hierarchical implementation of the local algorithm applied based on multigrid convergence criteria.

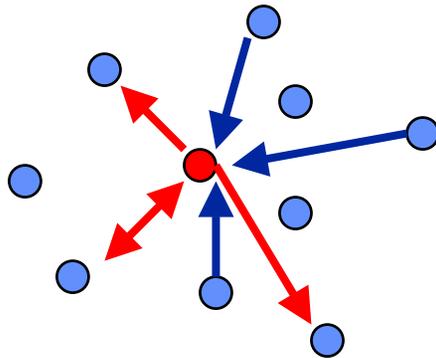


Global Super-scalar algorithms

Approach for functions like global max or min
in general $f(\text{global-information})$

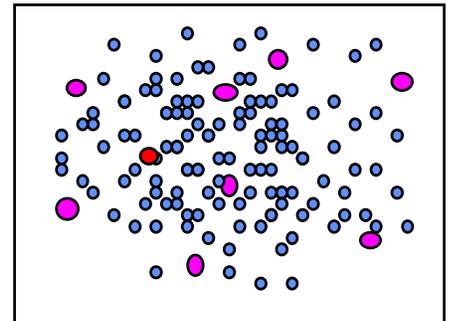
Global information (Kasidit Chanchio)

- Demonstrated natural fault tolerance in global max problem w/random, directed graphs



Probabilistic algorithm where probability of islands rapidly goes to zero as number of neighbors grows.

global



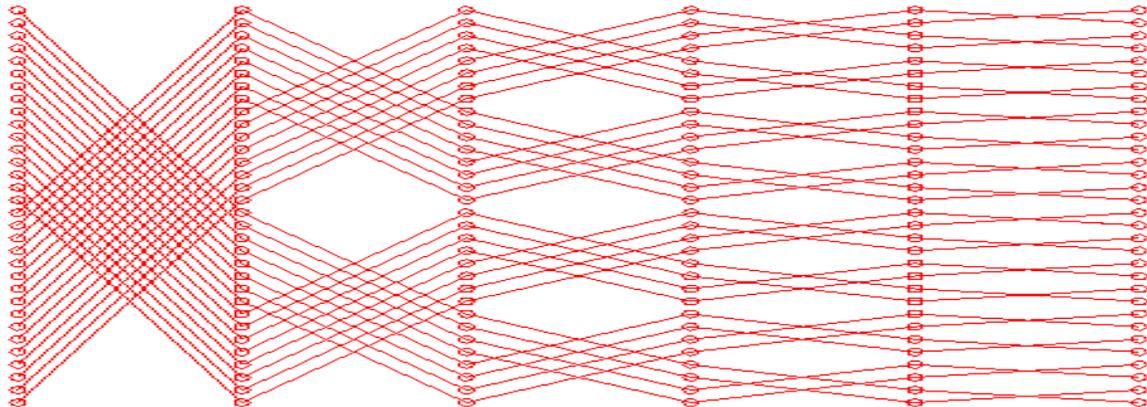
Future Work - FFT Super-scalar algorithm



FFT is a hybrid. It uses local and global information. While naturally fault tolerant Fourier transform algorithms can be created. No FFT ($n \log(n)$) algorithm found. So traditional detect/recovery methods used

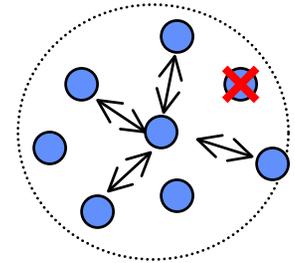
Recovery options being investigated for FFT algorithm

- Replication
- Recreate value from real space
- Distributed, local, in memory state recovery



When Natural Fault Tolerance Not possible

Adapting traditional detect/recover methods to super-scalar systems.

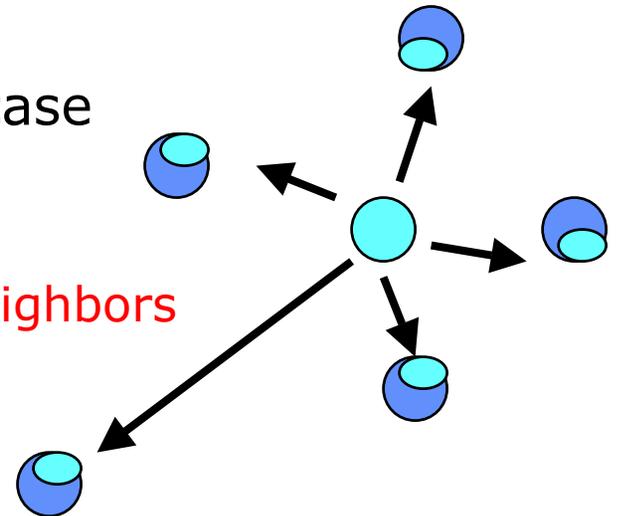


Distributed, local, in memory state recovery

Naturally synchronized by message passing rather than global synchronization of all nodes.

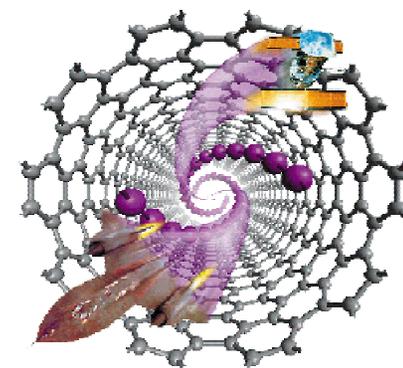
Ways to distribute state in traditional case

- Full multiple copies on neighbors
- Distribute nth of state to each neighbor
- Apply RAID concepts to state stored in neighbors



Applications for Petascale Virtual Machines

- Proteomics – DOE Genomes-to-Life project. Characterize all the molecular machines and regulatory networks in a single microbe.
- Nanotechnology – New KKR algorithm for first principles analysis of atom configurations
- System software monitoring and recovery – Blue Gene OS and runtime system



Develop Simulator for 100,000 proc System



Version 3.0 running now

- 5000 processors simulated at SC2001
- 35,000 processors simulated February 2002
- 100,000 processors simulated July 2002

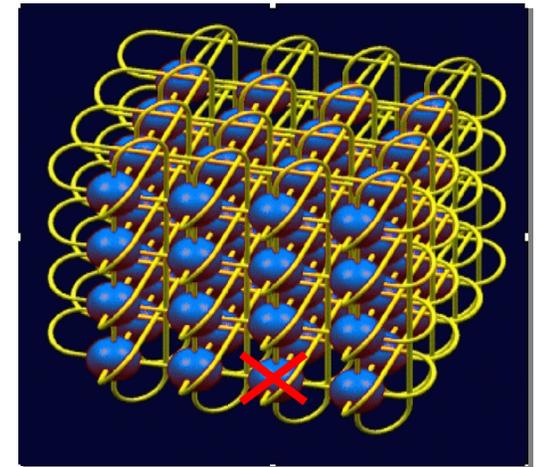


Simulator is a parallel application

- Runs on a Linux Cluster

Adjustable Topology

- Configured at startup (automated in next version)
- Each node sets what nodes it can send to
- What nodes it can receive from



Provides simulation of failures

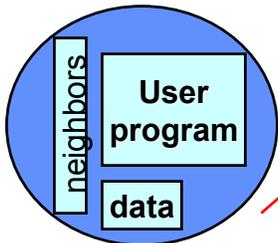
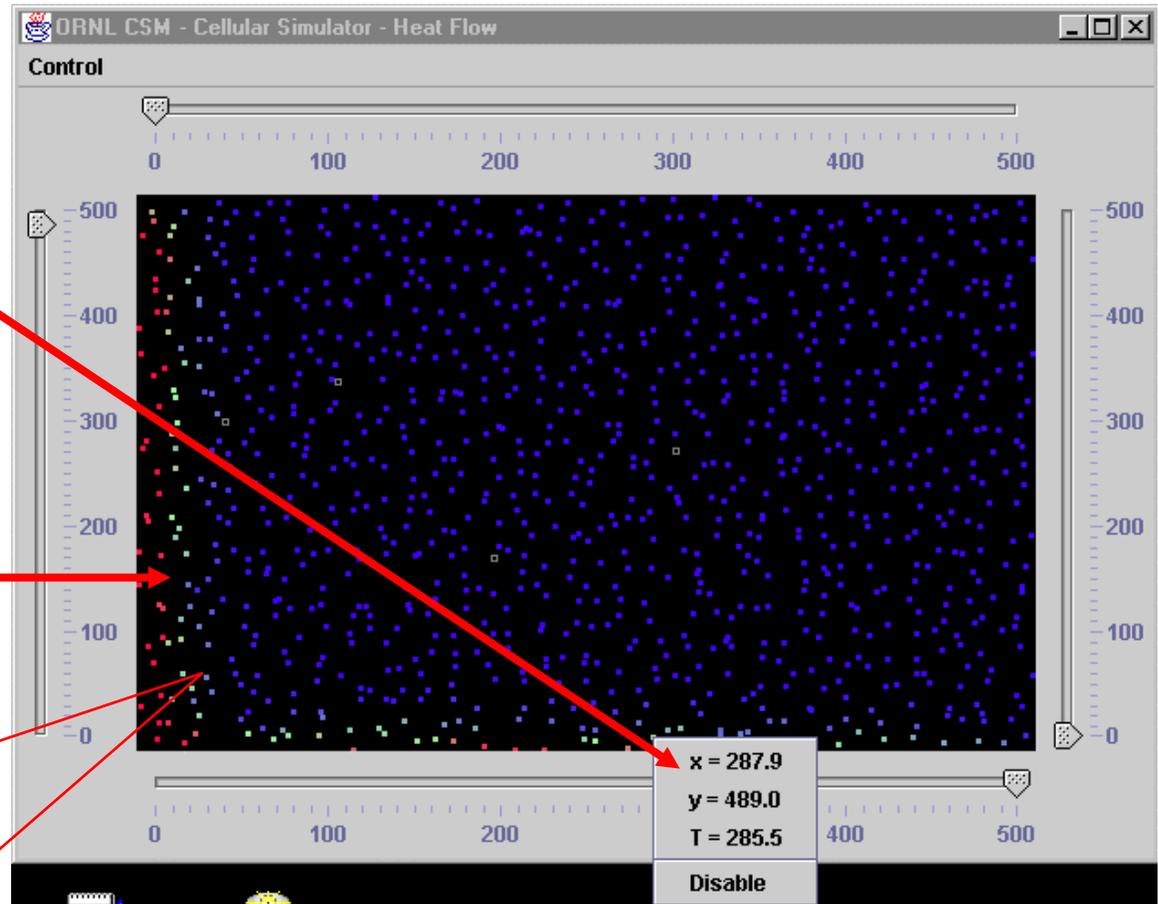
- Both single and groups or nodes can be set to fail

ORNL Super Scale Simulator



Clicking on node pulls up information and disable window

Simulator has function allows node to display an internal variable as color



- Focus on building **tool for apps** to think super scale
- And get them thinking about failure

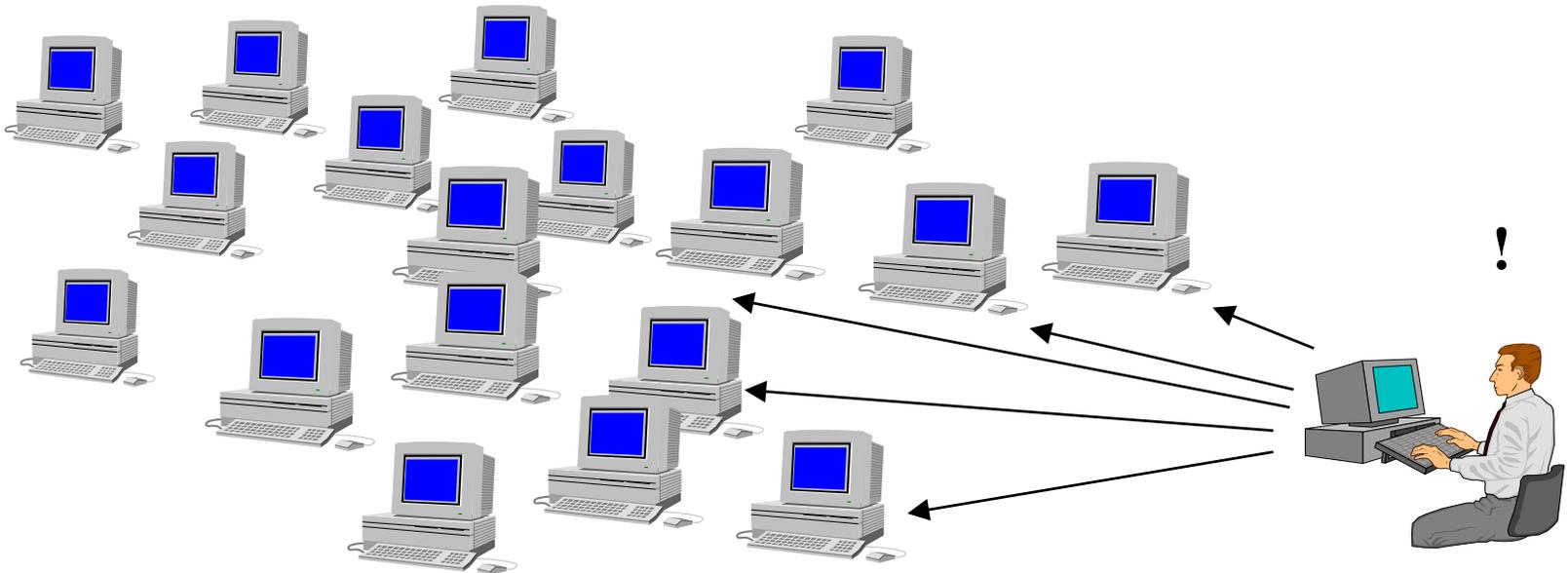
Web Computing Now becomes more interesting!

SETI@home and Entropia searching for primes, genes, and aliens.

- Aliens are harder to find.
- Last 24 hours: 1800 computers, 2 TF/s, 45 TF/day

So far only embarrassingly parallel applications are suitable.

These new super scalable algorithms are not just for Blue Gene!



Final Thought – How can you be sure?

Validation of answer on such large systems

Fault may not be detected

Recovery introduces perturbations

Result may depend on which nodes fail

Result looks reasonable but is actually wrong

Can't afford to run every job three times (SETI approach)

I'll just keep running
the job till I get the
answer I want

