Office of Science
U.S. DEPARTMENT OF ENERGY

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

Super-Scalable Algorithms for Computing on 100,000 Processors

Christian Engelmann and Al Geist

Network and Cluster Computing Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory, Oak Ridge, USA

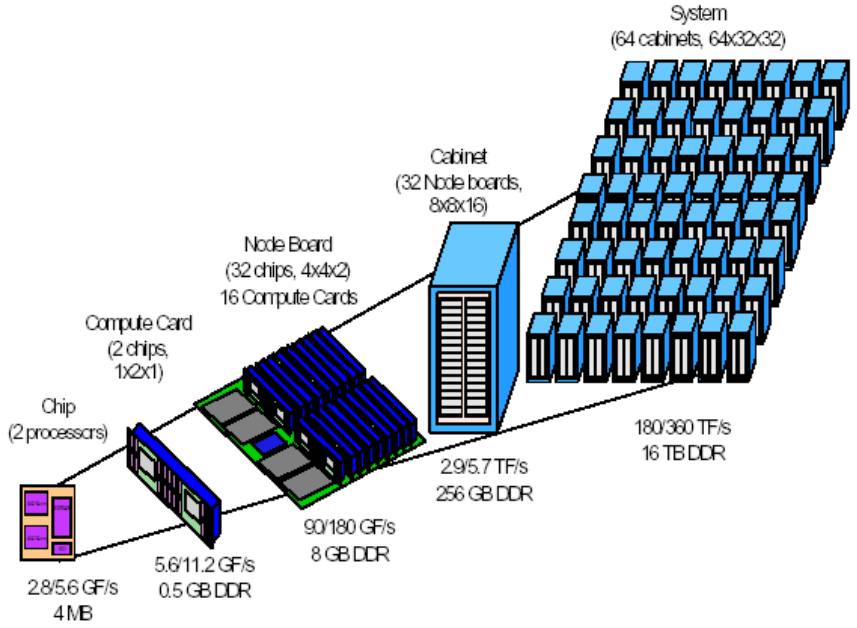
International Conference on Computational Science (ICCS) 2005
Emory University, Atlanta, USA

Super-Scale Architectures

- Current tera-scale supercomputers have up to 10,000 processors.
- Next generation peta-scale systems will have 100,000 processors and more.
- Such machines may easily scale up to 1,000,000 processors in the next decade.
- IBM is currently deploying the Blue Gene/L system at research institutions world-wide.

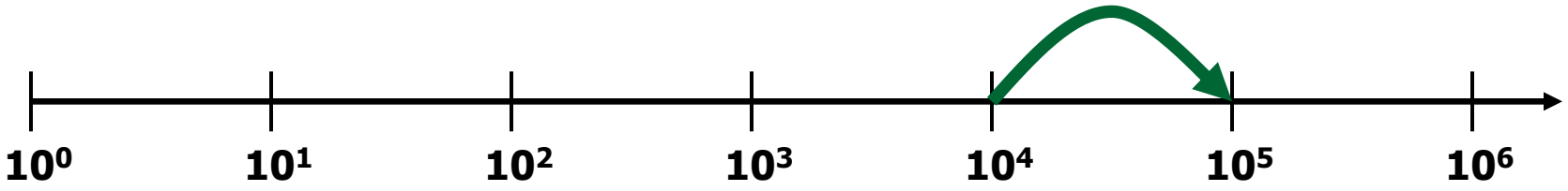
IBM Blue Gene/L

- 64K diskless nodes with 2 processors per node.
- 512MB RAM per node.
- Additional service nodes.
- 360 Tera FLOPS.
- Over 150k processors.
- Various networks.
- Operational in 2005.
- Partition (512 nodes) outages on single failure.
- MTBF = hours, minutes?



Scalability Issues

- How to make use of 100,000 processors?
- System scale jumps by a magnitude.
- Current algorithms do not scale well on existing 10,000-processor systems.
- Next generation super-scale systems are useless if efficiency drops by a magnitude.

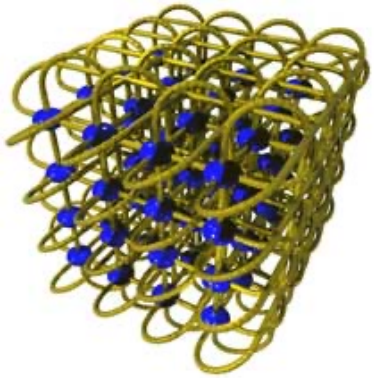


Fault-tolerance Issues

- How to survive on 100,000 processors?
- Failure rate grows with the system size.
- Mean time between failures (MBTF) may be a few hours or just a few minutes.
- Current solutions for fault-tolerance rely on checkpoint/restart mechanisms.
- Checkpointing 100,000 processors to central stable storage is not feasible anymore.

ORNL/IBM Collaboration

- Development of biology and material science applications for super-scale systems.
- Exploration of super-scalable algorithms.
  - Natural fault-tolerance.
  - Scale invariance.
- Focus on test and demonstration tool.



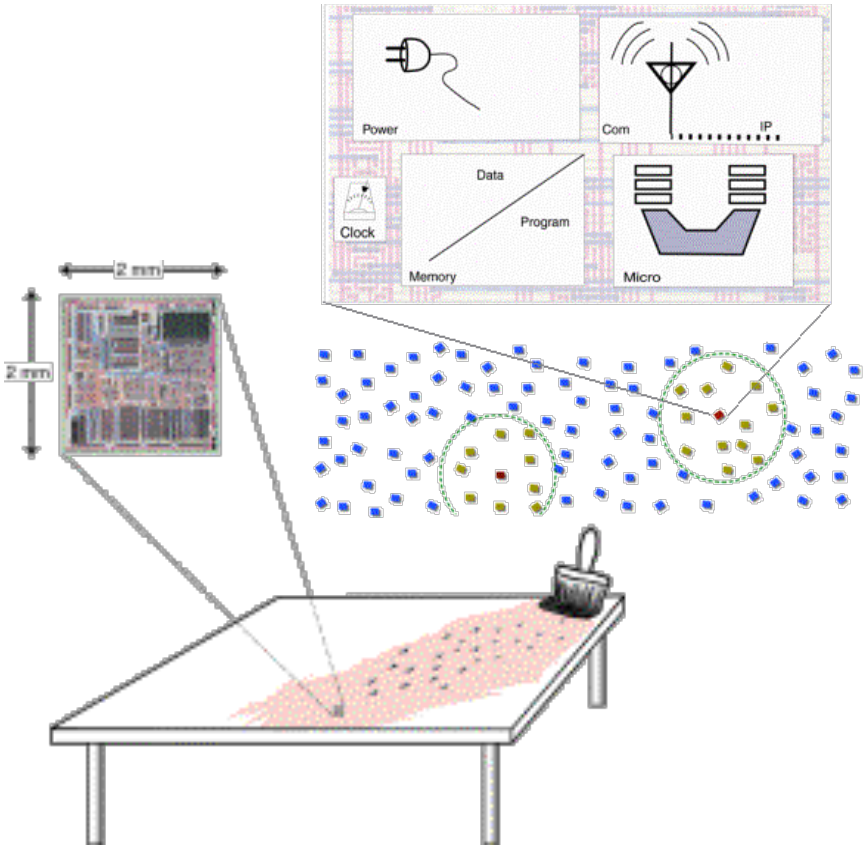
- Get scientists to think about scalability and fault-tolerance in super-scale systems!

Cellular Algorithms Theory

- Processes have only limited knowledge mostly about other processes in their neighborhood.
- Application is composed of local algorithms.
- Less inter-process dependencies, e.g not everyone needs to know when a process dies.
- Peer-to-peer communication with overlapping neighborhoods promotes scalability.

- MIT Media Lab. Research: Paintable Computing.

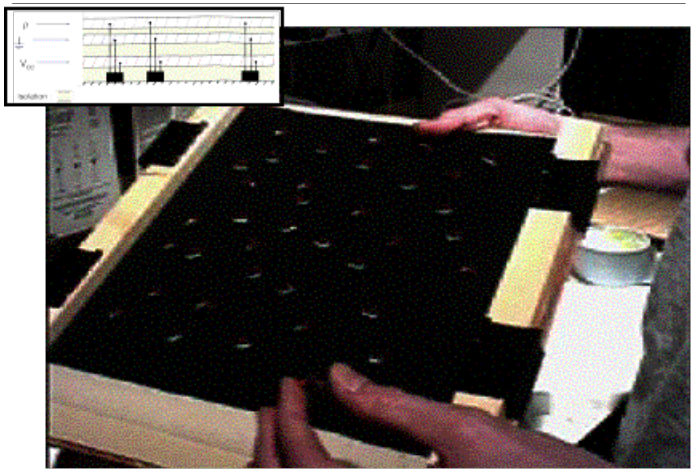
MIT Research: Paintable Computing

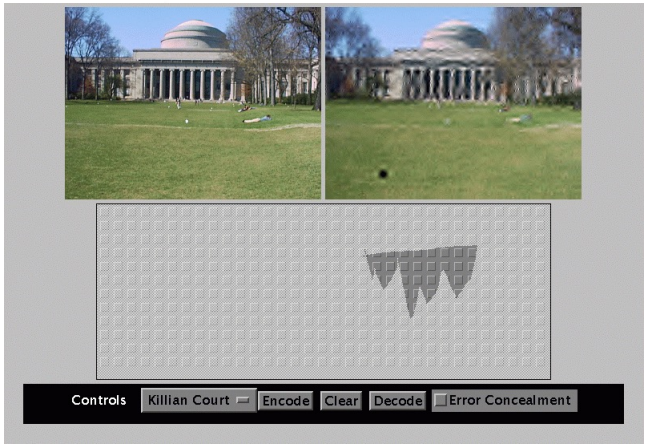


- In the future, embedded computers with a radio device will get as small as a paint pigment.
- Supercomputers can be easily assembled by just painting a wall of embedded computers.
- Applications are driven by cellular algorithms.

MIT Research: Pushpin Computing

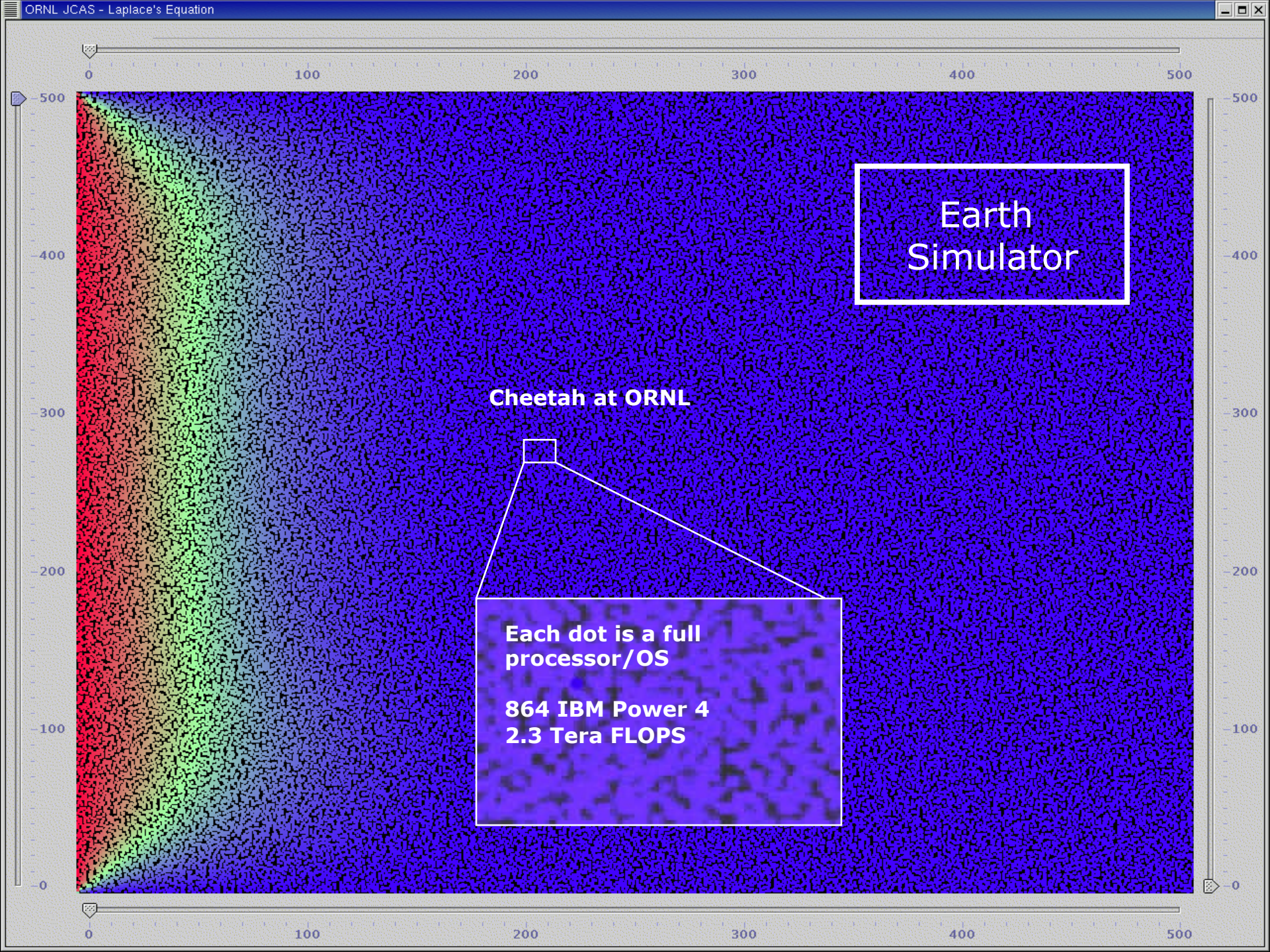
- 100 embedded nodes.
- 1.25m x 1.25m pushpin board provides power.
- Initial applications:
  - Distributed audio stream storage.
  - Fault-tolerant holistic data (image) storage.
- Ongoing research:
  - Sensor networks.

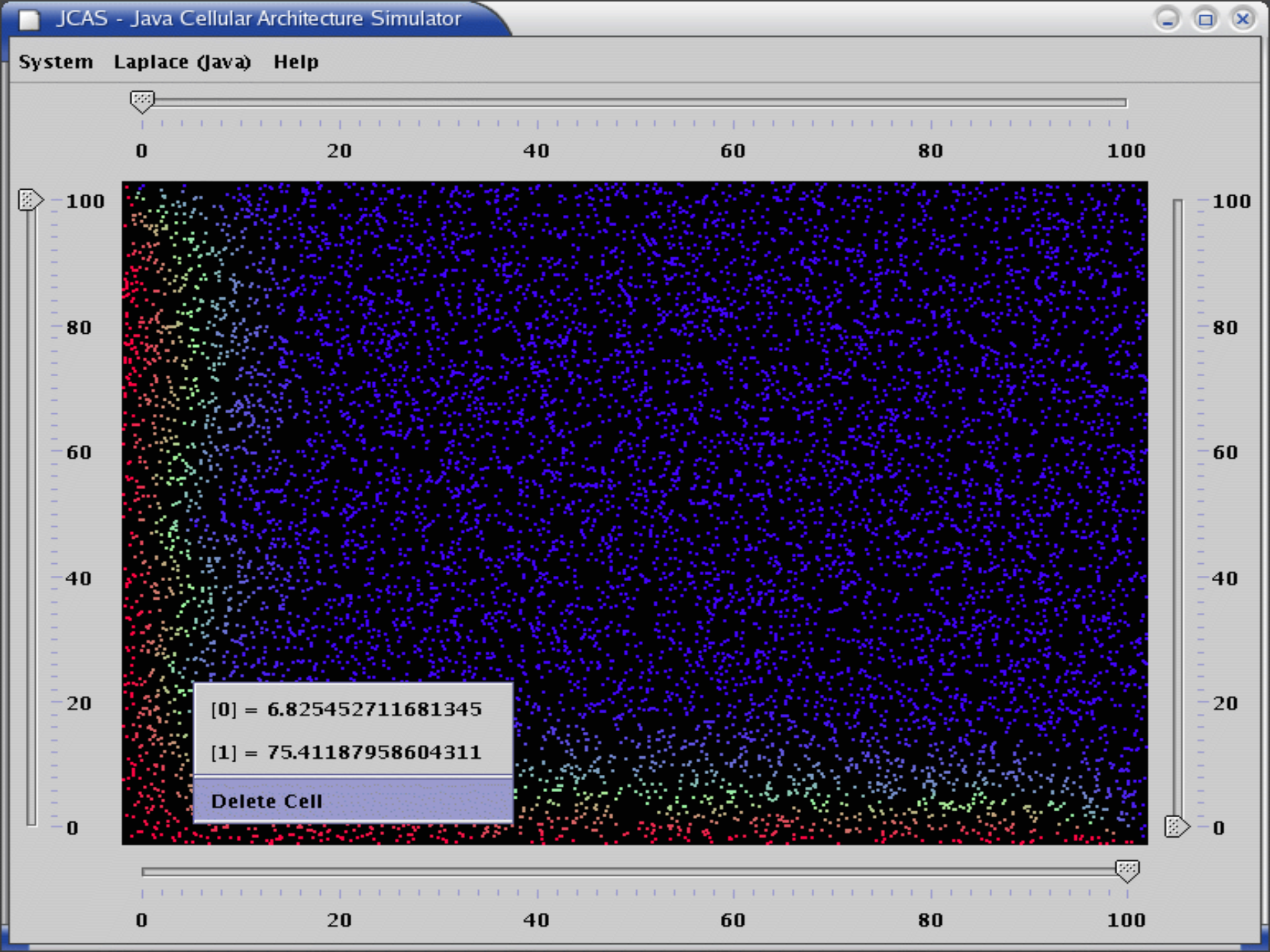




Cellular Architecture Simulator

- Developed at ORNL in Java with native C and Fortran application support using JNI.
- Runs as standalone or distributed application.
- Lightweight framework simulates up to 1,000,000 lightweight processes on 9 real processors.
- Standard and experimental networks:
  - Multi-dimensional mesh/torus.
  - Nearest/Random neighbors.
- Message driven simulation is not in real-time.
- Primitive fault-tolerant MPI support.


ORNL JCAS - Laplace's Equation
0
100
200
300
400
500
Earth Simulator
Cheetah at ORNL
Each dot is a full processor/OS
864 IBM Power 4
2.3 Tera FLOPS


JCAS - Java Cellular Architecture Simulator
System
Laplace (Java)
Help
0
20
40
60
80
100
[0] = 6.825452711681345
[1] = 75.41187958604311
Delete Cell

Super-scalable Algorithms Research

- Extending the cellular algorithms theory to real world scientific applications.
- Exploring super-scale properties:
  - Scale invariance – fixed scaling factor that is independent from system and application size.
  - Natural fault-tolerance – algorithms get the correct answer despite failures without checkpointing.
- Gaining experience in programming models for computing on 100,000 processors.

Explored Super-scalable Algorithms

- Local information exchange:
  - Local peer-to-peer updates of values.
  - Mesh-free chaotic relaxation (Laplace/Poisson).
  - Finite difference/element methods.
  - Dynamic adaptive refinement at runtime.
  - Asynchronous multi-grid with controlled or independent updates between different layers.
- Global information exchange:
  - Global peer-to-peer broadcasts of values.
  - Global maximum/optimum search.

Super-scalable Fault Tolerance

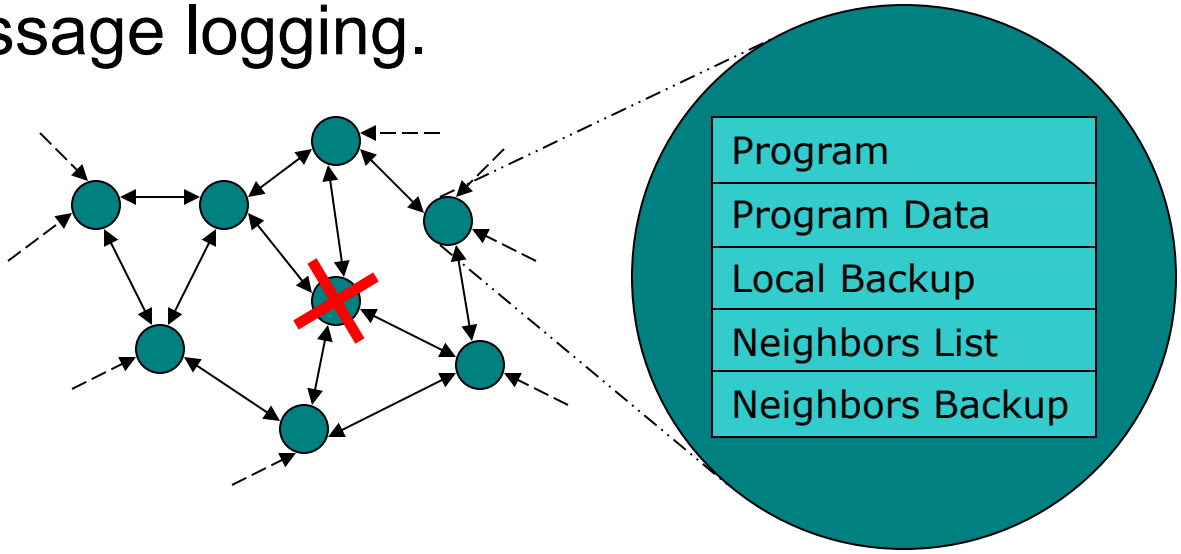
- For non-naturally fault tolerant algorithms.
- Does it makes sense to restart all 100,000 processes because of one failure?
- The mean time between failures (MTBF) is likely to be a few hours or just a few minutes.
- Traditional centralized checkpointing and message logging are limited by bandwidth (bottleneck).

➔ Frequent checkpointing decreases app. efficiency.

➔ The failure rate is going to outrun the recovery rate.

Super-scalable Diskless Checkpointing

- Decentralized peer-to-peer checkpointing.
- Processors hold backups of neighbors.
- Local checkpoint and restart algorithm.
- Coordination of local checkpoints.
- Localized message logging.



Super-scalable Algorithms Research

- Super-scale systems with 100,000 and more processors become reality very soon.
- Super-scalable algorithms that are scale invariant and naturally fault-tolerant do exist.
- Diskless peer-to-peer checkpointing provides an alternative to natural fault-tolerance.
- A lot of research still needs to be done.

Office of Science
U.S. DEPARTMENT OF ENERGY

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

Super-Scalable Algorithms for Computing on 100,000 Processors

Christian Engelmann and Al Geist

Network and Cluster Computing Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory, Oak Ridge, USA

International Conference on Computational Science (ICCS) 2005
Emory University, Atlanta, USA