Massively Parallel Computing on Peer-to-Peer Networks

Team Timeout
Jon Ludwig
Prashant Gahlowt
Young Suk Moon

Overview

- Application
- Network
- Peer-to-Peer Model
- Pastry
- Goals

Application Characteristics

- Computing tasks can be highly parallelized
  - Minimal serial requirements
  - Data can be efficiently divided up and reduced
- Several independent tasks / many instances of the problem
- Each node participating in the computation is interested in the result

Example Application

- Distributed rendering of randomized fractal images
  - Render slices of fractal images in parallel
- No dependence between pixel values (highly parallelized)
- Many possible variations on parameters (independent tasks)

Network Model

- Variable number of compute nodes
- Large amount of data
- All producers of data are also consumers
- Compute nodes will often join and drop out of the network
- Decentralized and self-organizing
- No access to high performance dedicated nodes

Load Sharing

- Minimize:
  - Idle cycles
  - Recovery time from faults
  - Communication overhead
  - Lost data
  - Redundant communication

Structured Peer-to-Peer

- No permanent or central servers
- Little distinction between client and servers
- Each peer does not need to maintain a connection to many of the other peers
- Peers can route messages to any other peer on the network through a series of hops
- Overhead scales logarithmically as peers join

Advantages of Peer-to-Peer

- Little or no access to dedicated servers is needed
- Reduces the bottleneck of centralized servers
- Increases fault tolerance by eliminating the single point of failure
- Capacity increases with the addition of new nodes

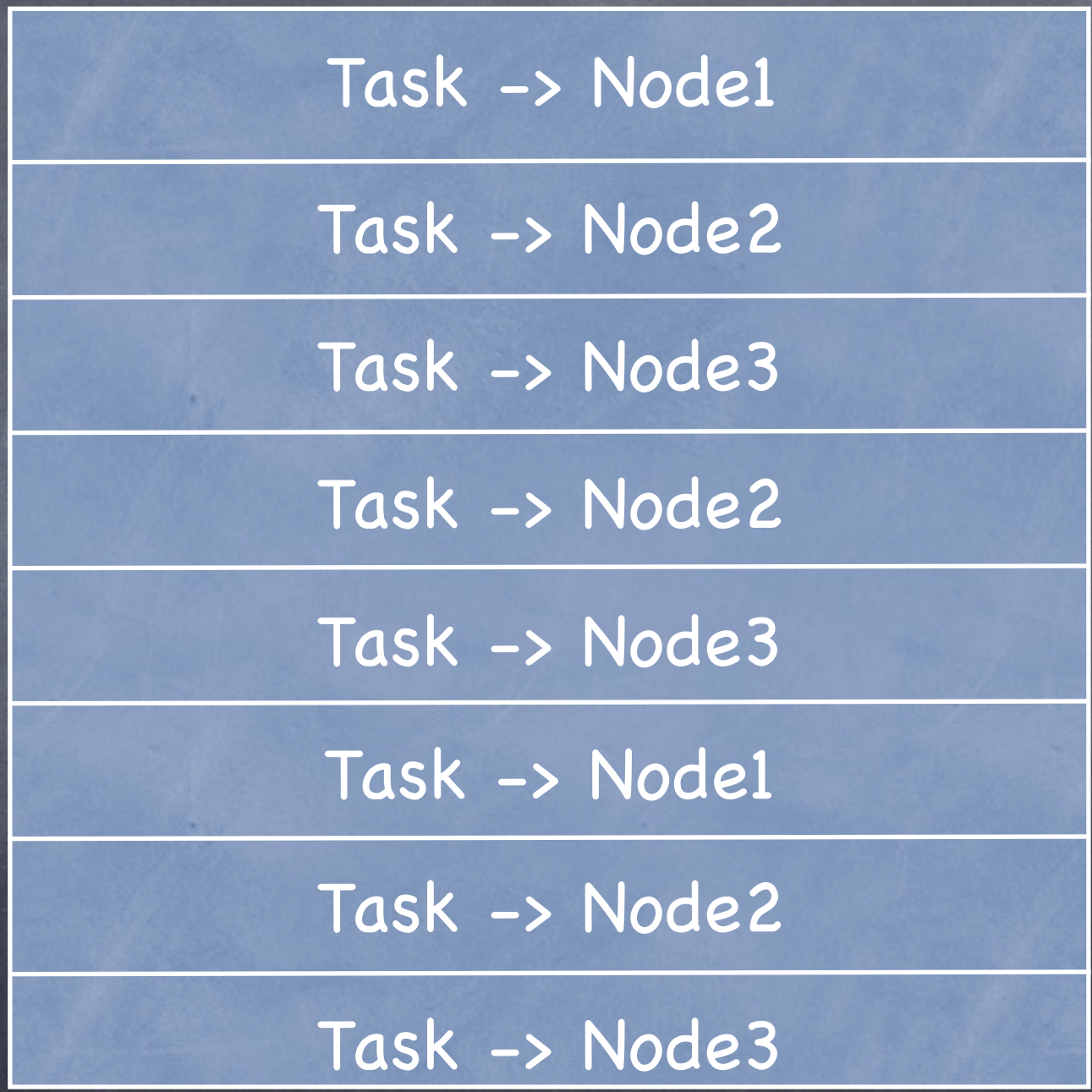
Our Approach

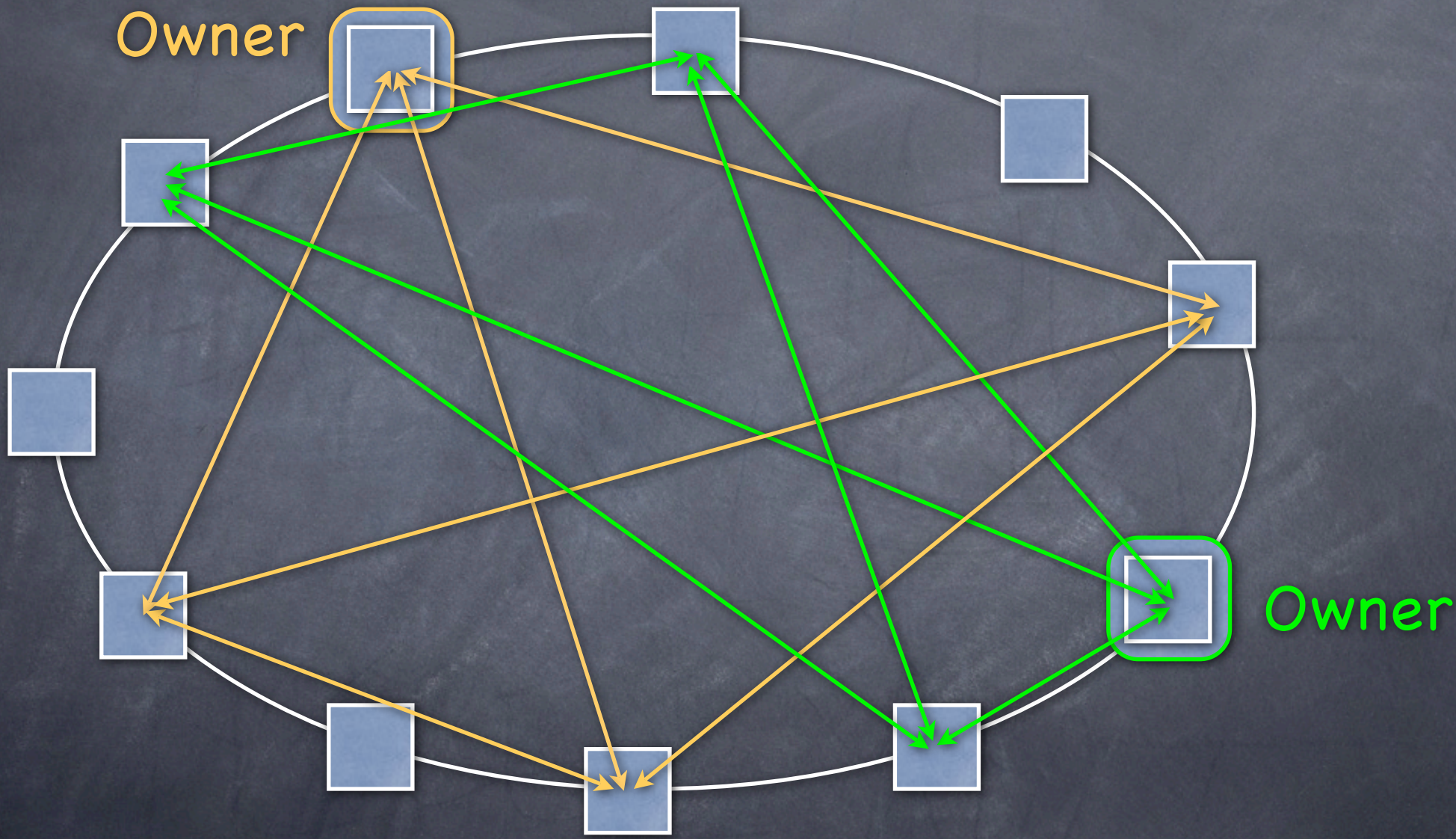
- Create a structured P2P network for efficient communication
- Any node may create and distribute a job
- All nodes maintain a queue of task to process
- Nodes which own a job distribute tasks to peers

Our Approach

- If a node drops out, its tasks are reassigned to other nodes
- If the owner of a job drops out a new owner is negotiated autonomously
- Redundant copies of completed work
- When nodes complete a task their work is distributed to others working on the same job

Job -> Node1


Task -> Node1
Task -> Node2
Task -> Node3
Task -> Node2
Task -> Node3
Task -> Node1
Task -> Node2
Task -> Node3


Owner
Owner

Pastry

- Its a generic P2P Object location and Routing scheme.

Pastry Features:

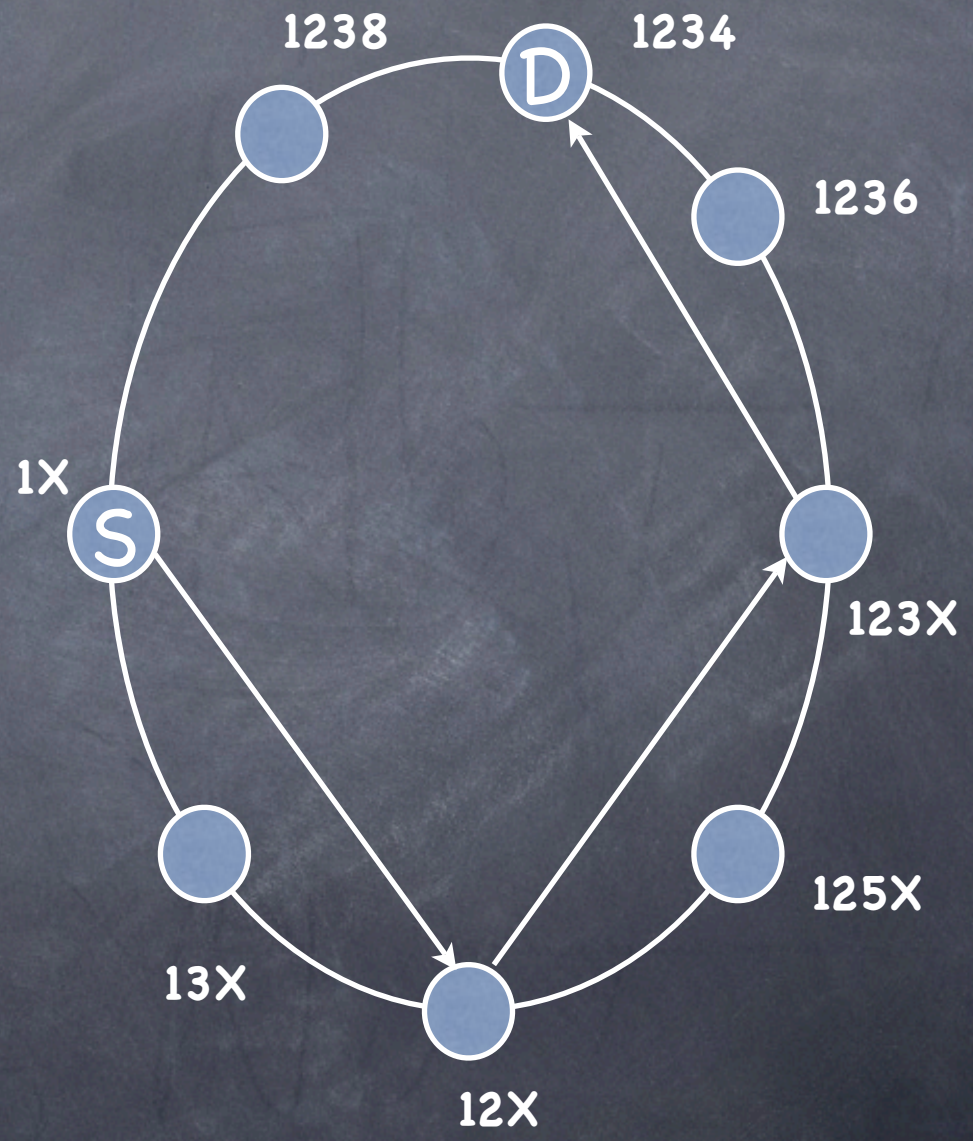
- A self organizing Overlay Network of nodes.
- Completely Decentralized.
- Scalable.
- Reliable and Fault Resilient.

Pastry Design:

- A Node to join the network, sends a request to a random node in the set of "Live Nodes".
- Each Node has a unique 128-bit NodeID, assigned randomly (eg. SHA of IP Address).
- Adjacent NodeID Nodes could be geographically apart.
- Each Node maintains a "Routing Table", "Neighborhood Set" and a "Leaf Set".
- Each Message to be routed has a Key.

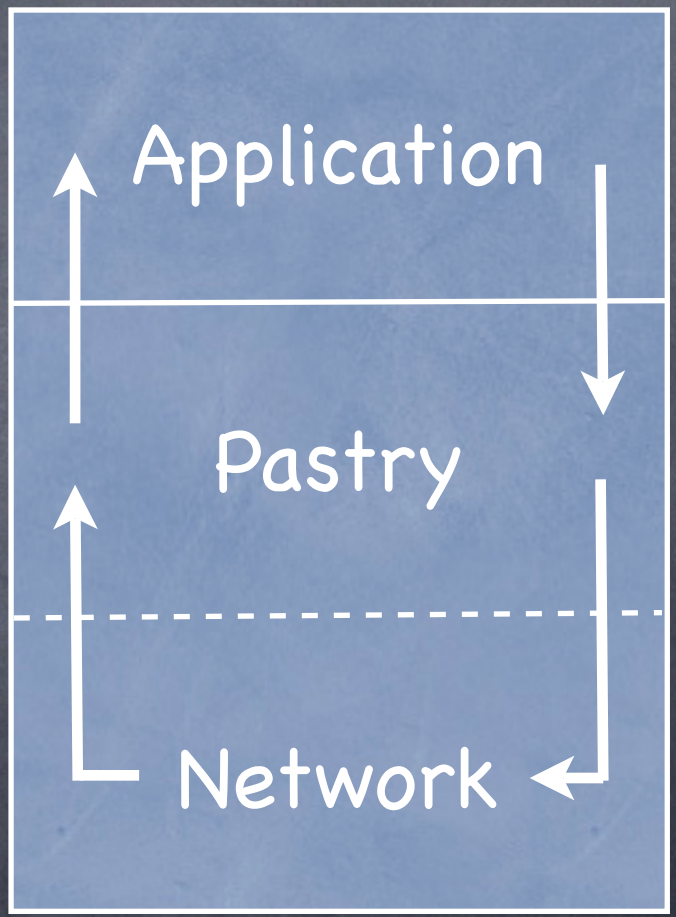
Pastry Routing:

- Routing Algorithm is executed as soon as a Message arrives to a Node.
- Step 1: Check if the key is in the range of Leaf Set, if yes, destination found. If not...
- Step 2: Forward the message to a Node with shared prefix that is longer in one than the current Node.
- Step 3: In case if no such Node is found, forward the message to with at least a shared prefix but numerically closer.
- **The procedure always CONVERGES**



Software Deliverable:

- A Java GUI Application
- Pastry Overlay at Middleware
- Network



Goals

- Fault Tolerance
- Performance
- Scalability
- Decentralization

References

- A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems". IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, pages 329-350, November, 2001.

- D. Caromel, A. Costanzo and C. Mathieu, "Peer-to-peer for computational grids: mixing clusters and desktop machines". Parallel Computing, Volume 33, Issues 4-5, Large Scale Grids, May 2007, Pages 275-288.

- M. Castro, P. Druschel, A-M. Kermarrec and A. Rowstron, "Scalable Application-level Anycast for Highly Dynamic Groups", NGC 2003, Munich, Germany, September 2003.

- D.S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu, "Peer-to-Peer Computing". HP Labratories, Palo Alto, March, 2002.