Creating a Robust Desktop Grid using Peer-to-Peer Services

Alan Sussman
Department of Computer Science and UMIACS

UNIVERSITY OF MARYLAND
Desktop Grid Computing

Growth of Internet (Internet Worlds Stats)

Years

Number of Users (millions)
Confluence of P2P and Grid

Robustness, Reliability and Scalability?
Hard Problems / Issues

• Submitting jobs
• Finding a resource that meets the minimum resource requirements of a job
• Load balancing
• Resilience to failure
System Architecture

1. Insert Job J
2. Route Job J
3. Initiate Matchmaking Algorithm
4. Find Job J
5. FIFO Job Queue
6. Return Job J
Workload Assumptions

- **Must** accommodate heterogeneous clusters of nodes running heterogeneous batches of jobs
- **Clustering** in nodes (resource capabilities) and jobs (requirements)
  - A small number of equivalent classes of nodes
  - Parameter sweeps, e.g., N-body or weather simulations

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Jobs</th>
<th>Clustered</th>
<th>Mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clustered</td>
<td></td>
<td></td>
<td>Condor</td>
</tr>
<tr>
<td>Mixed</td>
<td></td>
<td>BOINC/SETI@Home</td>
<td></td>
</tr>
</tbody>
</table>
Goals of Matchmaking Algorithms

• **Low overhead**
  – Routing must not add significant overhead

• **Completeness**
  – A valid assignment of a job to a node must be found if such an assignment exists
    • TTL-based mechanisms are not applicable

• **Precision**
  – Resources should not be wasted

• **Load balance**
  – Distribute load across multiple candidates
Basic Assumptions

• Underlying *Distributed Hash Table* (DHT)
  – Object location and routing in P2P network
  – Reformulate the problem of matchmaking to one of routing

• *Job* in the system
  – Data and associated profile
  – All jobs are *independent*

• Optimization criterion
  – Minimize time to complete all jobs (combination of throughput and response time)
Modified Content-Addressable Network

• Basic CAN
  – Logical $d$-dimensional space
    • zone, neighbors, greedy forwarding

• Formulate the matchmaking problem as a routing problem in CAN space
  – Treat each resource type as a distinct CAN dimension
  – Map nodes and jobs into the CAN space
    • Resource capabilities and Requirements, respectively
  – Search for the closest node whose coordinates in all dimensions meet or exceed the job’s requirements
Modified Content-Addressable Network

• **Virtual Dimension**
  – Clustering of nodes and jobs
    • Resource capabilities and Requirements
    • Distribution of ownership of a zone and Load imbalance
  – Supplement the *real* dimensions
    • Corresponding to node capabilities
  – Coordinates for nodes and jobs for the virtual dimension generated *uniformly at random*
Improving CAN-based Algorithm

- Employing Dynamic Aggregated Resource Information (HPDC’07)

- Aggregate Resource Information
- Choose the least loaded direction
- Push a job into under-loaded region
- Stop pushing
- Choose the best run node

Diagram:
- Memory Dimension
- CPU Dimension
- Node A
- Node B
- Job
- Routing
Rendezvous Node Tree

- *Implicit* tree built on top of P2P network
  - 1-1 mapping from DHT (*Chord*) nodes to RN-Tree nodes

- Why use a tree?
  - Need to *aggregate* current resource information to perform matchmaking
  - Aggregated Resource Information
    - *Maximal* amount of each resource available at some node in the subtree rooted at a node
Results from Simulations (Grid 2006)

• CAN and RN-Tree algorithms balance load almost as well as centralized algorithm
  – with low overhead (few messages)
• Overall, the CAN algorithm produces significantly lower wait times than RN-Tree for most workloads
  – with comparable overhead
  – and with dynamic aggregate load info, CAN is better for all workloads
Current Status

• Resource discovery algorithms thoroughly simulated and verified

• CAN-based implementation ongoing
  – Basic CAN services working – node join, leave, job assignment
  – Basic CAN matchmaking working
    • Enhanced with dynamic aggregate load info under way
  – Basic authentication mechanism for hosts and jobs in place, based on certificates and public-key authentication
  – Job management and GUI client interface under development
Future Work

• Deploying the prototype system for real workloads and real machines

• Better characterization of real workloads
  – via consultation with Astronomy collaborators, and automated mining of Condor system logs
The Project Team

• Faculty members
  – Alan Sussman, Pete Keleher, Bobby Bhattacharjee, Derek Richardson (Astronomy), Dennis Wellnitz (Astronomy)

• Prototype implementation
  – Michael Marsh, Beomseok Nam

• Matchmaking algorithms and simulations
  – Jik-Soo Kim

• Project funding from NASA and NSF
  – to develop algorithms, and build and deploy the system