My Cray can do that?
Supporting Diverse Workloads on the Cray XE6

Shane Canon
Lavanya Ramakrishnan
Jay Srinivasan

Lawrence Berkeley National Lab

May 1, 2012
We are in the midst of a Scientific Data Explosion.
Cost per Genome

Moore’s Law

Two Year Period

Logarithmic Scale

Source: National Human Genome Research Institute
MATERIALS PROJECT

A Materials Genome Approach

Accelerating materials discovery through advanced scientific computing and innovative design tools.

Materials Explorer
Search for materials information by chemistry, composition, or property.

Lithium Battery Explorer
Find candidate materials for lithium batteries. Get voltage profiles and oxygen evolution data.

Crystal Toolkit
Convert between CIF and VASP input files. Generate new crystals by substituting or removing species.

Phase Diagram App
Computational phase diagrams for closed and open systems. Find stable phases and study reaction pathways.

Reaction Calculator
Calculate the enthalpy of tens of thousands of reactions and compare with experimental values.

Structure Predictor
Predict new compounds using data-mined substitution algorithms.

Database Statistics

19120 materials
214 intercalation batteries
3050 bandstructures
4158 conversion batteries

Press Highlights

Beyond Fossil Fuels: Finding New Ways to Fill the Tank

Latest News
Common Themes

• Throughput Oriented / Embarrassingly parallel
• Rapidly Increasing demand for computation (outpacing Moore’s Law)
• Often Data Intensive
• Scaling from desktop or mid-range scales
Map/Array Job

Large Input

Filter/Map

Output

Output

Filter/Map

Output

Filter/Map

Output

Filter/Map

Output
Map/Reduce

Large Input

Filter/Map

Filter/Map

Filter/Map

Filter/Map

Shuffle

Reduce

Reduce

Reduce

Reduce

Output
Approaches

• Private/User Allocation
  – Task Farmer
  – MySGE
  – MyHadoop

• Shared
  – CCM/Torque
Private Allocation

“MOM” Nodes

Server

Job Script

Compute Nodes
TaskFarmer

Server
- Portable
- Reads in query genes
- Tracks progress and re-runs failed tasks
- Maintains checkpoint
- Collects output from clients

Client
- Can run any executable or script
- Gathers command line arguments from server
- Fetches input from server and pushes back results

Parallel Job

Server (can run anywhere)

Client Tasks $O(16k)$

Output collected at server

Hopper
Strengths of MapReduce and Hadoop

- Fault Tolerance Model
- Data Locality
- Simple Programming Model
- Hides Complexity
- Domain Specific Extensions
- Strong Community
HDFS Architecture

- Client
  - Metadata ops to Namenode
  - Read from Datanodes
  - Write to Datanodes

- Namenode
  - Metadata (Name, replicas, ...): /home/foo/data, 3, ...

- Datanodes
  - Replicates Blocks
  - Block ops

- Replication

- Rack 1

- Rack 2
Data Flow in MapReduce

Node 1
Mapper

Node 2
Mapper

Node N
Mapper

Node 1
Reducer

Node 2
Reducer

Node N
Reducer

Map

Shuffle

Reduce
• User submits a single parallel job
• Personnel SGE scheduler is started
• User can submit jobs to SGE without modifications
• User still needs to think about scaling issues
Common Challenges and Solutions

• Name Services (i.e. passwd/ldap)
  – PRELOAD library to trap getpw* calls

• Gathering Hostnames
  – (Could probably get this from ALPS)
  – Use aprun to gather nids and generate host list

• Master service runs on “mom” node
Load imbalance can lead to wasted cycles and additional charging
Other users can’t take advantage of idle cores
Running a shared-node Serial workload on the XE-6 using CCM
• CCM can be used to “convert” XE-6 (MPP) compute nodes into standard “cluster-like” nodes with a regular Linux environment.
• To run a serial workload on these “CCM nodes” requires they be accessible as regular cluster nodes to the batch system
  – This cannot be done using the regular batch system
  – This requires starting up a separate batch system instance
  – Done using a special CCM “job” which starts up the server and client daemons – the server is started up on the standard XE-6 MOM nodes, and the clients are on the XE-6 CCM compute nodes
Mechanics of running a shared-node serial workload

- “Special” user submits a job to the ccm_queue, asking for as many nodes as required to handle a serial workload (subject to CCM limits), and for the maximum time allowed.
- “Special job” starts up pbs_server on XE-6 MOM node with alternate ports
- Job then runs pbs_mom on allocated CCM compute nodes (under alternate ports)
- Job starts up scheduler (Maui or pbs_sched) which communicates with the alternate resource manager (RM)
- At this point, other users (user1, user2, etc) can submit jobs to the CCM compute nodes (which have now been essentially repurposed as a separate cluster supporting a serial workload)
<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSX</th>
<th>Req'd Memory</th>
<th>Req'd Time</th>
<th>S Time</th>
<th>Elap S Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>nido00002:35000</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22129</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:04</td>
</tr>
<tr>
<td>nido00008/0</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22132</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:04</td>
</tr>
<tr>
<td>nido00008/1</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22135</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:04</td>
</tr>
<tr>
<td>nido00008/2</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22138</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:04</td>
</tr>
<tr>
<td>nido00008/3</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22153</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:04</td>
</tr>
<tr>
<td>nido00008/4</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22199</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:03</td>
</tr>
<tr>
<td>nido00008/5</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22233</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:03</td>
</tr>
<tr>
<td>nido00008/6</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>22284</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>00:03</td>
</tr>
<tr>
<td>nido00008/7</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26161</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
<tr>
<td>nido00008/16</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26166</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
<tr>
<td>nido00008/17</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26186</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
<tr>
<td>nido00008/18</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26198</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
<tr>
<td>nido00008/19</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26239</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
<tr>
<td>nido00008/20</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26288</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
<tr>
<td>nido00008/21</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26332</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
<tr>
<td>nido00008/22</td>
<td>jay</td>
<td>serial</td>
<td>tst.job</td>
<td>26394</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>00:10</td>
<td>R</td>
<td>--</td>
</tr>
</tbody>
</table>
### Active Jobs

<table>
<thead>
<tr>
<th>JOBNAME</th>
<th>USERNAME</th>
<th>STATE</th>
<th>PROC</th>
<th>REMAINING</th>
<th>STARTTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>50</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>51</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>52</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>53</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>54</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>55</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>56</td>
<td>canon</td>
<td>Running</td>
<td>1</td>
<td>00:00:00</td>
<td>Mon Apr 30 09:36:34</td>
</tr>
<tr>
<td>41</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
<tr>
<td>42</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
<tr>
<td>43</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
<tr>
<td>44</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
<tr>
<td>45</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
<tr>
<td>46</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
<tr>
<td>47</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
<tr>
<td>48</td>
<td>jay</td>
<td>Running</td>
<td>1</td>
<td>00:08:58</td>
<td>Mon Apr 30 09:35:32</td>
</tr>
</tbody>
</table>

16 Active Jobs  16 of  24 Processors Active (66.67%)
1 of  1 Nodes Active   (100.00%)  

### Idle Jobs

<table>
<thead>
<tr>
<th>JOBNAME</th>
<th>USERNAME</th>
<th>STATE</th>
<th>PROC</th>
<th>WCLIMIT</th>
<th>QUEUITIME</th>
</tr>
</thead>
</table>

0 Idle Jobs

### Blocked Jobs

<table>
<thead>
<tr>
<th>JOBNAME</th>
<th>USERNAME</th>
<th>STATE</th>
<th>PROC</th>
<th>WCLIMIT</th>
<th>QUEUITIME</th>
</tr>
</thead>
</table>

Total Jobs: 16  Active Jobs: 16  Idle Jobs: 0  Blocked Jobs: 0
Limitations

• Current approach uses a static assignment of nodes.
  – Initial request for CCM nodes needs cannot be changed on the fly, but multiple requests can be made

• CCM communication occurs over TCP/IP, so the high-performance network is not available. (Can’t share uGNI)

• Zhengi Zhao/Helen He’s presentation on CCM for other uses of CCM and some of the limitations
Future Work

• Continue to Improve CCM/Torque Approach
  – Finish testing and phase into production
  – Dynamically resize serial partition

• Improve Hadoop Implementation
  – Optimize shuffle phase for high-bandwidth network
Suggested Cray Optimizations

• Local storage (SSD)
  – Many applications and frameworks rely on local storage
  – Useful for Data Intensive Apps

• Improvements to CCM/DVS
  – Tools to facilitate running Python/Perl at scale
  – Tools to help caching data at scale

• Improvements to ALPS

• Better ways to cleanup after a job
• Increasing demand to support new workloads
  – Driven by improving instruments
  – New classes of modeling and simulation
• NERSC has developed four approaches to supporting new workloads
• The Cray platform is surprisingly flexible:
  – x86/Linux underpinnings help
  – CCM and other extensions have further simplified matters
Lavanya Ramakrishnan

Jay Srinivasan

This work was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
Thank you!

Contact Info:
Shane Canon
SCanon@lbl.gov