Computational Clustering

Jesse Becker

Northwestern University
McCormick School of Engineering

3 October 2006

Introduction

“Begin at the beginning,” the King said gravely, “and go on till you come to the end: then stop.”

--Lewis Carroll
Alice's Adventures in Wonderland

Me

- Work for the Engineering School at NU.
- Support for several computational research groups (in addition to IT infrastructure)
  - Molecular dynamics
  - Complex systems
  - Finite element Analysis
- Local Red Hat Network proxy
- On-campus software mirror
- Unix stuff...

Stuff in the queue

- Background / History
- Cluster planning and installation
- Using / Queuing

Types of clustering

- High Availablity
  - Failover/redundancy
  - Linux-HA project
- Load Leveling
  - Focus on job throughput
  - Good for workstations (Condor project)
- Beowulf clusters
  - Complicated / big jobs
  - What we're talking about today

Short history

- Supercomputing in the old days (pre-1994)
  - IBM Mainframes
  - Crays
  - EXPENSIVE
  - Gov't and Gov't related use
    - Weather
    - Nuclear weapons testing
    - Scienctific simulations (physics!)
    - Cryptography
    - *etc*

Short history (cont.)

1993...

- Thomas Sterling and Donald Becker working at Goddard Space Flight Center (MD).
- Idea for COTS system
  - Cheap networking (Ethernet)
  - Cheap Unix OS (Linux—DB wrote network drivers!)

...1994

- 16 node cluster online “Wiglaf”
  - Speed demon: 66Mhz 486DX4 processors
  - $40,000

History: recent past

- Whole industry developed.
- Rack mount hardware over workstations
  - Workstations still around though…
- Cluster-in-a-box / turn-key systems
- Small clusters are “easy”
- Big clusters are hard (and expensive)

Current Clusters

Small…Aluminum(?)

- Hydra
  - 32 node, dual 2.6GHz Xeon, 2GB RAM/node
- Caramulo
  - 28 node, dual 2.6GHz Opteron, 4GB RAM/node
- Nutzy
  - 4 node, 500Mhz PII

Big Iron

- ASC Purple
  - Sandia NL, 12,544 POWER5 chips, AIX, 7.5 MW of power, 16M BTUs
- Blue Gene/L
  - IBM, 65k PPC CPUs, AIX/Linux
- Thunderbird
  - Sandia, 4512 Dell 1850s

Current Clusters

- Super Computers

ASC Purple

- Non-Super Computers

Hydra and Cusask

Nutzy

Preperation

Things to know pre-install

1. Understand your problem!
2. Know your code
   - Memory
   - Network
   - CPU
   - IO
3. 80% of time is spent in 20% of the code

Choices: Hardware

- Same hardware is nice
  - "Similar" is okay.
  - Mixed clusters are possible, but harder
    - Need a good job scheduler
- Replacements
  - Same hardware makes replacement easy
- Buy good hardware

CPU: AMD vs. Intel

AMD

- Better memory bandwidth (hypertransport)
- Cheaper (?)

Intel

- Faster *raw* number crunching
- Limited memory bandwidth (CPUs shared bus)

Memory

- More memory == good
- Swap == *very bad*
  - As soon as you start swapping, performance tanks

Disk

- Slowest part of the system (10^{-9} sec *vs* 10^{-3} sec)
- Slow IO can cripple a cluster
- RAID
  - Absolutely required
  - RAID 10 if possible
  - RAID != backup

Network

- 2nd slowest part of the system
- GigE
  - Cheap / Easy
  - Latency is awful
  - NIC / Switch makes a huge different
    - Tune settings – Intel cards are good for this
- Inifiniband / Myrinet
  - Better latency / bandwidth
  - Double cost of a node
  - Still need a management network…

Remote access

- KVM
  - Very handy
- KVM over IP
  - Expensive, but handy
- Serial console

Environmental

- Cooling
  - 1-2 tons of AC/rack
  - 6 tons for blades
    - 1 ton = 12,000BTU
- Power
  - 400W per node…32 nodes = 14KW…
- Security
- “Environmental” cost is half the total cost

Design…

- Network architecture
- IO systems / Storage
  - Backups
- User management
  - Resource limits
  - Quotas (disk/CPU)
  - Accounting
- Queuing

Installation!

Frontend

- Frontend / Head node / Management node
- Controls rest of the cluster
  - User management
  - Queue management
- Frequently has primary data storage
- Application exports

Frontend install issues

- Like a standard server install
- Base system
  - Userspace tools
    - Development stuff (gcc, gdb, icc)
    - Editors, analysis tools, etc
  - Shared applications (Matlab, MD, *etc*)
  - Security (firewalls, private network, etc)
    - Package updates?
- Storage (quotas)
- User accounts (resource access)

Compute Nodes

- Actually do the work
- Installs should be automated
  - Or at least cloneable…
- Scalable install/configuration method is key.
- Config management after install?
  - Cfengine, *et al*
  - Do we care? Reinstall!

Compute Node Install Methods

Image Installs

- “Golden Master”
- Easy to create

```
cat /dev/hda > disk.img
```

- Hard to change
- What about different hardware?

Metadata Installs

- Care about configuration, not specific files
- Hard to create
- Easy to manage
- Handles different hardware

Compute node install issues

- First few times are iterative
  1. Configure
  2. Install
  3. Test
- Things to consider
  - Partitoning
  - Software packages / configuration
  - System time
  - Kernel settings
  - User distribution?

ROCKs

ROCKS Cluster distribution

- From San Diego Supercomputing Center at University of California at San Diego
- Full time staff (at least three)
- Built of CentOS
- ***Heavy*** use of kickstart installs (and RPM)
- Flexible
- Active mailing list and wiki
- Full MPI support, Intel compilers, other goodies

ROCKS install

- Architechures: x86, x86_64, ia64
- Supports ethernet, Myrinet, Inifiniband
- Modest hardware requirements:
  - Head node:
    - 20GB disk
    - ~800MB RAM
    - 2 ethernet ports

Compute node
~6GB disk
512MB RAM
ethernet port

Customization

- Modular install using “Rolls”
- A few base rolls (kernel, OS, webserver, etc)
- Collection of semi-related packages
- Job-specific rolls
  - Java
  - Condor
  - Bioinformatics
  - Visualization

Cluster Administration

- Centralized user administration via 411
  - 411 is a secure file distribution system
  - Simpler than NIS, more resilient, scales better
- MySQL to store some information
- XML files to store compute node configs.
- Easy to change
  - Add packages
  - Set config files
  - Kernel tuning

Example customizations

XML file (abbreviated)

```
<kickstart>
<description>
  extend-compute.xml: Local customizations to compute.xml
</description>

<package> subversion           </package>
<package> fftw                 </package>
<package disable=”1”> sendmail </package>

<post>
  <file name="/etc/ntp.conf">
  restrict 10.1.1.1 mask 255.0.0.0
  broadcastclient
  authenticate no
  </file>

  chkconfig ntpd on

</post>
</kickstart>
```

Node installation

- Compute nodes boot off CDROM or PXE
- Fetch ks.cfg from head node via HTTP
- Starts *anaconda* (the redhat installer)
  - Partitioning
  - Installs RPM packages
  - %post section
- Reboots
- (about 12 minutes)

Queueing

“Garbage in, garbage out.”

--Traditional
(maybe Charles Babbage)

Why do we need a queue?

- In a perfect world, don't need it
  - Infinite resources
  - People are nice
- In the real world...
  - Resources are limited
  - Lots of people want them
  - People aren't nice

Queuing is a hard problem

- Can't make everyone happy all the time.
- Try to be equal and fair
  - Some things are more equal than others
  - Different purchase contributions
  - Some projects more important than others
- Cheaters...

Parts of a queue (1/2)

- Scheduler
  - Sorts the jobs
  - Manages resource access / permissions
  - Accounting
  - What the users complain about.
    - “why isn't my job running?”

Parts of a queue (2/2)

- Dispatcher
  - Sends jobs to compute nodes
  - Daemon on nodes
  - Runs jobs
  - Provides runtime environment
    - LD_LIBRARY_PATH
    - License file locations
    - What about *stdin, stdout,* and *stderr*?

Queuing software

- Direct logins
  - Bad idea
- atd/batch
  - Probably installed, very basic
- GNU Queue
  - Basic queuing, not as flexible as alternatives
- OpenPBS
  - Common in .edu
- Sun Gridengine
  - Best option?

Sun Grid Engine

- Open Source (but you can pay if you want)
- Handles scheduling, dispatching, accounting
- Under active development
- Runs on most Unix systems, and most architectures
- Scales to many thousands of jobs.

Using SGE

- All jobs are shell scripts
- SGE exports certain information (Job ID, hostname, etc) to the job
- Use `qsub` to submit jobs
- Use `qstat` to check on job status

Questions?
(and links)

- http://www.phy.duke.edu/~rgb/Beowulf/beowulf.php
- http://www.beowulf.org/
- http://www.rocksclusters.org/
- http://gridengine.sunsource.net/
- http://www.samag.com/documents/s=8817/sam0313c/0313c.htm
- http://www.cs.wisc.edu/condor/
- http://oscar.openclustergroup.org/
- http://dirk.eddelbuettel.com/quantian.html