Condor at EPFL

The Greedy Desktop Grid

Pascal Jermini & Michela Thiémard

EPFL-DIT-EX

April 21, 2009

Outline

1. Introduction
   - EPFL Resources
   - Looking for the right software solution...
2. Condor at EPFL
   - Deploying a grid
   - HPC Backfill
   - Greedy Status
   - Issues
   - Greedy Users
3. Swiss National Grid
4. Conclusion

EPFL Computing Resources

The Swiss Federal Institute of Technology of Lausanne counts about 10 000 people (students + collaborators), with a lot of computing needs.

Centrally available computing resources at EPFL

- Central general purpose HPC clusters *(about 16 TFlops)*
- A supercomputer (Blue Gene/L) *(23 TFlops)*

We require from the users to write *true* parallel code to take advantage of fast interconnects (Myrinet, InfiniBand)

What about “trivially” or “embarassingly” parallel applications?

EPFL Computing Resources

The Swiss Federal Institute of Technology of Lausanne counts about 10 000 people (students + collaborators), with a lot of computing needs.

Centrally available computing resources at EPFL

- Central general purpose HPC clusters *(about 16 TFlops)*
- A supercomputer (Blue Gene/L) *(23 TFlops)*

We require from the users to write *true* parallel code to take advantage of fast interconnects (Myrinet, InfiniBand)

What about “trivially” or “embarassingly” parallel applications?

EPFL Computing Resources

The Swiss Federal Institute of Technology of Lausanne counts about 10 000 people (students + collaborators), with a lot of computing needs.

Centrally available computing resources at EPFL

- Central general purpose HPC clusters *(about 16 TFlops)*
- A supercomputer (Blue Gene/L) *(23 TFlops)*

We require from the users to write *true* parallel code to take advantage of fast interconnects (Myrinet, InfiniBand)

What about “trivially” or “embarassingly” parallel applications?

EPFL Computing Resources

The Swiss Federal Institute of Technology of Lausanne counts about 10 000 people (students + collaborators), with a lot of computing needs.

Centrally available computing resources at EPFL

- Central general purpose HPC clusters *(about 16 TFlops)*
- A supercomputer (Blue Gene/L) *(23 TFlops)*

We require from the users to write *true* parallel code to take advantage of fast interconnects (Myrinet, InfiniBand)

What about “trivially” or “embarassingly” parallel applications?

Desktop resources at EPFL

Many desktop machines are available at EPFL:

- More than 800 machines in student classrooms
- More than 3 000 machines belonging to the staff

Most of them are not used during the night and the weekend.
This is a lot of potentially unused CPU cycles!

Our goals

- Federate and consolidate all these unused cycles to create a low-cost “cluster”
- Provide a simple (or at least unified) access to the desktop resources

Desktop resources at EPFL

Many desktop machines are available at EPFL:

- More than 800 machines in student classrooms
- More than 3 000 machines belonging to the staff

Most of them are not used during the night and the weekend.
This is a lot of potentially unused CPU cycles!

Our goals

- Federate and consolidate all these unused cycles to create a low-cost “cluster”
- Provide a simple (or at least unified) access to the desktop resources

Desktop resources at EPFL

Many desktop machines are available at EPFL:

- More than 800 machines in student classrooms
- More than 3 000 machines belonging to the staff

Most of them are not used during the night and the weekend.
This is a lot of potentially unused CPU cycles!

Our goals

- Federate and consolidate all these unused cycles to create a low-cost “cluster”
- Provide a simple (or at least unified) access to the desktop resources

Middleware Requirements

- Least intrusive as possible on the compute nodes
- Compute nodes can easily enter and leave the pool
- Reasonable data security on compute nodes
- Multi-platform: Architecture and Operating System
- Has job accounting and job history
- Can work with centralized submission servers
- Compatible with “standard” grid middlewares (integration into a wider grid?)
- Free (as in no cost) if possible

CONDOR!!

Middleware Requirements

- Least intrusive as possible on the compute nodes
- Compute nodes can easily enter and leave the pool
- Reasonable data security on compute nodes
- Multi-platform: Architecture and Operating System
- Has job accounting and job history
- Can work with centralized submission servers
- Compatible with “standard” grid middlewares (integration into a wider grid?)
- Free (as in no cost) if possible

CONDOR!!

Challenges

Since the machines are not centrally managed, it is not easy to deploy a Campus Grid from a central location.

- Not all classrooms administrators are willing to install Condor
- Fear of unknown software running on their PCs
- Ecological excuse
- We have to accomodate n different methods of classroom deployments

This can be solved by appropriate administrators “education”:

- Live demos to prove that Condor is harmless
- Central directive stating that all PCs bought with central IT credits must be part of the Grid

Our current configuration

Grid effort at EPFL started in 2006, with the creation of the "Greedy Pool"

Central servers

- One Central Manager, also hosting checkpoints and accounting databases
- One submit host, hosting home directories for users
- Quill is being used for accounting data collection
- No shared filesystem: all data transferred through the Condor file transfer mechanism (almost impossible to have a common filesystem in an heterogeneous environment!)

Our current configuration II

Compute nodes

- Jobs can start only between 8pm and 7am and 24h/24 during the weekend
- The machine's owner has always the priority over Condor jobs
- An already running job can continue its execution if the machine's owner is not back at 7am
- If the owner comes back, the job is suspended for 10 minutes, and then killed if the owner stays more than that
- Another grid user cannot preempt an already running job, even if the user has more priority
- Communications between compute nodes and submit host secured with SSL

Our current configuration III

User policies

- Relatively short jobs: 3-4 hours, not enforced but we keep an eye on it
- No debugging, only production code
- Try to minimize data transfers
- Single-processor jobs only: no MPI, no OpenMP

Tools developed at EPFL

Besides basic and advanced documentation for the users, we developed a few in-house tools for reporting and to help resource managers:

- Web-based "`condor_config.local`" generator
- Statistical data collection

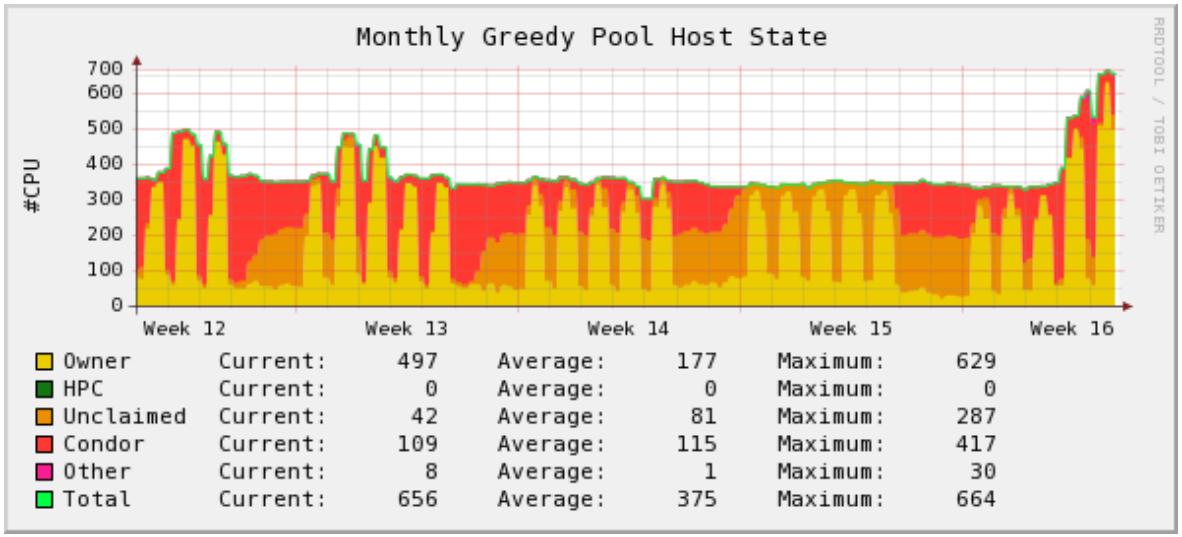
Configuration file generator

- Helps users to configure time-slots for when Condor can run jobs on their machine (if they want to customize that)
- Also used to configure correctly the custom attributes to be advertised by Condor
- Custom attributes are used for "non-standard" software installed on the node (Matlab, Mathematica, . . .)
- We advertise the path and the version of these softwares

Statistics

We have developed two tools to keep track of usage of the pool:

- Graphical overview of job queue, status of the pool (by Operating System, Architecture, current state of the compute nodes)
  - Data is gathered with `condor_q` and `condor_status`, parsed and fed to an RRD database
  - Graphics are generated with RRD for the following periods: last day, last week, last month and last year
  - Example: hosts state for the last month:



Statistics II

- Detailed historical statistics of jobs that have run on Greedy
  - Jobs repartition by user, Operating System, Architecture
  - Data is gathered from historical data stored by Condor in a Postgres database with Quill
  - Statistical data is computed and stored in a compact form into a mySQL database. HTML output is then generated daily
  - Example: monthly usage of the grid, by operating system:

Operating System	Number of runned jobs	Walltime					
		%	Total	Minimum	Mean	Maximum	Standard deviation
WINNT51	5034	85.84	29265:50:00	00:00:36	05:48:49	81:58:56	06:17:07
LINUX	5918	10.33	3520:11:40	00:00:21	00:35:41	03:47:45	00:23:19
OSX	905	3.67	1252:02:00	00:00:47	01:23:00	74:55:27	02:33:07
WINNT52	67	0.15	52:30:43	00:01:04	00:47:01	02:07:31	00:21:43
WINNT60	2	0.00	01:27:49	00:38:58	00:43:54	00:48:51	00:06:59

Compute node deployment

We offer custom-made Condor installation packages for all our supported platforms that:

- contain an EPFL-tailored `condor_config` file
- prepare SSL certificates requests (users just have to submit the request via a web-site, and he will receive back the signed certificates by email)
- set-up automatic updates of Condor binaries and configuration from a central repository

And for mass-deployments in classrooms:

- customized installers to accomodate the local classrooms deployments procedures (it's often on a case-by-case basis)
- we provide support for configuring Condor for classrooms with special needs

Backfilling HPC clusters

Why?

- The scheduler of the cluster may leave “holes” in the jobs schedule
- The scheduler may be down for some reasons, and the cluster is essentially idle (let’s take advantage of this!)

Challenge:

- Cluster nodes are very often behind a firewall and/or on a non-routable private network

How?

- Generic Connection Broker or Flocking
- Prologues and Epilogues to enable and disable Condor jobs after and before “traditional” HPC jobs
- Always give priority to cluster jobs: we only “scavenge” unused cycles!

Backfilling HPC clusters

Why?

- The scheduler of the cluster may leave “holes” in the jobs schedule
- The scheduler may be down for some reasons, and the cluster is essentially idle (let’s take advantage of this!)

Challenge:

- Cluster nodes are very often behind a firewall and/or on a non-routable private network

How?

- Generic Connection Broker or Flocking
- Prologues and Epilogues to enable and disable Condor jobs after and before “traditional” HPC jobs
- Always give priority to cluster jobs: we only “scavenge” unused cycles!

Backfilling HPC clusters

Why?

- The scheduler of the cluster may leave “holes” in the jobs schedule
- The scheduler may be down for some reasons, and the cluster is essentially idle (let’s take advantage of this!)

Challenge:

- Cluster nodes are very often behind a firewall and/or on a non-routable private network

How?

- Generic Connection Broker or Flocking
- Prologues and Epilogues to enable and disable Condor jobs after and before “traditional” HPC jobs
- Always give priority to cluster jobs: we only “scavenge” unused cycles!

Cluster backfilling at EPFL

- It has been implemented on two centrally-managed clusters
- Uses Generic Connection Broker on a front-end node
- No ill-effects noticed until now
- Gives an effective usage of the clusters near 100%

But. . .

- Temporarily disabled until some other (non-Condor) issues are resolved
- We may offer this as a new service for non-central clusters
- In case of busy clusters, jobs may get evicted very often. Job checkpointing becomes essential

Cluster backfilling at EPFL

- It has been implemented on two centrally-managed clusters
- Uses Generic Connection Broker on a front-end node
- No ill-effects noticed until now
- Gives an effective usage of the clusters near 100%

But. . .

- Temporarily disabled until some other (non-Condor) issues are resolved
- We may offer this as a new service for non-central clusters
- In case of busy clusters, jobs may get evicted very often. Job checkpointing becomes essential

Greedy status

To date, Greedy counts about 660 cores, of which:

- 510 Windows cores
- 25 MacOS X cores
- 125 Linux cores

corresponding to about 800 Gflops.

Since September 1st, 2006, more than 2 500 000 computing hours have been scavenged on Greedy!

Expansion of the pool will continue this summer!

Greedy status

To date, Greedy counts about 660 cores, of which:

- 510 Windows cores
- 25 MacOS X cores
- 125 Linux cores

corresponding to about 800 Gflops.

Since September 1st, 2006, more than 2 500 000 computing hours have been scavenged on Greedy!

Expansion of the pool will continue this summer!

Greedy status

To date, Greedy counts about 660 cores, of which:

- 510 Windows cores
- 25 MacOS X cores
- 125 Linux cores

corresponding to about 800 Gflops.

Since September 1st, 2006, more than 2 500 000 computing hours have been scavenged on Greedy!

Expansion of the pool will continue this summer!

Current Issues

Generally speaking we are very pleased with Condor. We are still suffering from a few not-too-critical issues:

- Memory leaks in the collector. Observation leads to believe that SSL and misconfigured execution hosts are the root cause
- Periodical crashes of DBMSD daemon, probably due to a too big historical database. May need some serious Postgres tuning
- Some instances of Windows jobs not being correctly killed when the owner comes back (still under investigation)

Who uses Greedy? And how?

Users of Greedy come from different scientific fields, mainly:

- Physics: modelling of condensed matter
- Chemistry
- Cryptography: large numbers factorization with Number Field Sieve
- Operations Research: simulation of granular assemblies

Most of them use home-made applications in C or C++. Others use Matlab or Mathematica scripts.

- At the beginning: very slow adoption rate from the users
- Now: job queue always full (> 2000 idle jobs), but only from few users
- Our next challenge: promote more heavily Greedy
- We are open to all EPFL's population

Who uses Greedy? And how?

Users of Greedy come from different scientific fields, mainly:

- Physics: modelling of condensed matter
- Chemistry
- Cryptography: large numbers factorization with Number Field Sieve
- Operations Research: simulation of granular assemblies

Most of them use home-made applications in C or C++. Others use Matlab or Mathematica scripts.

- At the beginning: very slow adoption rate from the users
- Now: job queue always full (> 2000 idle jobs), but only from few users
- Our next challenge: promote more heavily Greedy
- We are open to all EPFL's population

Nation-wide grid effort

A project is under way to provide a grid infrastructure that crosses the boundaries of institutions: a national computing grid.

- Swiss National Grid Association (SwiNG): establishes and coordinates a grid at the Swiss level
- A collaborative project has been started in order to create this infrastructure
- Common middleware for all sites: Nordugrid/ARC
- Provides connector scripts for many local resources management systems, and amongst them Condor
- Proof-of-Concept deployement with Nordugrid/ARC has been successful, with some tweaks to the original ARC code to comply with local EPFL policies
- Full-scale project is underway, scheduled to be completed by March 2010.

In conclusion

Would we choose Condor again if we had the choice?

YES!!

Why?

- Extreme flexibility for the configuration and timeframe definitions
- Was designed exactly for our purpose (cycles scavenging)
- Can interoperate with other middlewares

In conclusion

Would we choose Condor again if we had the choice?

YES!!

Why?

- Extreme flexibility for the configuration and timeframe definitions
- Was designed exactly for our purpose (cycles scavenging)
- Can interoperate with other middlewares

In conclusion

Would we choose Condor again if we had the choice?

YES!!

Why?

- Extreme flexibility for the configuration and timeframe definitions
- Was designed exactly for our purpose (cycles scavenging)
- Can interoperate with other middlewares

More information...

- The Greedy home page (EPFL pool):
  `http://greedy.epfl.ch`
- Mail contact of Greedy administrators at EPFL:
  `grid-admins@groupes.epfl.ch`
- The Swiss National Grid Association (SwiNG) home page:
  `http://www.swing-grid.ch`
- The EPFL central computing facilities home page:
  `http://hpc-dit.epfl.ch`