PBS GridWorks

Urgent Computing Workshop

Scheduling – A Review

Bill Nitzberg
Altair Engineering
April 2007

Who is "Bill Nitzberg"?

CTO, Altair Grid Technologies
Altair Engineering, Inc.

- Ph. D., CS, Univ. of Oregon
  - "Collective Parallel I/O"

NASA – Mgr, Parallel Research

- MPI-2 I/O editor
- Whitney Cluster project lead
- Grid Forum, area director
- OpenPBS development
- NASA's Information Power Grid

Altair Engineering, Inc.

- Chief architect, PBS GridWorks
- Open Grid Forum, board member
- Grid and standards evangelist

Scheduling and Resource Management

PBS GridWorks

- Batch
- Monitor
- Schedule
  - Policy – who's allowed to do what, when, and for how long?
  - Optimization – given policy, make the "best use of resources"
- Provision
- Execute
- Limit
- De-provision
- *Fix*
- Account

Goals of Scheduling

PBS GridWorks

Fundamental scheduling goals are divergent

- User → Complete work as soon as possible
- System Manager → Keep resources 100% utilized
- CIO → Drive enterprise-wide strategic direction (efficiently)

Urgent Computing goals…

- User → Complete work as soon as possible
- System Manager → Complete work as soon as possible with minimal impact to other users
- CIO → Complete work as soon as possible with minimal cost when not in use

Run Job…

PBS GridWorks

Next

- Best effort
- Wait for space
  - Highest job priority
  - Express queue

Now

- Oversubscribe
- Make space
  - Preempt – suspend
  - Preempt – checkpoint
  - Preempt – kill & start over

At

- Specify a time
  - Advance reservations
    - Could also be Next or Now
  - Dedicated time

→ Negative Impact →

Scheduling Mechanisms for Urgent Computing

PBS GridWorks

Best Effort – Normal Usage

- No impact
- Probably the most likely technology to be exploited for Urgent Computing

Wait for Space – Job Priority

- Works with all existing scheduling systems
- Minimal impact
- Maximum waiting time can be modulated by setting other Scheduling Policy limits (like max job walltime)

Now – Oversubscribe

- Schedulers support this, but nobody implements this in practice
  - As both Urgent Job and existing jobs can easily thrash
- Could be a possibility for some types of job mixes

Make Space – Preemption

- Suspend
  - Impact depends on memory use & overall system reliability
  - Urgent job can start immediately
  - Supported everywhere (except IBM Blue Gene)
- Application-level checkpoint
  - Usually minimal data, equivalent to original input file
  - Urgent job can start quickly
  - Only available for select applications
- System-level checkpoint (save entire memory space)
  - Can be “expensive”
    - Amortize checkpoint during run (slowing down all jobs), or
    - Write entire image at checkpoint time (slowing Urgent job start)
  - Existing technology is immature (but improving!)
- Kill & start over
  - Urgent job can start immediately
  - Lost work of killed/requeued job

Specify a Time – Advance Reservations

- Maps well to Urgent Computing scenarios
  - Lead times should allow reservation to fit without preemption
  - Reservation is: start + end + required resources
- Orthogonal to Next and Now
  - Traditional: reserve “1000 cpus and 1 TB memory” from 8a – 10a today
  - ASAP: reserve “…” Next
  - Preemptive: reserve “…” Now
- Support across multiple scheduling systems (though not all)
- Variable impact – may leave large “holes” that cannot be backfilled
- Good choice for co-allocation
  - Works alone or in concert with preemption

Expected Technology Improvements

PBS GridWorks

- Priorities
- Preemption
- Reservations
- Staging
- Provisioning (including VMs)

Expect continued improvements in all of the above areas to naturally evolve over the next few years…

Less Expected Technology Improvements

PBS GridWorks

- System-level Checkpointing – commercial viability unclear (outside niches)
- Deadline-based scheduling
- Speculative execution support
- Ramp-up allocation of resources
  - Get 100 cpus now, add 128 in 20 min, … (Next and next and next …)
- More information modeling
  - Represent choices in requirements and system availability
    - Computing needs tend to vary with available resources
    - E.g., run one big high-fidelity model or many low-fidelity models…
- Programatic interfaces for all of the above
  - To support aggregating resources from multiple independent organizations

Other Thoughts…

- You can always game the system
  - And this may be good enough for not-quite-so Urgent Computing
- Altruistic preemption (at user level, not just CIO level)
  - Allow users to denote their jobs as altruistic, and offer up their resources
- Policies -- What policies are stakeholders willing to accept?
- What about Commercial (licensed) applications?

It's EveryWare!

- Easy to use
- Hard to break
- Do more (with less)
- Keep track and plan

sales@pbspro.com