Predicting Bounds on Queuing Delay in Space-Shared Computing Environments

John Brevik, Daniel Nurmi, Rich Wolski

University of California Santa Barbara
Using HPC Machines

- Scientists previously had access to one or few HPC (High Performance Computing) machines
- Trends in commodity clusters has resulted in more HPC systems
- Grid computing efforts have led to higher degree of accessibility
  - Uniform software infrastructure
  - Easier to be granted access
- Modern HPC user has simultaneous access to many systems
Choices

• Clusters, cycle-harvesting farms, parallel machines, SMP machines, etc
• Differ in significant ways
• Given an application and a pool of HPC systems, which do we choose to get fastest turnaround time?
Factors

- **Architecture**
  - How well does my app perform on this architecture?
- **Data locality**
  - Is my data accessible, quickly, on this machine?
- **Software/Environment**
  - Are the tools my app requires available?
- **Execution delay**
  - What is the delay between when I decide to run my app and when the app actually executes?
  - **Batch Queue Wait Time**
Queuing Delay

- Most HPC sites employ space-sharing to manage workload
  - Batch queue system (PBS, Torque, LoadLeveler)
- Overall application turnaround time = queue delay + execution time
Queuing Delay

Mean = .48
Queuing Delay

• Modern batch queue software provides little if any batch queue wait time estimation
  – Requires perfect knowledge of scheduler, job execution time
  – Requires no cancellation or policy change

• Problem: can we provide **predictions** to help mitigate the effects of queuing delay overhead on overall turnaround time?
Our Approach

• Previous efforts focus on mean predictions
• Provide user with bound predictions, with quantifiable confidence, on queue wait time
  – Often real question is, “How long will my job wait at most?”
• Not an ‘expected wait time’ prediction
• Answer question, “At most, how long will my job wait 95% of the time?”
This Work

• Analyze Data
• Propose Prediction Methodology
• Perform Experiment
• Evaluate Results
• Future Work
Batch Queue Data

• Collected job traces from 10 machines ranging 9 years of HPC (several million jobs)
  – Feitelson Parallel Workloads Archive
  – Current systems (TACC, Teragrid, etc)

• Real time monitoring system currently gathering job data from machines in operation
More Data

365 days cnsidell/ALL

143 days dante/dque

10 days ncsateragrid/dque

27 days tsubame/default
Prediction Methodology

• Questions of form, “At most, how long will the next job wait, Q percent of the time?”
  – Quantile prediction
  – $Q_{th}$ Quantile - Value which $Q$ percent of data points are less than or equal to

• Use our own non-parametric technique to make quantile predictions from historical values: Binomial Method (BM)
Prediction Methodology

• Simple application of BM results in inaccurate results

• Queuing delay fluctuates over time
  – Machine dynamism
  – Big events

• Jobs are not treated equally by scheduler
  – Job characteristics (nodes)
  – Backfilling
Changepoints

Day in the life - UofC Teragrid
Changepoint Detection

• Idea: Use only useful history for making predictions
• Assumption: Underlying distribution changes drastically and infrequently
• Rare event: Consecutive observations above .95 quantile
  – Find improbable number consecutive failures in synthetic data
  – Flag ‘rare event’ when we see same number of consecutive failures in real data
• If encounter rare event, trim history and continue
Grouping Jobs

- Have available more information than just submission time and queue wait time
  - Number of nodes requested
- Queried variety of system operators for ‘reasonable’ requested node ranges
- Settled on
  - 1 - 4
  - 5 - 16
  - 16 - 64
  - 65+
Grouping Jobs

Requested Nodes: 1 - 4

Requested Nodes: 17 - 64
Improved Predictor

• Use Binomial Method
  – Accurate non-parametric quantile predictor

• Introduce changepoint detector
  – Attempt to only use relevant history

• Introduce job clustering
  – Attempt to isolate like jobs

• Binomial Method Batch Predictor (BMBP)
Experiment

- Choose quantile to predict: .95
  - Prediction of upper bound delay a job will experience 95% of the time
- Three methods
  - BMBP
  - Log-normal with history trimming
  - Log-normal without history trimming
- Examine both correctness and accuracy of each method
  - Correct: 95% or more predictions $\geq$ actual wait time
  - Accurate: median ratio of actual wait time over prediction
Results

- **Correctness**
  - 40/68 Log-normal no-trim
  - 59/68 Log-normal with-trim
  - 68/68 BMBP

- **Correct and more accurate**
  - 6/68 Log-normal no-trim
  - 15/68 Log-normal with-trim
  - 46/68 BMBP

- **BMBP correct for all data sets**, significantly **more accurate** than log-normal based methods
Interesting Observations

TACC and Datastar Upper 95% Predictions
Thursday February 24, 2005

345678 seconds = 4 Days

12 seconds

Time (seconds)
Interesting Observations

Datastar 95% Predictions
June 1, 2004 to July 1, 2004, 1-4 and 17-64 Processors
Current Status/Future Work

• Compare against other **parametric** quantile predictors
  – Weibull, hyper-exponential

• We have added **automatic job grouping**
  – Model based clustering
  – Improves accuracy

• Use batch queue wait time predictions for workflow task **resource selection**
  – SC’06 paper

• Online batch queue prediction tools
Thanks

- Next Generation Software (NGS) program
- VGrADS project
- San Diego Supercomputer Center
- http://nws.cs.ucsb.edu/batchq