Predicting batch queue job wait times for informed scheduling of urgent HPC workloads

Nick Brown, EPCC
n.brown@epcc.ed.ac.uk
Urgent computing
VESTEC system: Urgent computing

- The VESTEC system federates over numerous HPC machines
- Run job on which machine is most suited
  - To address limitations in batch queue for this sort of workload
- However accurate job placement is required to make correct choices
Machines used in this work

ARCHER2: HPE Cray EX, 5860 nodes, 314880 jobs in the standard queue and 73472 jobs in the short queue

Cirrus: HPE/SGI 8600 system with 280 nodes, 582200 jobs

4-cabinet: Early ARCHER2 HPC Cray EX, 1000 nodes, 373560 jobs
Slurm’s queue prediction

• Tracked the lifetime of all jobs submitted on ARCHER2 and Cirrus over two week period

| Predictions accurate within | ARCHER2 | Cirrus |
|-----------------------------|---------|--------|
|                            | initial | best   | initial | best |
| 1 minute                   | 5.13%   | 16.55% | 0.42%   | 4.12% |
| 5 minutes                  | 12.47%  | 23.51% | 0.42%   | 4.26% |
| 10 minutes                 | 19.91%  | 30.99% | 0.85%   | 4.40% |
| 30 minutes                 | 41.31%  | 58.25% | 2.69%   | 11.78%|
| 1 hour                     | 58.40%  | 70.25% | 4.40%   | 16.34%|
| 2 hours                    | 69.91%  | 79.17% | 8.38%   | 25.99%|
| 6 hours                    | 81.47%  | 90.59% | 30.11%  | 52.70%|
| 12 hours                   | 90.45%  | 94.39% | 50.85%  | 67.90%|
| 24 hours                   | 95.39%  | 99.31% | 77.98%  | 86.93%|

• Around 83% of jobs updated their estimates, and over 50% have five or more estimates
• Can see ARCHER2 is far more accurate than Cirrus here
Basic KNN model

- Trained a basic K-Nearest Neighbours model to act as a foundation for our work
  - Two versions: *basic* uses only the number of nodes and wall time requested, *temporal* also includes submission time and day.

  | Predictions accurate within | Standard queue | Short queue |
  |---------------------------|---------------|------------|
  |                           | basic     | temporal | basic     | temporal |
  | 1 minute                  | 9.27%     | 22.67%   | 33.53%    | 66.08%   |
  | 5 minutes                 | 28.79%    | 35.53%   | 46.46%    | 78.16%   |
  | 10 minutes                | 39.53%    | 41.37%   | 50.47%    | 83.76%   |
  | 30 minutes                | 52.60%    | 53.32%   | 96.73%    | 92.21%   |
  | 1 hour                    | 64.37%    | 61.85%   | 98.55%    | 96.18%   |
  | 2 hours                   | 72.05%    | 68.68%   | 99.37%    | 97.50%   |
  | 6 hours                   | 83.28%    | 80.70%   | 99.67%    | 99.23%   |
  | 12 hours                  | 88.31%    | 87.40%   | 99.68%    | 99.67%   |
  | 24 hours                  | 94.39%    | 92.68%   | 100.00%   | 100.00%  |

- Can see correlation between when a job was submitted and accuracy of prediction
- Short queue is easy to predict for here as jobs are much more similar, the major challenge is standard queue
Basic KNN model

- Trained temporal model for each of our machines
  - 80% of data for training, 20% for testing. However split on days rather then data elements to provide a fair test

| Predictions accurate within | ARCHER2 | Cirrus | 4-cabinet |
|----------------------------|---------|--------|-----------|
| 1 minute                   | 22.67%  | 51.48% | 12.41%    |
| 5 minutes                  | 35.53%  | 61.20% | 24.35%    |
| 10 minutes                 | 41.37%  | 65.28% | 29.27%    |
| 30 minutes                 | 53.32%  | 75.38% | 38.73%    |
| 1 hour                     | 61.85%  | 80.03% | 44.87%    |
| 2 hours                    | 68.68%  | 88.47% | 55.24%    |
| 6 hours                    | 80.70%  | 93.52% | 71.18%    |
| 12 hours                   | 87.40%  | 96.43% | 81.31%    |
| 24 hours                   | 92.68%  | 98.49% | 89.34%    |

- Can see for ARCHER2 more accurate predictions are better than Slurm’s estimator but less so for less accurate predictions
- Cirrus has much better prediction than Slurm’s estimator!
Queue state as an input

- Correlation between time and date is because this represents a state in the queue
  - Provide queue state as input to models based on histograms which are binned to represent pattern of the queue at the time of submission

- This somewhat improved accuracy but still not perfect predictions

| Name       | Description                                           |
|------------|-------------------------------------------------------|
| nodes Req  | Number of nodes requested by the job                  |
| req wtime  | The requested wall time (hours)                       |
| day        | Day of the week (0-6)                                 |
| hour       | Hour of the day (0-23)                                |
| s_q_jobs   | The number of queued jobs                             |
| s_q_nodes  | The total number of nodes requested by the queued jobs|
| s_q_work   | The total work (nodes × wall time) of the queued jobs |
| m_q_wait   | The median time queued jobs have been waiting for to run (hours) |
| d_q_nodes[0-7] | Histogram of the nodes requested by queued jobs (8 values) |
| d_q_work[0-7]| Histogram of work requested by queued jobs (8 values) |
| d_q_wait[0-7]| Histogram of wait times for queued jobs (8 values) |
| s_r_jobs   | The number of running jobs                            |
| s_r_nodes  | The total number of nodes requested by the running jobs|
| s_r_work   | The total remaining work (nodes × remaining time) of the running jobs |
| d_r_nodes[0-7] | Histogram of the nodes requested by running jobs (8 values) |
| d_r_work[0-7]| Histogram of remaining work of running jobs (8 values) |
| d_r_remain[0-7]| Histogram of remaining times for running jobs (8 values) |

| Predictions accurate within | ARCHER2 | Cirrus | 4-cabinet |
|-----------------------------|---------|--------|-----------|
| 1 minute                    | 28.52%  | 63.45% | 18.43%    |
| 5 minutes                   | 37.09%  | 69.48% | 24.54%    |
| 10 minutes                  | 38.81%  | 72.10% | 32.88%    |
| 30 minutes                  | 59.34%  | 76.74% | 39.28%    |
| 1 hour                      | 61.70%  | 79.89% | 45.00%    |
| 2 hours                     | 70.20%  | 87.70% | 62.28%    |
| 6 hours                     | 81.74%  | 91.60% | 79.94%    |
| 12 hours                    | 88.18%  | 97.74% | 91.44%    |
| 24 hours                    | 93.35%  | 98.73% | 92.80%    |
Stochastic queue state generation

• The challenge is that the amount of work in the queue is unknown
  • Users provide maximum wall times, but these do not necessarily represent the actual wall time of the jobs
    • E.g. on ARCHER2 and 4-cabinet on average 8 times overestimation of job wall time and 6 times on Cirrus.
  • Significantly impacts the overall prediction accuracy

• Therefore generate 100 queue states for each prediction job, each of these comprises randomly generated runtimes but they follow the general distribution of runtimes on the machine so are representative of real jobs.
Stochastic queue state generation

- Once generated run each of these random queue states as inputs to our model and then combine predictions to generate the overall mean job start prediction

- This improves accuracy further, for the first time we beat Slurm’s estimator at all levels of accuracy
Boosted trees

• K-nearest neighbours (KNN) is a very simple model
  • Instead boosted trees enables us to capture non-linear relationships in the data and has been shown to work well for similar workloads.
  • We use the XGBoost library here

• Improves accuracy but not a silver bullet!
Combining classification and regression

- Can combine classification and regression approaches
  - First classify if jobs are immediate starters (e.g. start in less than 10 seconds) and if so they that is the predicted start time
  - Otherwise categorise as one of seven starting categories

| Job start time category          | ARCHER2         | Cirrus         | 4-cabinet       |
|---------------------------------|-----------------|----------------|-----------------|
|                                | exact | relaxed | exact | relaxed | exact | relaxed | exact | relaxed |
| Immediately                     | 73.81% | -       | 89.69% | -       | 88.67% | -       |
| Up to 1 minute                  | 73.60% | 83.45%  | 88.38% | 91.48%  | 68.45% | 75.33%  |
| Between 1 and 5 minutes         | 63.98% | 78.93%  | 75.48% | 80.04%  | 43.26% | 69.32%  |
| Between 5 and 10 minutes        | 61.51% | 71.81%  | 62.79% | 78.15%  | 61.95% | 72.10%  |
| Between 10 and 30 minutes       | 75.90% | 89.55%  | 65.54% | 80.26%  | 71.93% | 80.54%  |
| Between 30 minutes and 1 hour   | 60.54% | 82.20%  | 70.62% | 84.16%  | 71.48% | 74.52%  |
| Between 1 and 4 hours           | 70.91% | 74.77%  | 80.68% | 84.55%  | 77.02% | 82.26%  |
| Over 4 hours                    | 80.26% | 83.34%  | 87.03% | 93.00%  | 77.55% | 82.85%  |

- Exact reports the accuracy of correct predictions being made, relaxed is both correct predictions and those miss-predicted but only in either of the two neighbouring categories
Combining classification and regression

- We undertake these classifications for all one hundred stochastic queue states.
- Specific boosted trees regression models are trained for each category (with neighbours) to undertake job start time predictions.
Combining classification and regression

- This combination of classification and regression considerably improved our performance across all machines.

- We can accurately predict start times of around 65% of jobs within 1 minute of actual time on ARCHER2 and 4-cabinet, and over 76% on Cirrus.

- This extends to three quarters within 10 minutes on ARCHER2 and 4-cabinet and 89% on Cirrus.
Runtime

- Prediction accuracy is our major concern, but we also must run our predictions within a timely fashion.
- All model runs undertaken on a 26-core Intel Xeon Platinum (Skylake) 8170 CPU.

| Activity                                                          | Type     | ARCHER2 (seconds) | Cirrus (seconds) | 4-cabinet (seconds) |
|-------------------------------------------------------------------|----------|-------------------|------------------|---------------------|
| Cumulative Distribution Function (CDF)                            | Training | 264               | 420              | 310                 |
| Histogram bin identification                                       | Training | 6981              | 11298            | 7256                |
| Split training and test data including binning                    | Training | 4322              | 6223             | 5098                |
| Generating random queue test states                               | Training | 7650              | 14890            | 8442                |
| Train Section IV stochastic queue KNN model                       | Training | 20884             | 70667            | 33410               |
| Train Section V classification and regression boosted trees models| Training | 13574             | 34589            | 15911               |
| Single job prediction for Section IV stochastic queue KNN model   | Prediction| 0.88              | 0.89             | 0.89                |
| Single job prediction for Section V classification and regression boosted trees models | Prediction | 0.11             | 0.18             | 0.10                |

- Model training takes a long time, but only needs to be done once.
- Prediction for our most accurate approach is a tenth of a second per machine.
User insights

• Whilst urgent workloads are our major focus, it is also possible to gain insights from running predictions through our models.

Avoid requesting 4 hours max wall time with 32 nodes, instead select 8 or 12 hours.

With 16 nodes if you can stay at maximum wall time of 2 hours or less, as beyond this there is an increase in job wait time.
Conclusions

• Our approach delivers significantly improved accuracy than Slurm’s estimations and previous machine learning approaches

• We use a combination of classification and regression models to most accurately predict the queue wait time

• Whilst our major focus has been for urgent workloads, and the accuracy delivered is reasonable for this, there are also numerous other uses too.