SLearn: A Case for Task Sampling based Learning for Cluster Job Scheduling

Akshay Jajoo* Y. Charlie Hu Xiaojun Lin Nan Deng

* Now with Nokia Strategy and Technology
Challenges in Cluster Scheduling

• **Online:** Jobs arrive online
  – They compete for shared resources

• **Diverse SLOs:** Different jobs have different SLOs
  – Deadline, fast completion, least network usage, etc.

• **Non-clairvoyant:** Runtime properties of these jobs are unknown
  – Resource needs are diverse
An Effective Approach for Cluster Scheduling

Learn runtime characteristics of pending jobs
Widely Used Approach for Learning

History-based learning:
Learning from past execution history
(Learning in Time)
Previous Work on History-based Learning

- 3Sigma (Eurosys 2018)
- Don’t Cry Over Sp.... (ATC 2017)
- Morpheus (OSDI 2016)
- Perforator (SoCC 2016)
- JamiasVu (Techreport, 2016)
- Corral (Sigcomm 2015)
- Rayon (SoCC 2014)
- Jockey (EuroSys 2012)

Learning offline models for recurring jobs
Periodically updating the offline learned models
Learning for new jobs
Assumption 1: Most of the jobs are recurring #✗#

✓ Corral [43] and Jockey [30] → only 40% recurring jobs
✓ Morpheus [44] → 60% are recurring

Assumption 2: Same or similar jobs will perform consistently over time #✗#

✓ In a Microsoft cluster, within a year ratio for two types of machines changed from 80/20 to 55/45 [44]
  – Same job’s performance varies by 40% on the two machines
✓ 50% of applications were updated every month [44]
Poor Prediction Accuracy of History-based Learning

• State-of-the-art history-based predictor, 3Sigma [47],
  – Predicts with more than 100% error for 23% of jobs
  – For a significant fraction of jobs, it’s error is more than 1000%

[47]: 3Sigma, Eurosys 2018
An alternate approach for learning: SLearn - Learning in space
Key Insight: Spatial Dimension
Sampling the population is an effective way to learn its properties.

Can this spatial dimension be exploited to learn job properties?
SLearn: Online Learning by Sampling the Spatial Dimension

\[ T = \frac{T_1 + T_2}{2} \]

Total Size = 7 * T
Learning in Time (History) vs. Learning in Space (SLearn)

<table>
<thead>
<tr>
<th></th>
<th>Learning in Time</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning in Time</td>
<td>No/Yes</td>
<td>Depends</td>
</tr>
<tr>
<td>Learning in Space</td>
<td>Yes</td>
<td>Depends</td>
</tr>
</tbody>
</table>

1. Will learning in space be more accurate than in time?
2. If so, will the better accuracy be able to compensate for the delay due to sampling?
Trace based Variability Analysis
Variation Across Time vs Variation Across Space

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration</td>
<td>7 Months (Jan-July’16)</td>
<td>29 Days, May 2011</td>
<td>31 Days, May 2011</td>
</tr>
<tr>
<td>Machines</td>
<td>872 (441 + 431)</td>
<td>~12.5K</td>
<td>~8*12.5K</td>
</tr>
<tr>
<td>Number of Jobs</td>
<td>~0.4 Million</td>
<td>~0.7 Million</td>
<td>~30 Million</td>
</tr>
</tbody>
</table>

[1]: 2Sigma hedge fund: www.twosigma.com
CoVs of the Estimated Job Size: Space vs. Time

History (Time) = CoV in average task lengths of similar jobs

Tasks (Space) = \( \frac{\text{CoV of lengths of sampled tasks}}{\sqrt{\text{Number of tasks sampled}}} \)

Figure shows analysis results for 70 jobs selected at random from the 2Sigma trace.
Varying the History Length in Estimation across Time

Variation across tasks (Space) is lower than variation across history (time) for all history lengths.

CDFs of CoVs of estimated job size across history with varying history duration.
Varying the History Length in Estimation across Time

**2Sigma**

![Graph 1](image1.png)

**Google 2011**

![Graph 2](image2.png)
Varying the History Length in Estimation across Time

Other Properties

Google 2011

Google 2011
Learning in Time (History) vs. Learning in Space

<table>
<thead>
<tr>
<th>Learning in Time</th>
<th>Learning in Space</th>
<th>Prediction</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

1. Will learning in space be more accurate than in time?

2. If so, will the better accuracy be able to compensate for the delay due to sampling?
Generic Scheduler (GS)

Cluster Manager → Scheduler

1. Job submission
2. Pilot task schedule
3. Pilot task finish info.
4. Final job runtime

Predictor – SLearn/Baselines

Job with 2 pilot tasks (red) → Probe Queue → Priority Queues
SLearn Predictor – Design Overview

• Samples a small fraction of tasks for each job
• Schedules at a high priority till completion
• Uses their runtime to estimate job runtime
SLearn Predictor – Implementation Challenges

• How to ensure high priority for sampling tasks?
  – Sampling jobs are given 2\textsuperscript{nd} highest priority

• Won’t sampling delay be too hard on jobs with very few tasks?
  – Jobs under certain width are bypassed from sampling and given the highest priority

• How to determine sample tasks?
  – Sample size
    • Decided by an adaptive algorithm
  – Picking the sampling tasks
    • Random
Baseline Predictors and Policies

1. **3Sigma** (Eurosys 2018) – State-of-the-art history-based predictor
2. **3SigmaTL** – 3Sigma + thin bypass like SLearn
3. **Point Median** – Like 3Sigma, predicts median runtime
4. **LAS** (SoCC 2018) – Online approximation of SJF, *e.g.*, Kairos
5. **FIFO** – Basic Yarn scheduler
6. **Oracle** – 100% accurate prediction
Experiment Details

Workload – Extracted from larger traces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>New Name</td>
<td>2STrace</td>
<td>GTrace11</td>
<td>GTrace19</td>
</tr>
<tr>
<td>Number of jobs</td>
<td>1253</td>
<td>1251</td>
<td>1255</td>
</tr>
</tbody>
</table>

Experiments

JCT improvement
Working of adaptive sampling

Runtime error prediction
Real 150 node testbed

[1]: 2Sigma hedge fund: www.twosigma.com
Job Completion Time (JCT) Speedup

\[ \text{Speedup} = \frac{\text{JCT with baseline}}{\text{JCT with SLearn}} \]

Higher the speedup better is SLearn
Adaptive Sampling Algorithm in Action

The adaptive sampling algorithm effectively balances the trade-off between prediction accuracy and sampling overhead.
Learning Accuracy

2STrace

GTrace19

GTrace11
JCT Speedup on Testbed
150 Node Azure Cluster

CDF

JCT speedup over 3Sigma

GTrace11
2STrace
GTrace19
Additional Results and Future Work

• Additional Results: SLearn for DAG
  – 1.26 x over 3Sigma-DAG

• Future work:
  – Combining history and sampling
  – Dynamic adjustment of ThinLimit
  – Sampling in heterogeneous clusters
SLearn - Summary

• History-based learning suffers from an outdated history

• We propose and show the effectiveness of a new task-sampling based method to learn runtime properties of distributed jobs online

• We have performed large-scale trace analysis, simulation and testbed experiments

SLearn improves average job completion time over state-of-the-art history-based system, 3Sigma, by 1.56 ×