BATCH JOB SCHEDULING USING ENHANCED WALLTIME PREDICTION

Dalibor Klusáček¹
Mehmet Soysal²

¹CESNET, Czech Republic
²Steinbuch Centre for Computing (SCC), Karlsruhe Institute of Technology (KIT), Germany
Batch job schedulers rely on user-provided walltime estimates:
- When selecting proper queue (short/normal/long)
- When performing predictions (when and where will a job start)
- When staging data in advance of a computation
- When optimizing the performance / utilization of the system (backfilling)

Few examples:
- Computational Job
  - estimated resources
  - estimated walltime (max. job runtime)
When will a new job start?

- Current schedule
- Predicted start time

Physical nodes

new job

predicted start time

new job
Backfill new job to increase utilization

- New job backfill to increase utilization

Current schedule vs. backfilled schedule.
Users are bad and/or lazy when delivering estimates

- Estimates are typically very over-estimated and coarse grained
- To prevent jobs from being killed due to exceeding their walltime
- Fast vs. slow machines (worst case scenario: slow machine => larger runtime)
Suboptimal Scheduling

- Schedule is “shorter” than expected
- Inaccuracies then impact predictions

![Diagram showing predicted vs. actual start times for new jobs on physical nodes over time.](image)
Inaccuracies hamper scheduling (utilization & other metrics)

- No place to backfill the job...
- It is too late now to backfill...

Physical nodes

new job

Physical nodes

realtime

time

new job

new job

new job

new job
Inaccuracies hamper scheduling (utilization & other metrics)

No place to backfill the job…

unreachable opportunity
Walltime predictions has been largely studied in the past

Various techniques and approaches
- How to motivate users
- How to predict job runtime automatically

„Soft walltime“
- New feature in PBS Professional since 2017
- Actual implementation allowing for the use of such refined estimates
- Original estimate is still used for job killing (but not the soft walltime)
- A predictor must be implemented on your own

Soft walltime cannot be specified by user
- To prevent users from cheating
SIMULATIONS AND RESULTS
Considered Problems

- How good is simple predictor vs. user estimates?
  - Predictor uses historic runtimes of few past jobs to generate new estimate (soft walltime)

- What is its impact on performance?

- Metrics
  - Predictor’s accuracy – distribution of absolute errors (error = estimate – runtime)
  - Number of backfilled jobs
  - Avg. wait time

- Simulator and historic workloads used for the analysis
Does the predictor beat user-provided estimates?

YES! 😊
- **Simple predictor**
  - Using historic runtimes of few past jobs to generate new estimate (soft walltime)

- **CDF of absolute errors**
  - For all jobs
Does the same predictor work for all users?

Almost... 😊
“Per user” view of job’s estimate error

Original estimates

- job's estimate error (hours)

Predictor

- job's prediction error (hours)
Does the predictor increase backfilling opportunities?

YES! 😊
Backfilled Jobs

- **Number of backfilled jobs**
  - Indicates the ability of the scheduler to “fill gaps” with small-enough jobs
  - In order to increase utilization

- **Original estimates:** 7.8% of jobs is backfilled

- **Predicted estimates:** 15.3% of jobs is backfilled
  - Higher variability of estimates increases the chance to “fill gaps”
Does more accurate estimates guarantee lower avg. wait time?

NO! 😞
Comparison of job wait times using “heatmaps”

- Jobs divided into “buckets” wrt. #of CPUs and job duration
- A “bucket” then shows improvement/deterioration of avg. wait time when the predictor is used instead of the original user-provided estimate
- “blue color” => worse avg. wait time wrt. user estimate
- “red dot” => better avg. wait time wrt. user estimate
- Better precision does not imply better wait time
- Other factors (beside estimates’ precision) play important role
  - Job ordering as a result of “different” job duration (depending on prediction)
  - Differences in scheduling algorithms
  - No simple correlation between estimates’ accuracy and the performance

CONSERVATIVE backfilling

EASY backfilling
Predictions represent an interesting optimization option
- Better job start time and node predictions
- Can be used to reflect different computing capabilities of the underlying HW
  - E.g., assign smaller soft walltime if a job is scheduled on a faster and/or more I/O capable machine

User-transparent (does not require user cooperation)
- Also minimizes the possibility of users cheating the scheduler

Our goal is to deploy walltime predictors in our systems
- MetaCentrum & SCC (predictions, advanced data-staging)
- Further analyze our workloads and various predictors’ suitability
THANK YOU!

klusacek@cesnet.cz
mehmet.soysal@kit.edu