A UT/ORNL PARTNERSHIP
NATIONAL INSTITUTE FOR COMPUTATIONAL SCIENCES
NICS

Discovering the Petascale User Experience in Scheduling Diverse Scientific Applications: Initial Efforts towards Resource Simulation

Lonnie D. Crosby, Troy Baer,
R. Glenn Brook, Matt Ezell,
and Tabitha K. Samuel

Kraken (Cray XT5)

- Contains 9,408 compute nodes (112,896 cores)
  - each containing dual 2.6 GHz hex-core AMD “Istanbul” processors, 16 GB RAM, and a SeaStar 2+ interconnect.
- Peak Performance of 1.17 PF
- Scheduling Environment
  - TORQUE 2.4.8
  - Moab 5.4.3

Resource Scheduling Objectives

Efficiently produce scientific results by

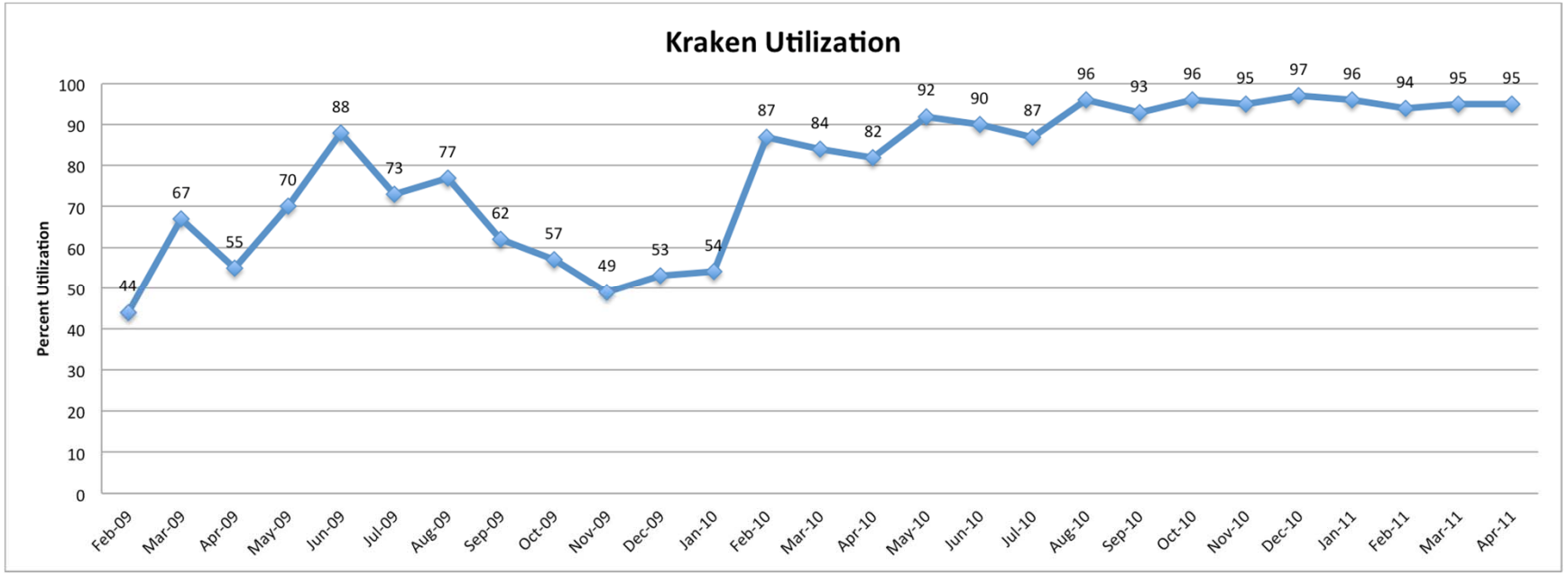
- maintaining high resource utilization (< 90%).
- providing reasonable throughput for all job classes.

How do scheduling policies affect

- resource utilization?
- user experience in terms of job throughput?

Data Collection: Utilization

- Utilization
  - Snapshot based method
    - System utilization is collected at regular intervals.
  - Collection of job statistics
    - Utilization is calculated over periods based on job statistics.



Data Collection: Job Statistics

- Database solution
  - Collects information from the Moab event logs for each job.
- Metrics
  - Resource Utilization
    - need resource downtime information to correct result.
  - Job distributions
    - node count
    - requested/used walltime
    - queue duration

job_stats
job_id: INTEGER
num_tasks: INTEGER
username: CHARACTER VARYING(512)
group_name: CHARACTER VARYING(512)
wallclock_limit: INTEGER
required_class: CHARACTER VARYING(512)
submission_time: INTEGER
dispatch_time: INTEGER
start_time: INTEGER
completion_time: INTEGER
system_queue_time: INTEGER
qos: CHARACTER VARYING(1024)
job_flags: CHARACTER VARYING(1024)
account_name: CHARACTER VARYING(512)
resource_mgr_extension: CHARACTER VARYING(512)
bypass_count: INTEGER
allocated_host_list: TEXT
app_simulator_data: TEXT
job_message: TEXT
effective_queue_duration: INTEGER
completion_code: INTEGER
job_events: CHARACTER VARYING(100)
date_of_file: INTEGER

Resource Simulation

- Moab Simulation Mode
  - Resource Trace
  - Workload Trace
  - Configuration
- Resource Trace
  - The list of all compute nodes. No node failures included.
- Workload Trace
  - from period May 1 – December 31, 2010. (99,072 cores)
  - Various job types were removed.
- Configuration
  - Modified policy set derived from production resource

Policy Definition

Production Resource

- Workload Trace
  - May 1 – Dec. 31, 2010
- Resource
  - Includes downtime and node failures
  - Preventative maintenance widows up to 8 hours.
- Policy
  - Priority based primarily on core count.
  - Backfill enabled
  - Reservation depth of one
  - Limits on the number of eligible jobs per user (5) and project (10).

Simulator

- Removed jobs
  - > half resource
  - that don't use compute nodes
- Resource
  - Constant 99,072 cores (8,256 nodes)
  - Regular weekly PM window of 10 minutes.
- Policy
  - User or project specific/temporal restrictions removed.
  - Queue Depth of 1,000

Resource Simulation Requirements

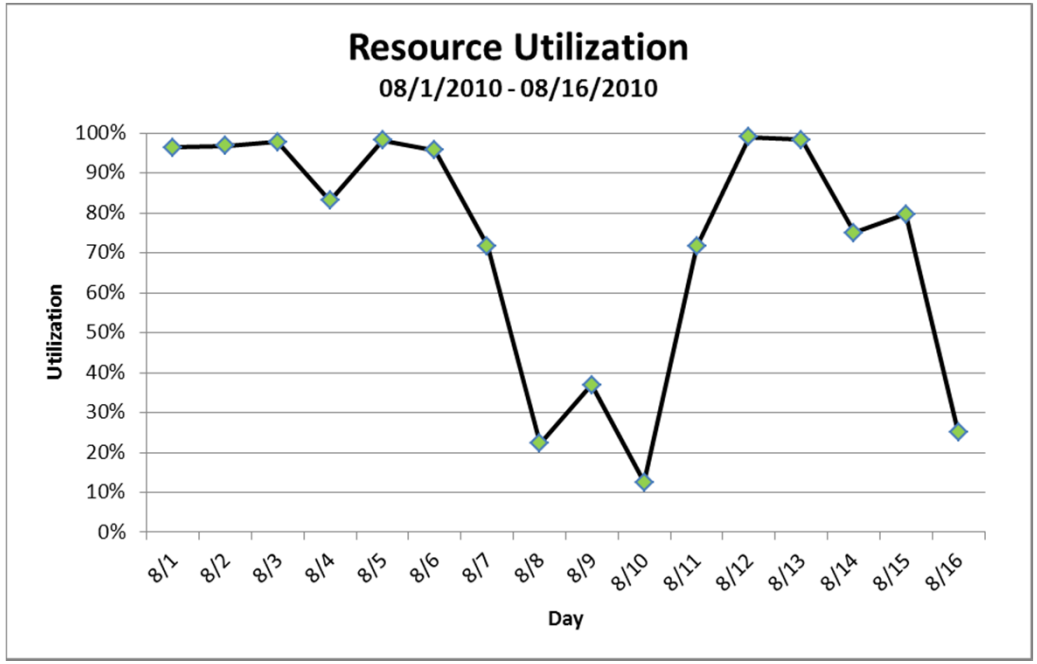
- Timeframe
  - Acquire job statistics over long time periods (6 months – 1yr)
  - Perform simulation in accelerated time (30 x)
- Reliable results
  - at least qualitative statistics with correct sign
  - at least qualitatively realistic behavior
- Goals
  - Experiment with policy changes
  - Determine the effect of changes on utilization and throughput

Simulation Experiment

- Jobs submitted on the production resource which do not have computational time remaining are given a quality of service (QoS) of negbal.
- This QoS receives a highly negative priority which makes the jobs the last to run. However, these jobs are eligible for backfill.
- Would disallowing these jobs from utilizing backfill adversely affect utilization or job throughput?

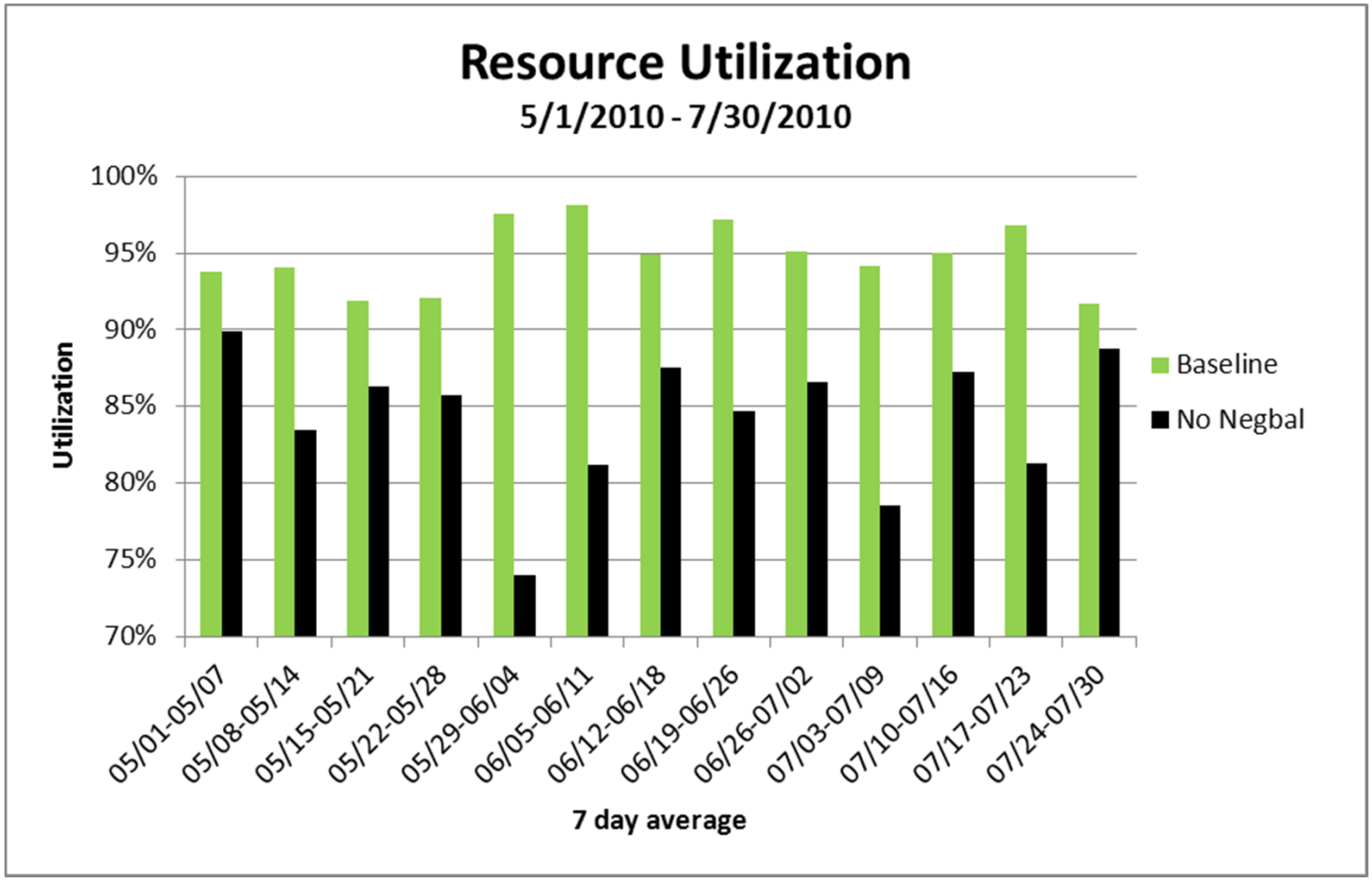
Some initial problems

- Classes of jobs remain in queue indefinitely
  - JOBCANCEL events
  - Jobs which are classified as Blocked by active policies
- Eventually starves the simulation of workload
  - May 1 – July 31, 2010
- Simulation Time step
  - Poll Interval
  - changed from 30 – 60s to 5 – 10 minutes.

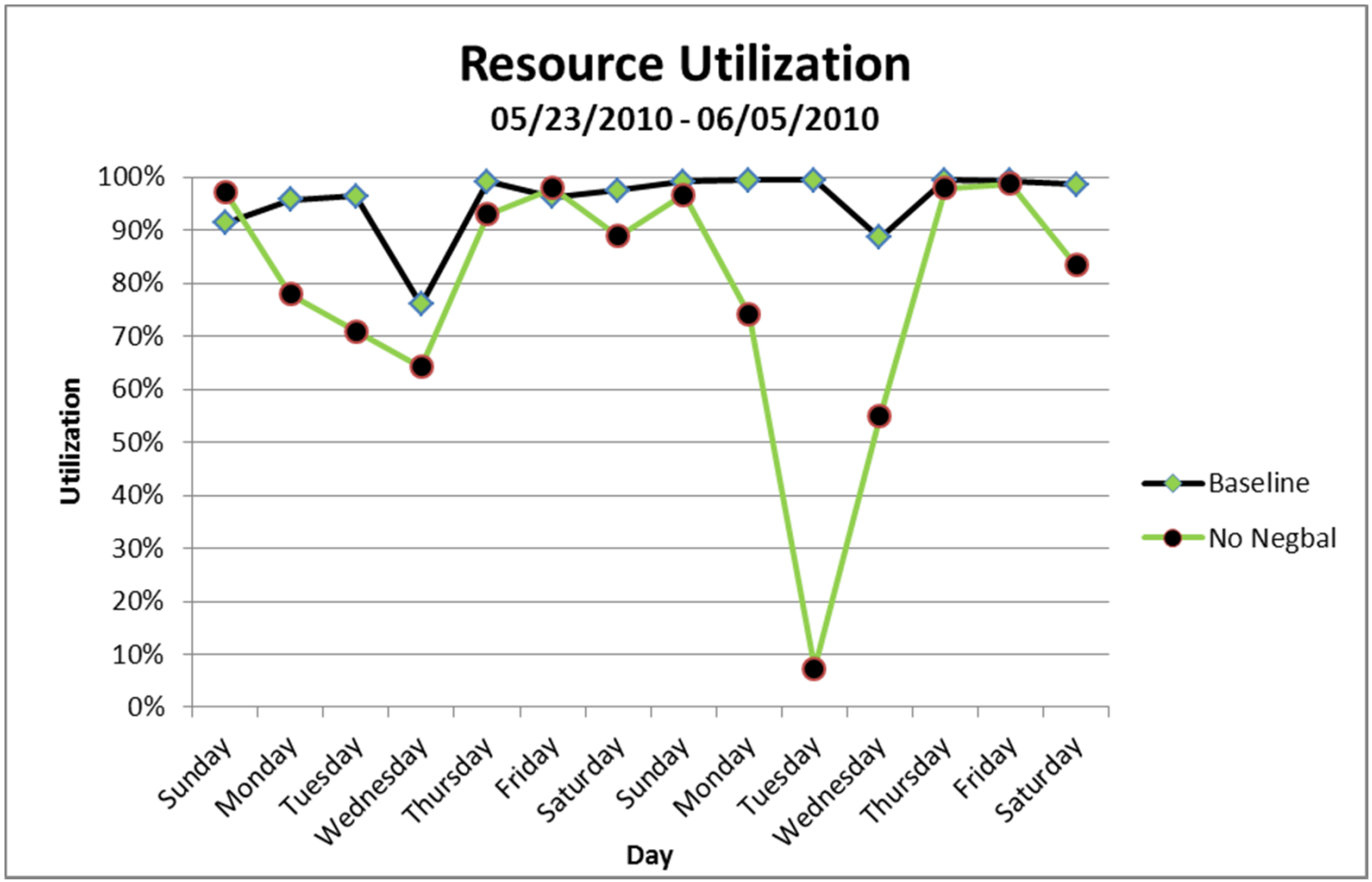


Utilization Results

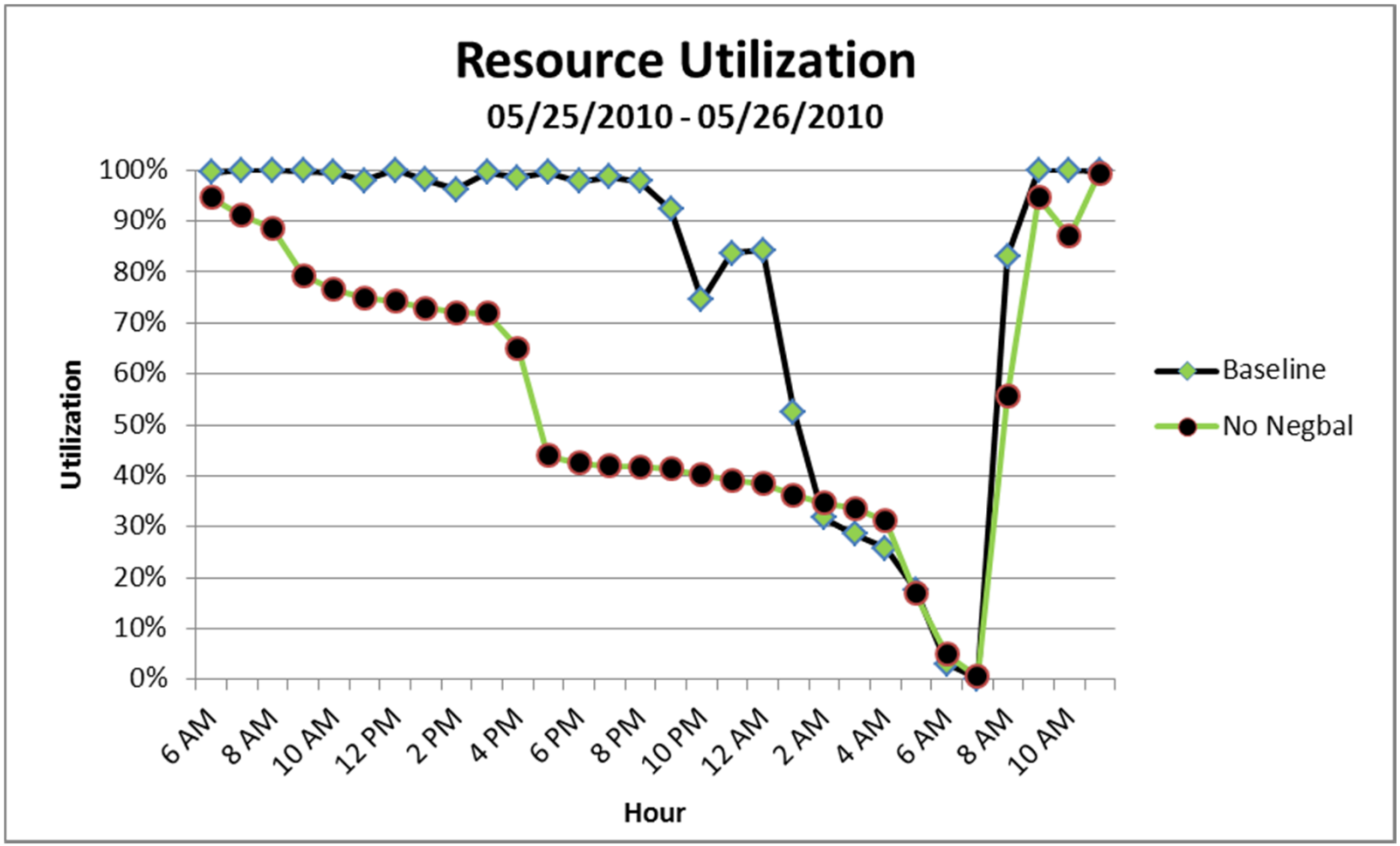
Average Utilization
Baseline: 95%
No Negbal: 84%



Utilization Results



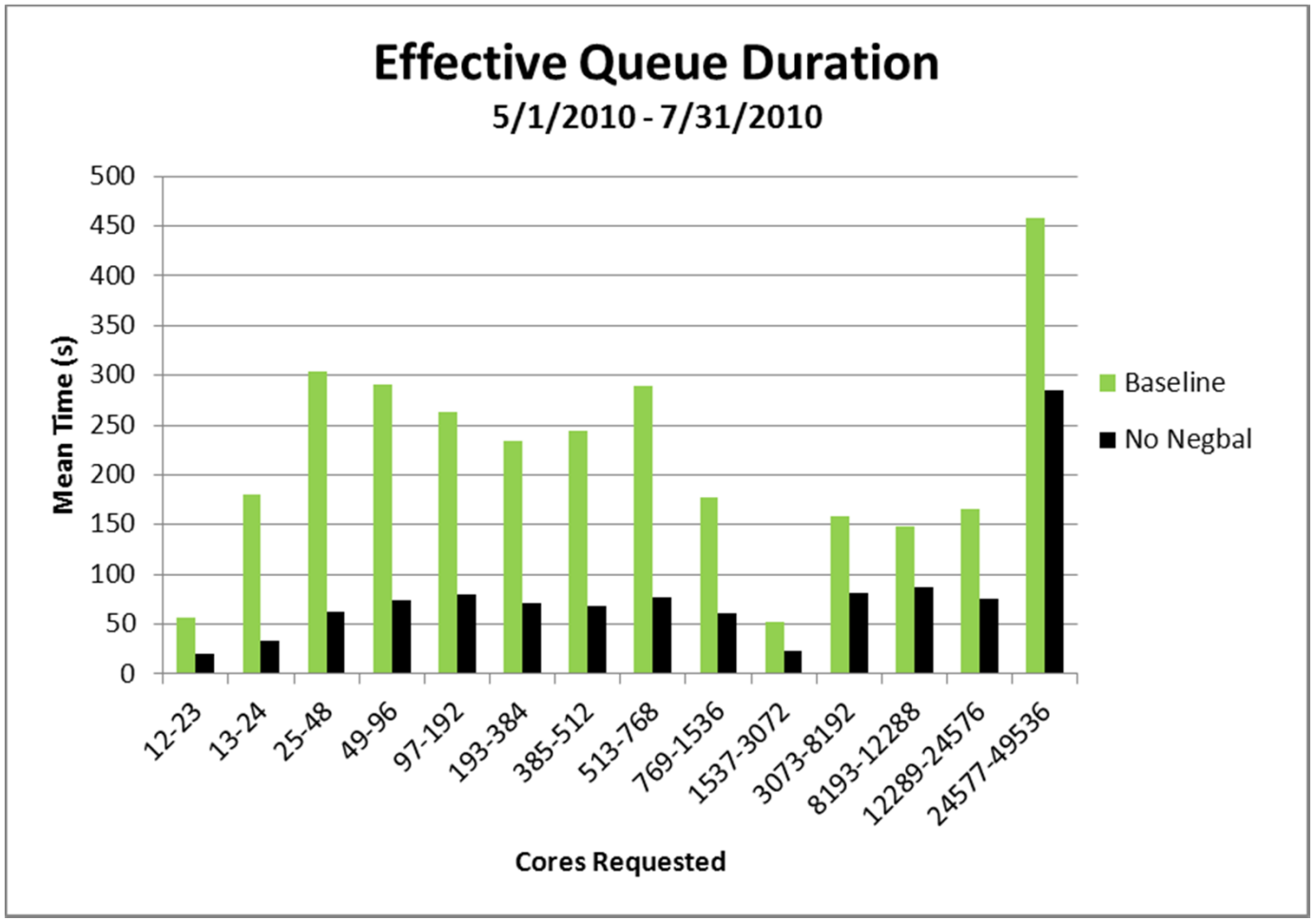
Utilization Results



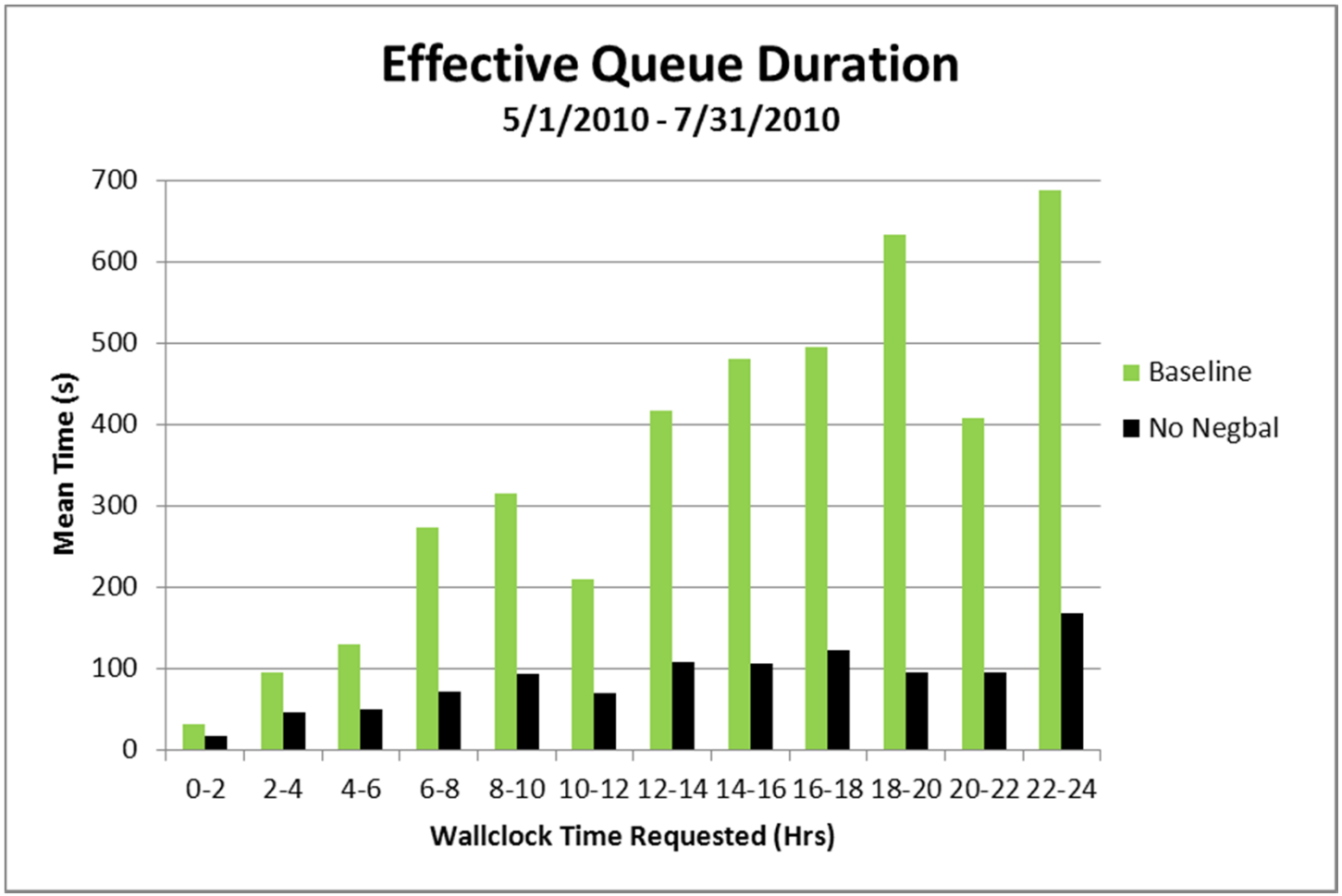
Some Conclusions

- Utilization drops drastically when “negbal” jobs are not allowed to backfill.
- Utilization drops seems to be partially due to less efficient draining profile. This profile seems to be due to a lack of jobs eligible for backfill.
- However,
  - Baseline: 11,774 “negbal” jobs were run (116,551 other jobs)
  - No negbal: 10,631 “negbal” jobs were run (99,091 other jobs)
- Jobs eligible for backfill seem to be plentiful
  - 55% of non-negbal jobs require less than 2 hours
  - 68% of non-negbal jobs require less than 512 compute cores

Effective Queue Duration



Effective Queue Duration



Probable Scenario

- Both experiments
  - Queue depth fills with immobile jobs, until no throughput possible
- No Negbal experiment
  - “Negbal” jobs also fill the queue depth, these are also largely immobile as long as other jobs are present.
  - When majority of queue depth (1,000) is composed of these jobs, any other available job get high effective priority.
  - When queue depth is filled with these jobs, they are run. However, backfill cannot be utilized.

Conclusion (Wish List)

- Fix simulation bugs
  - Remove problem of immobile jobs in queue.
- Simulation time step
  - Better control of simulation time step without a reliance on the Poll Interval.
- Queue formation
  - Utilize submission times present is workload trace.
  - Utilize a minimum queue depth to draw jobs in workload starvation situations.