Heterogeneity and Dynamicity of Clouds at Scale: Google Trace Analysis

Presented by: Boyan Li

Motivation

- Better understand the challenges in developing effective cloud-based resource schedulers.
- Algorithms implemented may not perform as expected.
- Need to examine the system characteristics in real world.

Overview

Workload Characteristic:

- Heterogeneity
- Dynamicity
- Poorly Predicted Resource Needs
- Resource Class Preferences and Constraints

Background

Google trace:

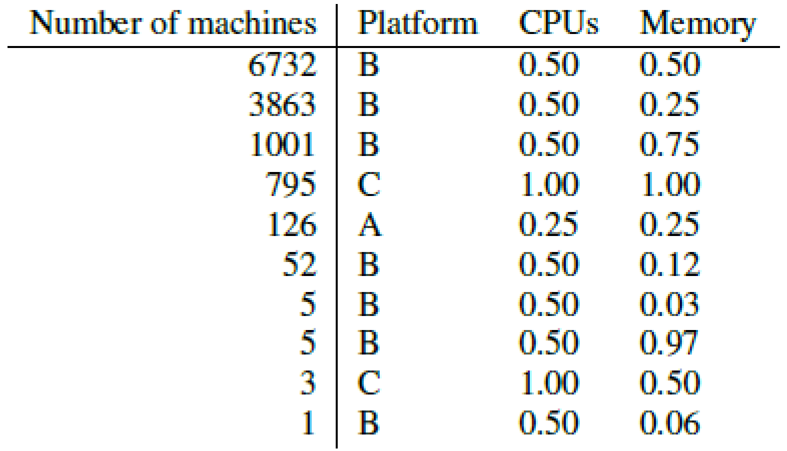
- Workload Type:
    - long-running services
    - DAG-of-task systems: many independent short tasks
    - high-performance (or throughput) computing
- Consists several concurrent traces for a month of activity in a single ~ 12K machine cluster.
- Describes hundreds of jobs.

Heterogeneity

- Machine Types and Attributes
- Workload Types
- Job Durations
- Task Shapes
- Distribution

Machine Types and Attributes

- Machines in the trace are of different CPU memory ratios.
- Machines are acquired over time using whatever configuration was most cost-effective, which is a common case for systems such as Google Compute Engine and Amazon AWS.



Number of machines	Platform	CPUs	Memory
6732	B	0.50	0.50
3863	B	0.50	0.25
1001	B	0.50	0.75
795	C	1.00	1.00
126	A	0.25	0.25
52	B	0.50	0.12
5	B	0.50	0.03
5	B	0.50	0.97
3	C	1.00	0.50
1	B	0.50	0.06

Table 1: Configurations of machines in the cluster. CPU and memory units are linearly scaled so that the maximum machine is 1. Machines may change configuration during the trace; we show their first configuration.

Workload Types

12 task priorities grouped into 3 sets:

- Production (9 - 11)
- Middle (2 - 8)
- Gratis (0 - 1)

Production priorities account for more resource usage.

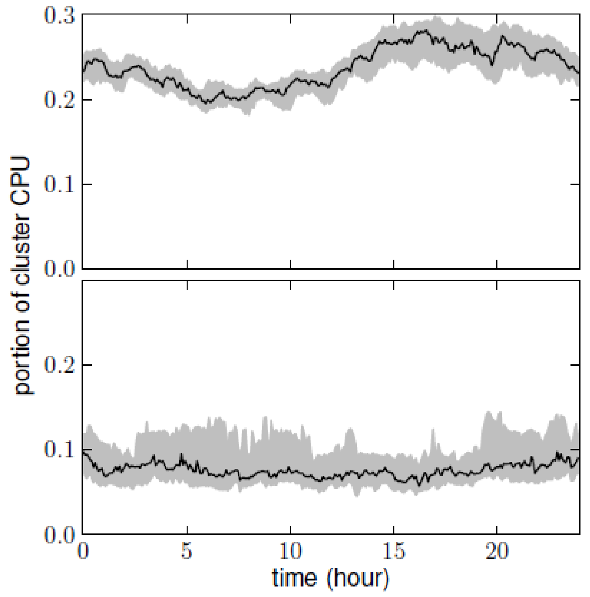


Figure 1: Normal production (top) and lower (bottom) priority CPU usage by hour of day. The dark line is the median and the grey band represents the quartiles.

Workload Types

Production priorities also include more long-duration jobs.

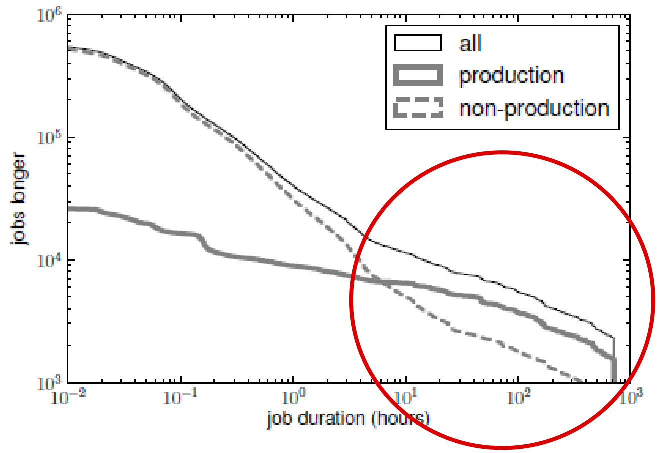


Figure 2: Log-log scale inverted CDF of job durations. Only the duration for which the job runs during the trace time period is known; thus, for example, we do not observe durations longer than around 700 hours. The thin, black line shows all jobs; the thick line shows production-priority jobs; and the dashed line shows non-production priority jobs.

Workload Types

The division by priority is not perfect.
There are both latency-sensitive and non latency-sensitive tasks in each priority set.

Dynamicity

- Machine Churn
- Task and Job Churn

Machine Churn

- Machines become unavailable more frequently: about 40% of the machines are unavailable to the scheduler at least once.
- The failures are suspected to represent machine maintenance. (lacking information to confirm this)
- These periods of unavailability last less than half an hour, which is more consistent with maintenance than hardware failure.
- At no point in the trace does it appear that less than 98% of the machines are available, and over 95% of the time, more than 99% of the machines are available.

Task and Job Churn

- Scheduler must decide where to place runnable tasks frequently.
  - hundreds of task placement decisions per second in peak,
  - several schedulings per second in quieter time
- Resubmissions account for nearly half of task submissions to the scheduler.
- Large spikes can be attributed to ‘crash-loops’

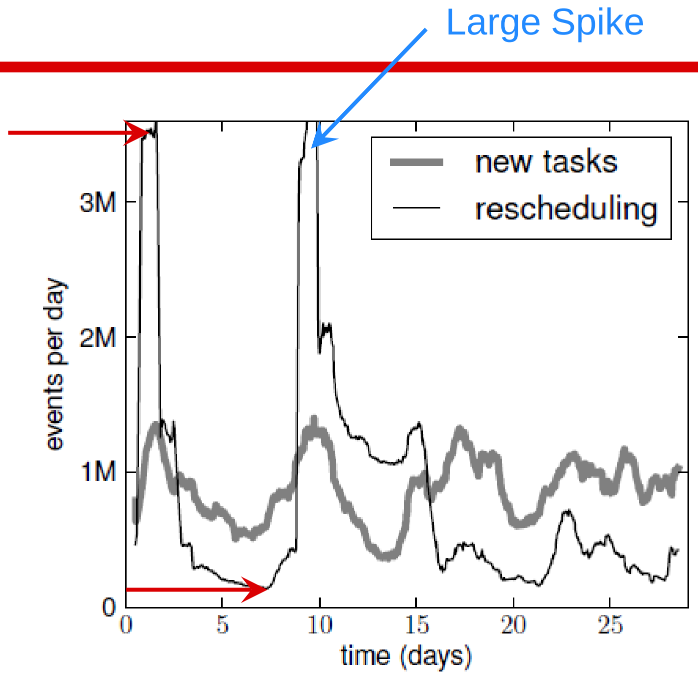


Figure 5: Moving average (over a day-long window) of task submission (a task becomes runnable) rates.

Task and Job Churn

Crash-loop:

- tasks of a job fail deterministically shortly after starting.
- approximately 2% of the memory-time requested comes from jobs that experience more than 10 failures per task
- mostly occur in lower priority jobs

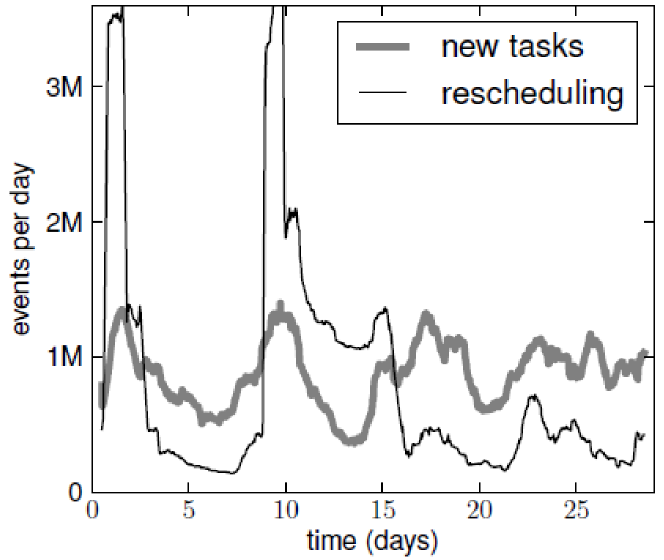


Figure 5: Moving average (over a day-long window) of task submission (a task becomes runnable) rates.

Task and Job Churn

Evictions:

- another common cause of task rescheduling
- evictions happen when another task of similar priority start to run in the same machine, which is examined to be unnecessary
- evictions happen so soon after the higher priority task is scheduled, it’s not likely that many of these evictions are driven by resource usage monitoring
- after around 30% of evictions, resources requested by the evicted tasks appear to remain free for an hour after eviction
- evictions are unnecessary or make way for brief usage spikes we cannot detect

Resource Usage Predictability

- Usage Overview
- Usage Stability
- Short Jobs
- Resource Requests
- Repeated Submissions

Usage Overview

- The cluster is heavily booked:
  - More than 80% of the memory
  - More than 100% of the CPU
- Overall usage is much lower:
  - average usage of memory does not exceed about 50%
  - CPU usage does not exceed about 60%

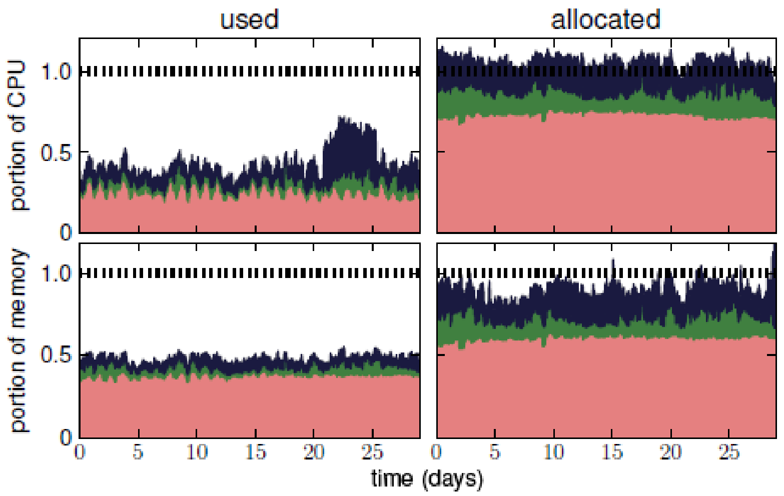


Figure 8: Moving hourly average of CPU (top) and memory (bottom) utilization (left) and resource requests (right). Stacked plot by priority range, highest priorities (production) on bottom (in red/lightest color), followed by the middle priorities (green), and gratis (blue/darkest color). The dashed line near the top of each plot shows the total capacity of the cluster.

Usage Stability

- When tasks run for several hours, their resource usage is generally stable.

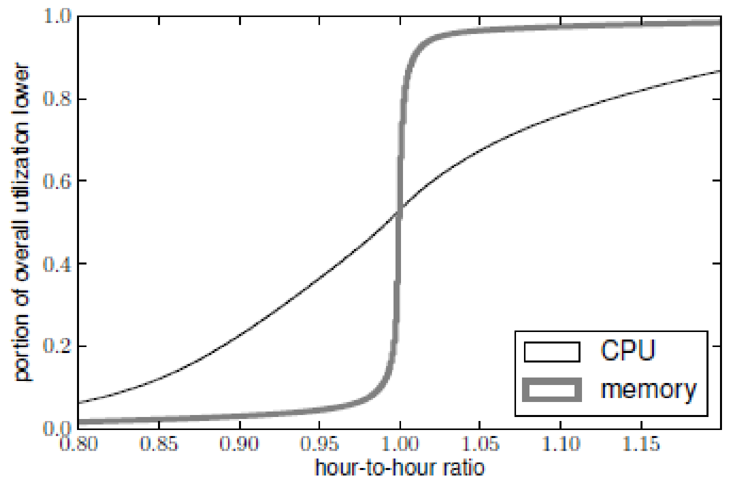


Figure 9: CDF of changes in average task utilization between two consecutive hours, weighted by task duration. Tasks which do not run in consecutive hours are excluded.

Usage Stability

- Due to the existence of large amount of small tasks, the system doesn’t change too much to fit into new tasks.
- Longer-running tasks tend to mimic other longer-running tasks.

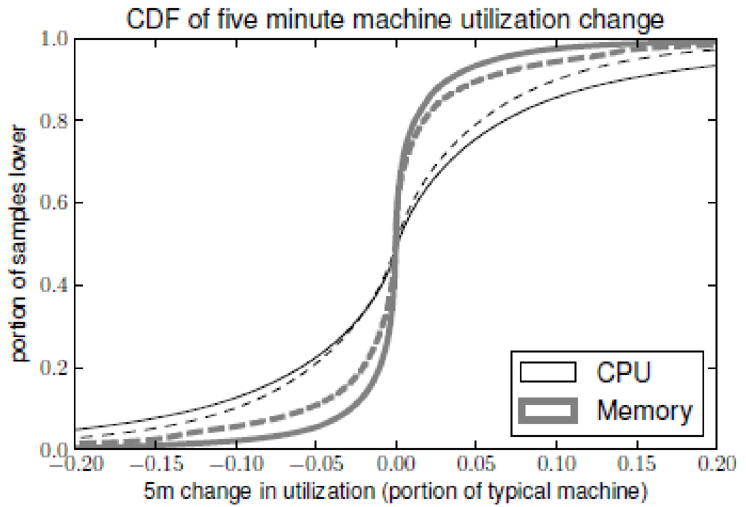
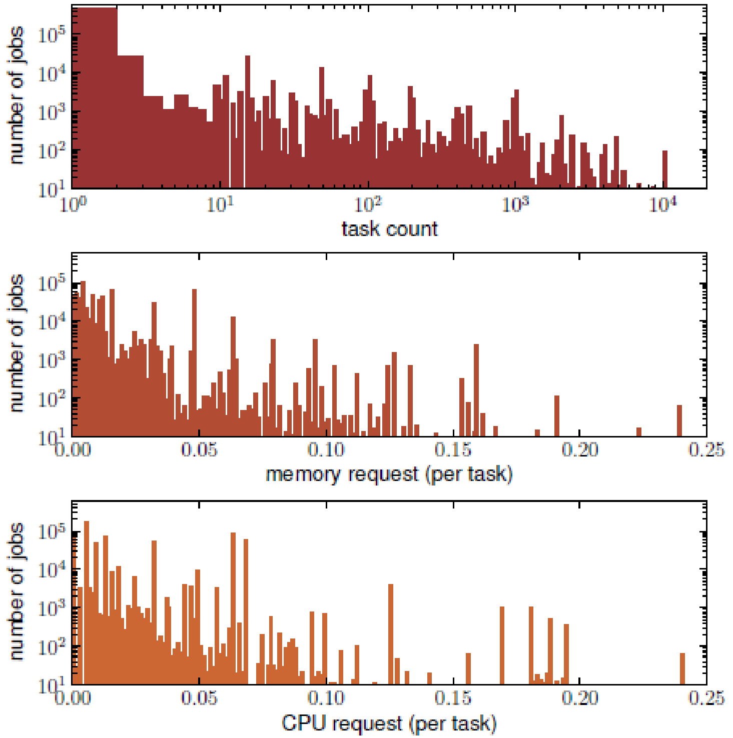


Figure 10: CDF of changes in average machine utilization between two consecutive five minute sampling periods. Solid lines exclude tasks which start or stop during one of the five minute sampling periods.

Resource Request

Non-automation

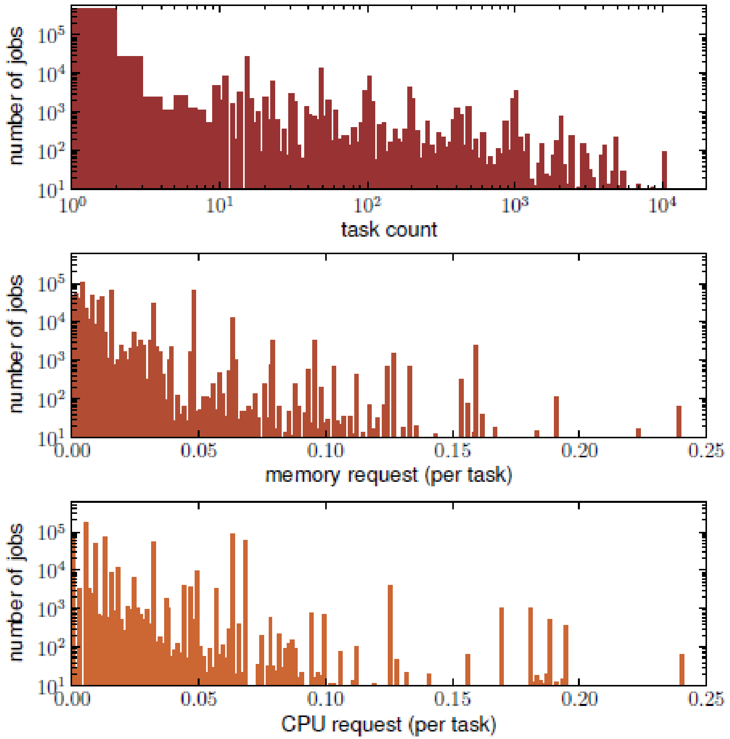
- specified manually - do not correspond to actual usage



Resource Request

Peak value is caused by the fact that users tend to specify round numbers.
e.g.: 4, 8, 16, or 256 cores
4G, 8G or 32G memory
Round number

- for memory, doesn't reflect real usage
- for CPU, accurately reflect CPU as a disproportionate number of tasks would use



Repeated Submissions

- Some programs are run repeatedly - source of resource prediction.
- Identify repeated jobs by “logical job name”.
- Frequently repeated jobs do not account for very much of the utilization of the cluster.
    - names repeated more than 5 times account for only 30% of utilization

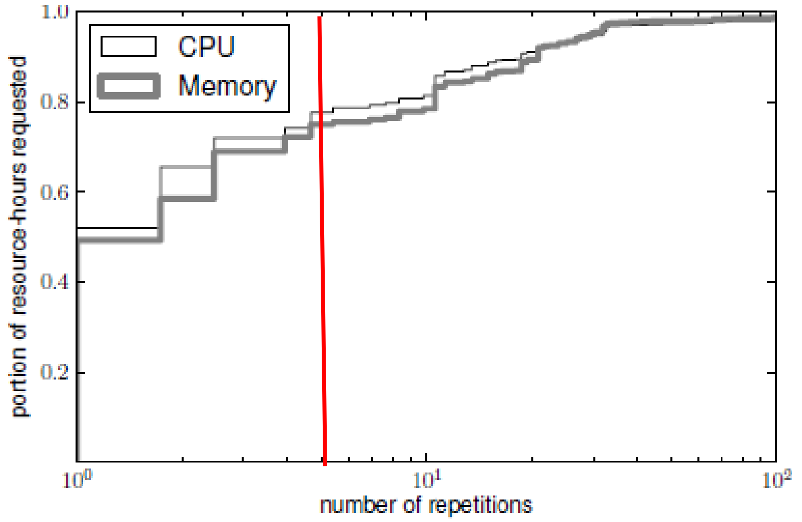


Figure 14: CDF of the portion of CPU-hour and memory-hours requested by jobs by the number of apparent repetitions.

Repeated Submissions

- More concerningly, the usage of repeated jobs is often not consistent.
- Without more information, predictions based on this notion of repeated jobs are only likely to be accurate within 25% for jobs accounting for less than half of usage (in memory-hours) of all the repeated jobs.

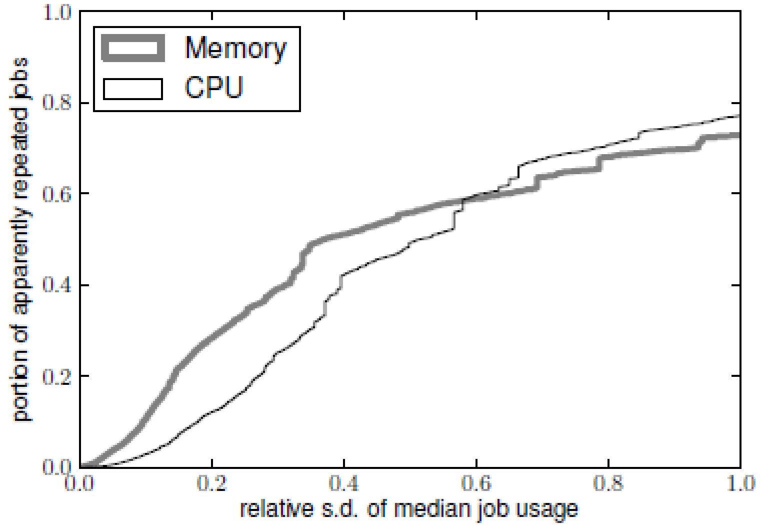


Figure 15: CDF of relative standard deviation of median repeated job utilization.

Conclusion

- Heterogeneity:
    - resources
    - tasks
    - constraints
- Dynamicity
    - workload changes quickly
- Need for new cloud resource schedulers.

Pros:

1. It’s conducted for sizable multi-purpose cluster which is being used at a regular basis, which reflects facts of the real world.
2. It reveals some flaws in the current system, which provides directions to improve the current system.

Cons

1. The way in which the researchers identify repeated jobs by identical name is suboptimal.
2. The data set is comparatively small.
3. The analysis may be biased to this specific cluster. It’s not general enough.

Takeaways

1. Differentiate jobs with the extent of latency-sensitive.
2. Develop novel scheduling algorithms for low level tasks to avoid large amount of resubmissions.
3. Need for a scheduler to adjust the allocated resource for a user according to its average usage.

Questions?