Università della Svizzera italiana

Faculty of Informatics

Bachelor Thesis

May 19, 2021

Understanding and Comparing Unsuccessful Executions in Large Datacenters

Claudio Maggioni

Abstract

The project aims at comparing two different traces coming from large datacenters, focusing in particular on unsuccessful executions of jobs and tasks submitted by users. The objective of this project is to compare the resource waste caused by unsuccessful executions, their impact on application performance, and their root causes. We will show the strong negative impact on CPU and RAM usage and on task slowdown. We will analyze patterns of unsuccessful jobs and tasks, particularly focusing on their interdependency. Moreover, we will uncover their root causes by inspecting key workload and system attributes such asmachine locality and concurrency level.

Advisor
Prof. Walter Binder
Assistant
Dr. Andrea Rosá

Advisor's approval (Prof. Walter Binder): Date:

Contents

1 Introduction **2**

2 Background information **2**
2.1 Introduction . . . 2
2.2 Rosà et al. 2015 DSN paper . . . 2
2.3 Google Borg . . . 2
2.4 Traces contents . . . 3
2.5 Overview of traces' format . . . 3
2.6 Remark on traces size . . . 3

3 Project requirements and analysis **4**

4 Analysis methodology **4**
4.1 Introduction on Apache Spark . . . 4
4.2 Query architecture . . . 4
4.2.1 Overview . . . 4
4.2.2 Parsing table files . . . 5
4.2.3 The queries . . . 5
4.3 Query script design . . . 5
4.3.1 The "task slowdown" query script . . . 5
4.4 Ad-Hoc presentation of some analysis scripts . . . 6

5 Analysis and observations **7**
5.1 Overview of machine configurations in each cluster . . . 7
5.2 Analysis of execution time per each execution phase . . . 7
5.3 Task slowdown . . . 7
5.4 Reserved and actual resource usage of tasks . . . 7
5.5 Correlation between task events' metadata and task termination . . . 7
5.6 Correlation between task events' resource metadata and task termination . . . 10
5.7 Correlation between job events' metadata and job termination . . . 10
5.8 Mean number of tasks and event distribution per task type . . . 10
5.9 Mean number of tasks and event distribution per job type . . . 17
5.10 Probability of task successful termination given its unsuccesful events . . . 17
5.11 Potential causes of unsuccesful executions . . . 17

6 Implementation issues – Analysis limitations **17**
6.1 Discussion on unknown fields . . . 17
6.2 Limitation on computation resources required for the analysis . . . 17
6.3 Other limitations 17

7 Conclusions and future work or possible developments **17**

1 Introduction

In today's world there is an ever growing demand for efficient, large scale computations. The rising trend of "big data" put the need for efficient management of large scaled parallelized computing at an all time high. This fact also increases the demand for research in the field of distributed systems, in particular in how to schedule computations effectively, avoid wasting resources and avoid failures.

In 2011 Google released a month long data trace of its own *Borg* cluster management system[1], containing a lot of data regarding scheduling, priority management, and failures of a real production workload. This data was the foundation of the 2015 Rosá et al. paper *Understanding the Dark Side of Big Data Clusters: An Analysis beyond Failures*[2], which in its many conclusions highlighted the need for better cluster management highlighting the high amount of failures found in the traces.

In 2019 Google released an updated version of the *Borg* cluster traces[3], not only containing data from a far bigger workload due to the sheer power of Moore's law, but also providing data from 8 different *Borg* cells from datacenters all over the world. These new traces are therefore about 100 times larger than the old traces, weighing in terms of storage spaces approximately 8TiB (when compressed and stored in JSONL format)[4], requiring considerable computational power to analyze them and the implementation of special data engineering tecniques for analysis of the data.

This project aims to repeat the analysis performed in 2015 to highlight similarities and differences in workload this decade brought, and expanding the old analysis to understand even better the causes of failures and how to prevent them. Additionally, this report will provide an overview on the data engineering tecniques used to perform the queries and analyses on the 2019 traces.

2 Background information

2.1 Introduction

TBD

2.2 Rosà et al. 2015 DSN paper

In 2015, Dr. Andrea Rosà, Lydia Y. Chen, Prof. Walter Binder published a research paper titled *Understanding the Dark Side of Big Data Clusters: An Analysis beyond Failures*[2] performing several analysis on Google's 2011 Borg cluster traces. The salient conclusion of that research is that lots of computation performed by Google would eventually fail, leading to large amounts of computational power being wasted.

Our aim with this thesis is to repeat the analysis performed in 2015 on the new 2019 dataset to find similarities and differences with the previous analysis, and ulimately find if computational power is indeed wasted in this new workload as well.

2.3 Google Borg

Borg is Google's own cluster management software. Among the various cluster management services it provides, the main ones are: job queuing, scheduling, allocation, and deallocation due to higher priority computations.

The data this thesis is based on is from 8 Borg "cells" (i.e. clusters) spanning 8 different datacenters, all focused on "compute" (i.e. computational oriented) workloads. The data collection timespan matches the entire month of May 2019.

In Google's lingo a "job" is a large unit of computational workload made up of several "tasks", i.e. a number of executions of single executables running on a single machine. A job may run tasks sequentially or in parallel, and the condition for a job's succesful termination is nontrivial.

Both tasks and jobs lifecyles are represented by several events, which are encoded and stored in the trace as rows of various tables. Among the information events provide, the field "type" provides information on the execution status of the job or task. This field can have several values, which are illustrated in figure 1.

Figure 2 shows the expected transitions between event types.

Type code	Description
`QUEUE`	The job or task was marked not eligible for scheduling by Borg's scheduler, and thus Borg will move the job/task in a long wait queue
`SUBMIT`	The job or task was submitted to Borg for execution
`ENABLE`	The job or task became eligible for scheduling
`SCHEDULE`	The job or task's execution started
`EVICT`	The job or task was terminated in order to free computational resources for an higher priority job
`FAIL`	The job or task terminated its execution unsuccesfully due to a failure
`FINISH`	The job or task terminated succesfully
`KILL`	The job or task terminated its execution because of a manual request to stop it
`LOST`	It is assumed a job or task is has been terminated, but due to missing data there is insufficent information to identify when or how
`UPDATE_PENDING`	The metadata (scheduling class, resource requirements, ...) of the job/task was updated while the job was waiting to be scheduled
`UPDATE_RUNNING`	The metadata (scheduling class, resource requirements, ...) of the job/task was updated while the job was in execution

Figure 1. Overview of job and task event types.

2.4 Traces contents

The traces provided by Google contain mainly a collection of job and task events spanning a month of execution of the 8 different clusters. In addition to this data, some additional data on the machines' configuration in terms of resources (i.e. amount of CPU and RAM) and additional machine-related metadata.

Due to Google's policy, most identification related data (like job/task IDs, raw resource amounts and other text values) were obfuscated prior to the release of the traces. One obfuscation that is noteworthy in the scope of this thesis is related to CPU and RAM amounts, which are expressed respetively in NCUs (*Normalized Compute Units*) and NMUs (*Normalized Memory Units*).

NCUs and NMUs are defined based on the raw machine resource distributions of the machines within the 8 clusters. A machine having 1 NCU CPU power and 1 NMU memory size has the maximum amount of raw CPU power and raw RAM size found in the clusters. While RAM size is measured in bytes for normalization purposes, CPU power was measured in GCU (*Google Compute Units*), a proprietary CPU power measurement unit used by Google that combines several parameters like number of processors and cores, clock frequency, and architecture (i.e. ISA).

2.5 Overview of traces' format

The traces have a collective size of approximately 8TiB and are stored in a Gzip-compressed JSONL (JSON lines) format, which means that each table is represented by a single logical "file" (stored in several file segments) where each carriage return separated line represents a single record for that table.

There are namely 5 different table "files":

`machine_configs`, which is a table containing each physical machine's configuration and its evolution over time;

`instance_events`, which is a table of task events;

`collection_events`, which is a table of job events;

`machine_attributes`, which is a table containing (obfuscated) metadata about each physical machine and its evolution over time;

`instance_usage`, which contains resource (CPU/RAM) measures of jobs and tasks running on the single machines.

The scope of this thesis focuses on the tables `machine_configs`, `instance_events` and `collection_events`.

2.6 Remark on traces size

While the 2011 Google Borg traces were relatively small, with a total size in the order of the tens of gigabytes, the 2019 traces are quite challenging to analyze due to their sheer size. As stated before, the traces have a total size of 8 TiB when stored in the format provided by Google. Even when broken down to table "files", unitary sizes still reach

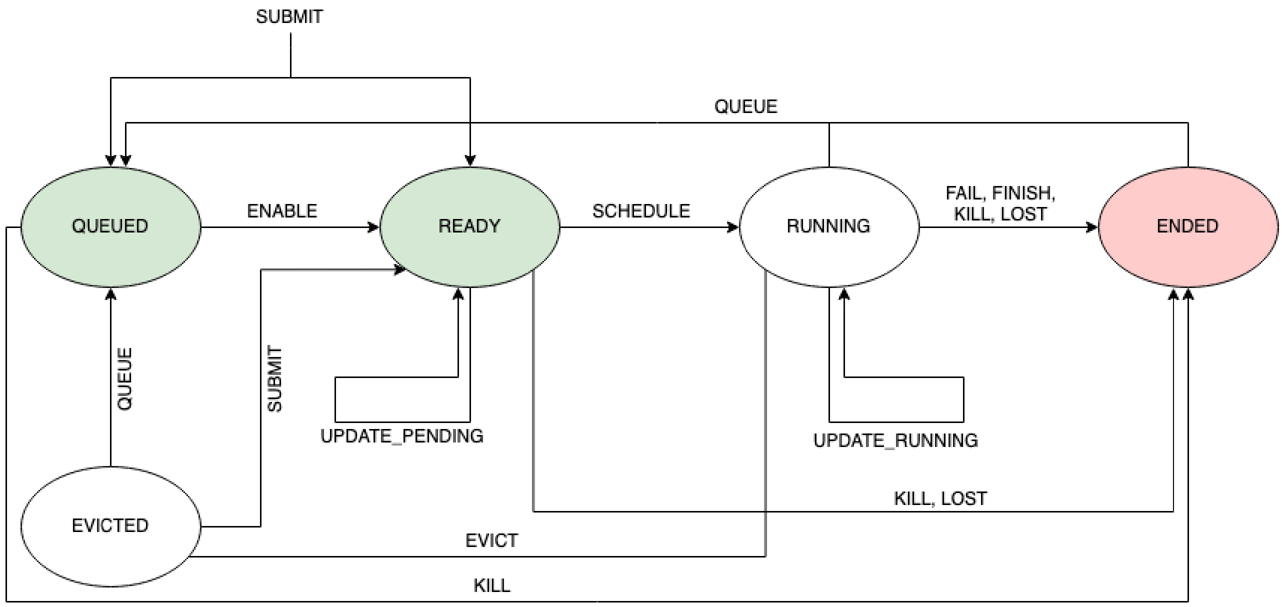


Figure 2. Typical transitions between task/job event types according to Google

the single tebibyte mark (namely for `machine_configs`, the largest table in the trace).

Due to this constraints, a careful data engineering based approach was used when reproducing the 2015 DSN paper analysis. Bleeding edge data science technologies like Apache Spark were used to achieve efficient and parallelized computations. This approach is discussed with further detail in the following section.

3 Project requirements and analysis

TBD (describe our objective with this analysis in detail)

4 Analysis methodology

Due to the inherent complexity in analyzing traces of this size, novel bleeding-edge data engineering tecniques were adopted to performed the required computations. We used the framework Apache Spark to perform efficient and parallel Map-Reduce computations. In this section, we discuss the technical details behind our approach.

4.1 Introduction on Apache Spark

Apache Spark is a unified analytics engine for large-scale data processing. In layman's terms, Spark is really useful to parallelize computations in a fast and streamlined way.

In the scope of this thesis, Spark was used essentially as a Map-Reduce framework for computing aggregated results on the various tables. Due to the sharded nature of table "files", Spark is able to spawn a thread per file and run computations using all processors on the server machines used to run the analysis.

Spark is also quite powerful since it provides automated thread pooling services, and it is able to efficiently store and cache intermediate computation on secondary storage without any additional effort required from the data engineer. This feature was especially useful due to the sheer size of the analyzed data, since the computations required to store up to 1TiB of intermediate data on disk.

The chosen programming language for writing analysis scripts was Python. Spark has very powerful native Python bindings in the form of the *PySpark* API, which were used to implement the various queries.

4.2 Query architecture

4.2.1 Overview

In general, each query written to execute the analysis follows a general Map-Reduce template.

Traces are first read, then parsed, and then filtered by performing selections, projections and computing new derived fields. After this preparation phase, the trace records are often passed through a `groupby()` operation, which by choosing one or many record fields sorts all the records into several "bins" containing records with matching values for the selected fields. Then, a map operation is applied to each bin in order to derive some aggregated property value for each grouping. Finally, a reduce operation is applied to either further aggregate those computed properties or to generate an aggregated data structure for storage purposes.

4.2.2 Parsing table files

As stated before, table "files" are composed of several Gzip-compressed shards of JSONL record data. The specification for the types and constraints of each record is outlined by Google in the form of a protobuffer specification file found in the trace release package[5]. This file was used as the oracle specification and was a critical reference for writing the query code that checks, parses and carefully sanitizes the various JSONL records prior to actual computations.

The JSONL encoding of traces records is often performed with non-trivial rules that required careful attention. One of these involved fields that have a logically-wise "zero" value (i.e. values like "0" or the empty string). For these values the key-value pair in the JSON object is outright omitted. When reading the traces in Apache Spark is therefore necessary to check for this possibility and insert back the omitted record attributes.

4.2.3 The queries

Most queries use only two or three fields in each trace records, while the original table records often are made of a couple of dozen fields. In order to save memory during the query, a projection is often applied to the data by the means of a `.map()` operation over the entire trace set, performed using Spark's RDD API.

Another operation that is often necessary to perform prior to the Map-Reduce core of each query is a record filtering process, which is often motivated by the presence of incomplete data (i.e. records which contain fields whose values is unknown). This filtering is performed using the `.filter()` operation of Spark's RDD API.

The core of each query is often a `groupby()` followed by a `map()` operation on the aggregated data. The `groupby()` groups the set of all records into several subsets of records each having something in common. Then, each of this small clusters is reduced with a `map()` operation to a single record. The motivation behind this way of computing data is that for the analysis in this thesis it is often necessary to analyze the behaviour w.r.t. time of either task or jobs by looking at their events. These queries are therefore implemented by `groupby()`-ing records by task or job, and then `map()`-ing each set of event records sorting them by time and performing the desired computation on the obtained chronological event log.

Sometimes intermediate results are saved in Spark's parquet format in order to compute and save intermediate results beforehand.

4.3 Query script design

In this section we aim to show the general complexity behind the implementations of query scripts by explaining in detail some sampled scripts to better appreciate their behaviour.

4.3.1 The "task slowdown" query script

One example of analysis script with average complexity and a pretty straightforward structure is the pair of scripts `task_slowdown.py` and `task_slowdown_table.py` used to compute the "task slowdown" tables (namely the tables in figure 7).

"Slowdown" is a task-wise measure of wasted execution time for tasks with a `FINISH` termination type. It is computed as the total execution time of the task divided by the execution time actually needed to complete the task (i.e. the total time of the last execution attempt, successful by definition).

The analysis requires to compute the mean task slowdown for each task priority value, and additionally compute the percentage of tasks with successful terminations per priority. The query therefore needs to compute the execution time of each execution attempt for each task, determine if each task has successful termination or not, and finally combine this data to compute slowdown, mean slowdown and ultimately the final table found in figure 7.

Figure 3 shows a schematic representation of the query structure.

The query first starts reading the `instance_events` table, which contains (among other data) all task event logs containing properties, event types and timestamps. As already explained in the previous section, the logical table file

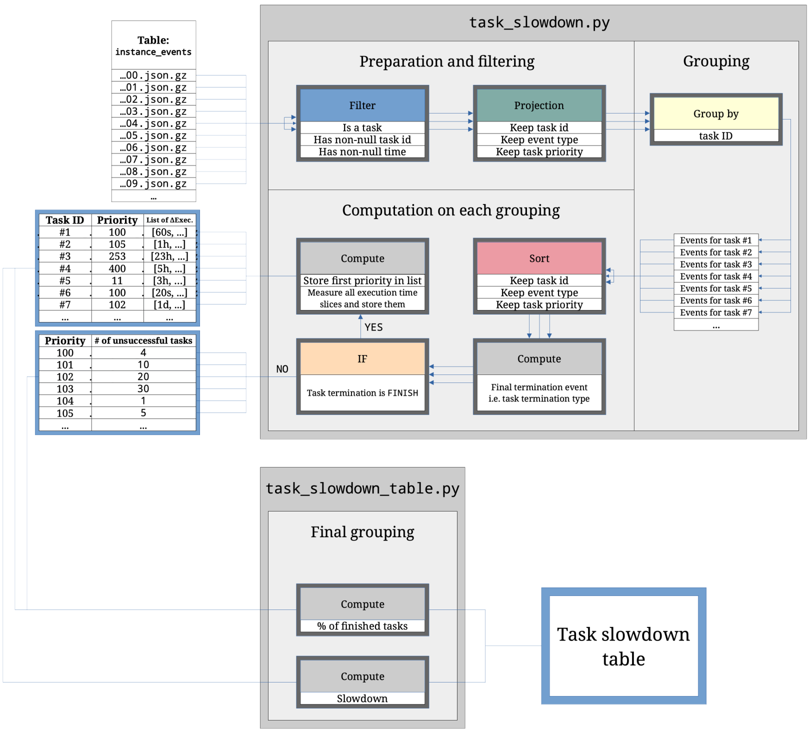


Figure 3. Diagram of the script used for the "task slowdown" query.

is actually stored as several Gzip-compressed JSONL shards. This is very useful for processing purposes, since Spark is able to parse and load in memory each shard in parallel, i.e. using all processing cores on the server used to run the queries.

After loading the data, a selection and a projection operation are performed in the preparation phase so as to "clean up" the records and fields that are not needed, leaving only useful information to feed in the "group by" phase. In this query, the selection phase removes all records that do not represent task events or that contain an unknown task ID or a null event timestamp. In the 2019 traces it is quite common to find incomplete records, since the log process is unable to capture the sheer amount of events generated by all jobs in a exact and deterministic fashion.

Then, after the preparation stage is complete, the task event records are grouped in several bins, one per task ID. Performing this operation the collection of unsorted task event types is rearranged to form groups of task events all relating to a single task.

These obtained collections of task events are then sorted by timestamp and processed to compute intermediate data relating to execution attempt times and task termination counts. After the task events are sorted, the script iterates over the events in chronological order, storing each execution attempt time and registering all execution termination types by checking the event type field. The task termination is then equal to the last execution termination type, following the definition originally given in the 2015 Rosá et al. DSN paper.

If the task termination is determined to be unsuccessful, the tally counter of task terminations for the matching task property is increased. Otherwise, all the task termination attempt time deltas are returned. Tallies and time deltas are saved in an intermediate time file for fine-grained processing.

Finally, the `task_slowdown_table.py` processes this intermediate results to compute the percentage of successful tasks per execution and computing slowdown values given the previously computed execution attempt time deltas. Finally, the mean of the computed slowdown values is computed resulting in the clear and coincise tables found in figure 7.

4.4 Ad-Hoc presentation of some analysis scripts

TBD (with diagrams)

5 Analysis and observations

5.1 Overview of machine configurations in each cluster

Refer to figure 4.

Observations:

- machine configurations are definitely more varied than the ones in the 2011 traces
- some clusters have more machine variability

5.2 Analysis of execution time per each execution phase

Refer to figures **??** and 5.

Observations:

- Across all cluster almost 50% of time is spent in "unknown" transitions, i.e. there are some time slices that are related to a state transition that Google says are not "typical" transitions. This is mostly due to the trace log being intermittent when recording all state transitions.
- 80% of the time spent in KILL and LOST is unknown. This is predictable, since both states indicate that the job execution is not stable (in particular LOST is used when the state logging itself is unstable)
- From the absolute graph we see that the time "wasted" on non-finish terminated jobs is very significant
- Execution is the most significant task phase, followed by queuing time and scheduling time ("ready" state)
- In the absolute graph we see that a significant amount of time is spent to re-schedule evicted jobs ("evicted" state)
- Cluster A has unusually high queuing times

5.3 Task slowdown

Refer to figure 7

Observations:

- Priority values are different from 0-11 values in the 2011 traces. A conversion table is provided by Google;
- For some priorities (e.g. 101 for cluster D) the relative number of finishing task is very low and the mean slowdown is very high (315). This behaviour differs from the relatively homogeneous values from the 2011 traces.
- Some slowdown values cannot be computed since either some tasks have a 0ns execution time or for some priorities no tasks in the traces terminate successfully. More raw data on those exception is in Jupyter.
- The % of finishing jobs is relatively low comparing with the 2011 traces.

5.4 Reserved and actual resource usage of tasks

Refer to figures 9 and 10.

Observations:

- Most (mesasured and requested) resources are used by killed job, even more than in the 2011 traces.
- Behaviour is rather homogeneous across datacenters, with the exception of cluster G where a lot of LOST-terminated tasks acquired 70% of both CPU and RAM

5.5 Correlation between task events' metadata and task termination

Refer to figures 11, 12, and 13.

Observations:

- No smooth curves in this figure either, unlike 2011 traces

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	8729	1.639218%
1.000000	0.500000	124234	23.329891%
0.591797	0.333496	103013	19.344801%
0.259277	0.166748	78078	14.662260%
0.708984	0.333496	55801	10.478864%
0.386719	0.333496	36237	6.804943%
0.958984	0.500000	31151	5.849843%
0.708984	0.666992	29594	5.557454%
0.386719	0.166748	27011	5.072393%
1.000000	1.000000	12286	2.307187%
0.591797	0.166748	9902	1.859496%
1.000000	0.250000	7550	1.417814%
0.958984	1.000000	3552	0.667030%
0.259277	0.333496	3024	0.567877%
0.591797	0.666992	1000	0.187790%
0.259277	0.083374	634	0.119059%
0.958984	0.250000	600	0.112674%
0.500000	0.062500	54	0.010141%
0.500000	0.250000	34	0.006385%
0.479492	0.250000	12	0.002253%
0.708984	0.250000	6	0.001127%
0.591797	0.250000	4	0.000751%
0.708984	0.500000	2	0.000376%
0.479492	0.500000	2	0.000376%

(a) All clusters

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	1377	1.623170%
0.591797	0.333496	29487	34.758469%
1.000000	0.500000	13440	15.842705%
0.708984	0.333496	12495	14.728764%
0.386719	0.333496	9057	10.676144%
0.386719	0.166748	5265	6.206238%
0.708984	0.666992	4608	5.431784%
1.000000	1.000000	4446	5.240823%
0.591797	0.166748	2484	2.928071%
0.958984	0.500000	1143	1.347337%
0.958984	1.000000	654	0.770917%
1.000000	0.250000	366	0.431431%
0.479492	0.250000	6	0.007073%
0.708984	0.250000	6	0.007073%

(b) A cluster

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	134	0.264812%
0.591797	0.333496	16184	31.982926%
1.000000	0.500000	9790	19.347061%
0.708984	0.333496	8448	16.694992%
0.958984	0.500000	5502	10.873088%
0.708984	0.666992	3832	7.572823%
1.000000	1.000000	2214	4.375321%
0.591797	0.166748	2152	4.252796%
0.386719	0.333496	816	1.612584%
0.958984	1.000000	618	1.221296%
0.591797	0.666992	500	0.988103%
0.386719	0.166748	412	0.814197%

(c) Cluster B

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	1466	2.274208%
0.259277	0.166748	15754	24.439204%
0.386719	0.333496	11104	17.225652%
0.591797	0.333496	10404	16.139741%
0.958984	0.500000	6634	10.291334%
1.000000	0.500000	5654	8.771059%
0.386719	0.166748	3580	5.553660%
0.708984	0.666992	2900	4.498774%
1.000000	1.000000	2736	4.244361%
1.000000	0.250000	2132	3.307375%
0.958984	1.000000	766	1.188297%
0.708984	0.333496	620	0.961807%
0.958984	0.250000	600	0.930781%
0.591797	0.166748	112	0.173746%

(d) Cluster C

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	498	0.794309%
0.591797	0.333496	28394	45.288376%
0.386719	0.333496	8402	13.401174%
0.259277	0.166748	8020	12.791885%
0.386719	0.166748	5806	9.260559%
0.708984	0.666992	4380	6.986092%
0.708984	0.333496	3924	6.258772%
0.591797	0.166748	2548	4.064055%
0.259277	0.333496	426	0.679469%
1.000000	0.500000	292	0.465739%
0.591797	0.250000	4	0.006380%
0.708984	0.500000	2	0.003190%

(e) Cluster D

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	536	0.671915%
0.259277	0.166748	38452	48.202377%
0.708984	0.333496	11786	14.774608%
0.958984	0.500000	8646	10.838389%
0.708984	0.666992	7606	9.534674%
1.000000	0.500000	5586	7.002457%
0.386719	0.166748	4470	5.603470%
0.259277	0.333496	1268	1.589530%
0.259277	0.083374	634	0.794765%
0.591797	0.333496	324	0.406158%
1.000000	0.250000	268	0.335957%
1.000000	1.000000	138	0.172993%
0.500000	0.062500	54	0.067693%
0.500000	0.250000	4	0.005014%

(f) Cluster E

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	1432	2.299958%
1.000000	0.500000	41340	66.396839%
0.708984	0.333496	6878	11.046866%
0.591797	0.333496	5564	8.936430%
0.958984	0.500000	2172	3.488484%
0.386719	0.166748	1544	2.479843%
0.708984	0.666992	1244	1.998008%
1.000000	0.250000	792	1.272044%
0.958984	1.000000	536	0.860878%
0.386719	0.333496	398	0.639234%
1.000000	1.000000	344	0.552504%
0.500000	0.250000	18	0.028910%

(g) Cluster F

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	1566	2.261568%
0.259277	0.166748	15852	22.892958%
1.000000	0.500000	11808	17.052741%
0.708984	0.333496	7968	11.507134%
0.591797	0.333496	7830	11.307839%
0.386719	0.166748	4690	6.773150%
0.708984	0.666992	4258	6.149269%
0.958984	0.500000	4196	6.059731%
0.386719	0.333496	3864	5.580267%
0.591797	0.166748	2606	3.763503%
1.000000	0.250000	2100	3.032754%
0.259277	0.333496	1330	1.920744%
0.958984	1.000000	778	1.123563%
1.000000	1.000000	378	0.545896%
0.500000	0.250000	12	0.017330%
0.479492	0.250000	6	0.008665%
0.479492	0.500000	2	0.002888%

(h) Cluster G

CPU (NCU)	RAM (NMU)	Machine count	% Machines
Unknown	Unknown	1720	2.933251%
1.000000	0.500000	36324	61.946178%
0.591797	0.333496	4826	8.230158%
0.708984	0.333496	3682	6.279205%
0.958984	0.500000	2858	4.873973%
0.386719	0.333496	2596	4.427163%
1.000000	1.000000	2030	3.461919%
1.000000	0.250000	1892	3.226577%
0.386719	0.166748	1244	2.121491%
0.708984	0.666992	766	1.306320%
0.591797	0.666992	500	0.852689%
0.958984	1.000000	200	0.341076%

(i) Cluster H

Figure 4. Overview of machine configurations in terms of CPU and RAM resources for each cluster

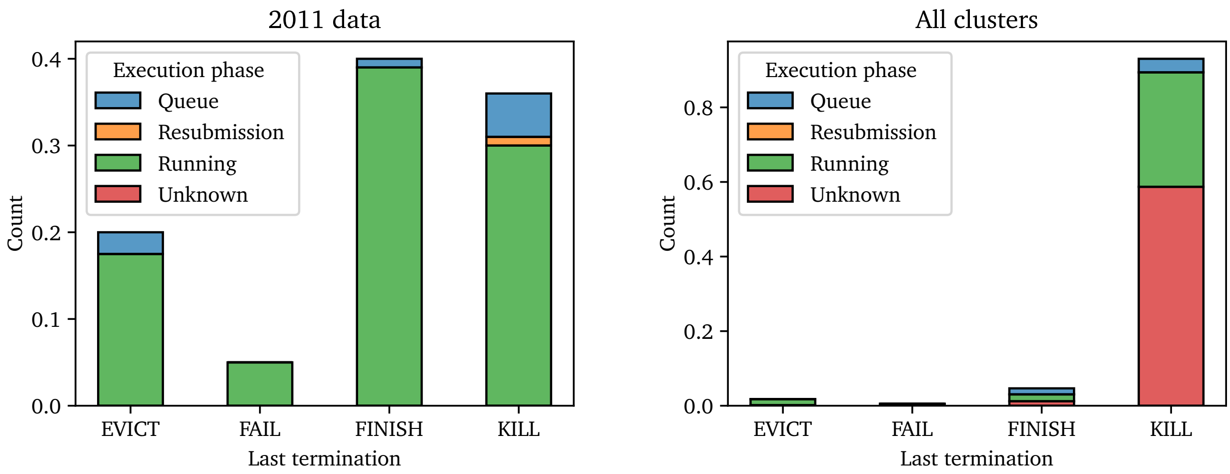


Figure 5. Relative task time (in milliseconds) spent in each execution phase w.r.t. task termination in 2011 and 2019 traces. X axis shows task termination type, Y axis shows total time % spent. Colors break down the time in execution phases. "Unknown" execution times are 2019 specific and correspond to event time transitions that are not consider "typical" by Google.

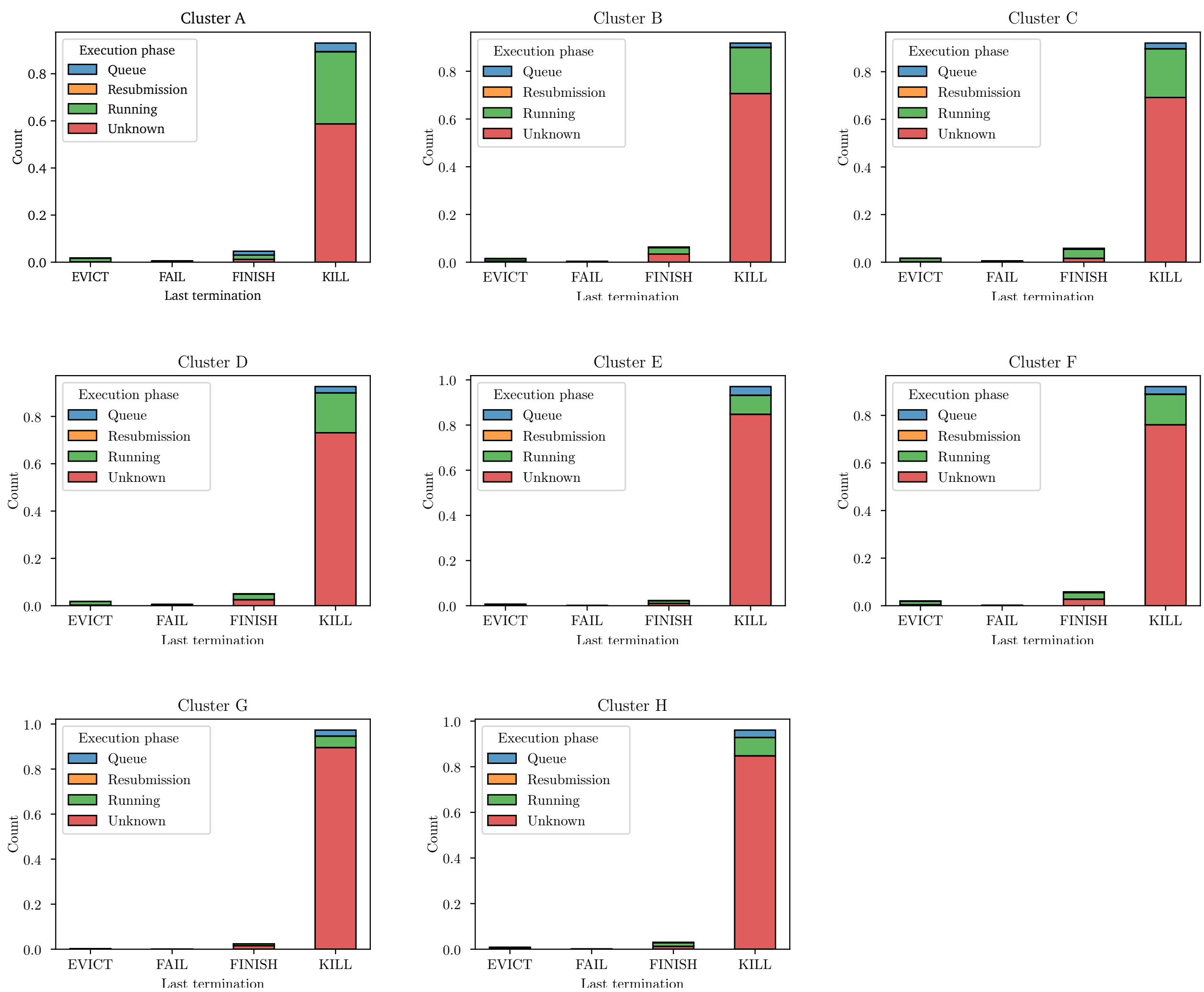


Figure 6. Relative task time (in milliseconds) spent in each execution phase w.r.t. clusters in the 2019 trace. Refer to figure 5 for axes description.

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slowdown
0	53.80%	2845	1767	3.37
1	67.44%	3598	2939	2.58
2	90.78%	1835	1782	1.15
3	95.62%	9683	8294	3.39
4	78.05%	2006	1890	1.69
5	100%	58	58	1
6	77.99%	567	567	1.02
8	45.48%	1159	1151	1.01
9	23.35%	504	496	1.07

(a) 2011 data

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slowdown
0	42.86%	1373.0	0.0	1.136770
3	0.0%	–	0.0	–
5	0.0%	–	0.0	–
10	0.0%	–	0.0	–
19	0.0%	–	0.0	–
24	0.0%	–	0.0	–
25	1.31%	86732.0	22.0	11.772172
50	0.0%	–	0.0	–
80	0.0%	–	0.0	–
100	0.0%	–	0.0	–
101	5.2%	65233.0	673.0	36.358841
102	0.0%	–	0.0	–
103	1.05%	8210.0	28.0	1.257530
105	22.9%	3651.0	616.0	1.733089
107	18.51%	1025.0	308.0	1.017332
114	0.07%	29364.0	2.0	1.003503
115	1.74%	10059.0	22.0	3.461721
116	3.03%	18226.0	71.0	1.102756
117	0.0%	2430.0	1.0	1.000000
118	1.28%	15072.0	163.0	3.340741
119	4.49%	19449.0	280.0	5.326446
170	0.0%	–	0.0	–
197	0.0%	–	0.0	–
199	0.0%	–	0.0	–
200	13.54%	54789.0	24.0	6.684155
201	0.0%	–	0.0	–
205	0.0%	–	0.0	–
210	0.0%	–	0.0	–
214	0.0%	–	0.0	–
215	0.0%	–	0.0	–
220	0.0%	–	0.0	–
360	3.36%	788069.0	42.0	2.241646
400	0.0%	–	0.0	–
450	1.15%	1182248.0	197.0	1.068893
500	0.0%	–	0.0	–

(b) 2019 data

Figure 7. Mean task slowdown for each cluster and each task **Priority**

- The behaviour of curves for 7a (priority) is almost the opposite of 2011, i.e. in-between priorities have higher kill rates while priorities at the extremum have lower kill rates. This could also be due bt the inherent distribution of job terminations;
- Event execution time curves are quite different than 2011, here it seems there is a good correlation between short task execution times and finish event rates, instead of the U shape curve in 2015 DSN
- In figure 12 cluster behaviour seems quite uniform
- Machine concurrency seems to play little role in the event termination distribution, as for all concurrency factors the kill rate is at 90%.

5.6 Correlation between task events' resource metadata and task termination

5.7 Correlation between job events' metadata and job termination

Refer to figures 14, 15, and 16.

Observations:

- Behaviour between cluster varies a lot
- There are no "smooth" gradients in the various curves unlike in the 2011 traces
- Killed jobs have higher event rates in general, and overall dominate all event rates measures
- There still seems to be a correlation between short execution job times and successfull final termination, and likewise for kills and higher job terminations
- Across all clusters, a machine locality factor of 1 seems to lead to the highest success event rate

5.8 Mean number of tasks and event distribution per task type

Refer to figure 17.

Observations:

- The mean number of events per task is an order of magnitude higher than in the 2011 traces

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
24	0.0%	–	–	–
25	0.33%	5769.0	1203.0	82.97
100	0.0%	–	–	–
101	81.92%	63305.0	6346.0	30.80
102	0.0%	–	–	–
103	14.99%	3074.0	3033.0	1.13
105	57.68%	1666.0	1750.0	1.08
107	53.93%	1022.0	1031.0	1.02
114	0.0%	–	–	–
115	4.11%	2041.0	2042.0	1.00
116	13.05%	4443.0	4443.0	1.03
117	0.0%	–	–	–
118	11.91%	1817.0	1814.0	1.00
119	21.26%	2250.0	2877.0	1.50
170	0.0%	–	–	–
200	27.21%	4546.0	16845.0	4.12
205	0.0%	–	–	–
210	0.0%	–	–	–
214	0.0%	–	–	–
215	0.0%	–	–	–
360	0.62%	514181.0	400580.0	2.92
400	0.0%	–	–	–
450	2.2%	686817.0	653878.0	1.14
500	0.0%	–	–	–

(a) Cluster A

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
0	45.19%	1351.0	1467.0	1.18
25	0.02%	10696.0	4121.0	133.48
80	0.0%	–	–	–
100	0.0%	–	–	–
101	66.48%	6069.0	5402.0	433.41
103	0.11%	19430.0	14897.0	1.65
105	0.46%	934421.0	392431.0	2.41
107	0.0%	–	–	–
114	0.68%	32949.0	30470.0	1.00
115	4.12%	25585.0	107089.0	5.92
116	8.32%	29290.0	29017.0	1.11
117	0.0%	–	–	–
118	0.31%	2776.0	2776.0	1.00
119	0.2%	193081.0	304469.0	2.56
170	0.0%	–	–	–
199	0.0%	–	–	–
200	30.92%	182604.0	466329.0	9.71
205	0.0%	–	–	–
210	0.0%	–	–	–
214	0.0%	–	–	–
215	0.0%	–	–	–
360	3.5%	1048245.0	495124.0	1.61
450	0.61%	1579367.0	1529555.0	1.06

(b) Cluster B

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
0	50.89%	933.0	1002.0	1.11
3	0.0%	–	–	–
10	0.0%	–	–	–
25	22.47%	171281.0	4551.0	8.19
100	0.0%	–	–	–
101	52.63%	6271.0	2498.0	421.49
103	0.01%	3344.0	7444.0	2.79
105	0.02%	1202141.0	863764.0	1.37
107	0.0%	7033.0	93102.0	14.71
114	0.02%	3148.0	3142.0	1.01
115	0.28%	14729.0	27168.0	1.98
116	0.01%	2846.0	2851.0	1.02
117	93.17%	2144.0	2144.0	1.00
118	0.0%	1114.0	1112.0	1.10
119	2.22%	573740.0	242446.0	2.04
170	0.0%	–	–	–
200	3.61%	352603.0	357993.0	4.14
205	0.0%	–	–	–
210	0.0%	–	–	–
214	0.0%	–	–	–
215	0.0%	–	–	–
360	4.37%	769284.0	442062.0	2.06
450	1.51%	1390175.0	1319771.0	1.07

(c) Cluster C

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
0	26.52%	1398.0	1469.0	1.12
5	0.0%	–	–	–
25	16.29%	21432.0	4037.0	65.68
100	0.0%	–	–	–
101	45.31%	8391.0	3317.0	315.95
103	0.0%	6791.0	6647.0	1.07
105	0.05%	825749.0	924081.0	2.90
107	0.0%	300532.0	174837.0	1.55
114	0.0%	–	–	–
115	5.19%	12598.0	26142.0	2.19
116	0.13%	9268.0	10955.0	1.28
117	85.71%	10969.0	10969.0	1.00
118	0.05%	24041.0	30599.0	2.05
119	0.44%	184484.0	172746.0	3.02
197	0.0%	–	–	–
199	0.0%	–	–	–
200	6.53%	279565.0	349364.0	5.51
205	0.0%	–	–	–
210	0.0%	–	–	–
214	0.0%	–	–	–
215	0.0%	–	–	–
360	1.59%	650116.0	390151.0	2.48
450	0.61%	938727.0	523665.0	1.33

(d) Cluster D

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
0	42.81%	802.0	1127.0	1.44
25	5.34%	32247.0	38946.0	2.68
100	0.0%	–	–	–
101	0.02%	30603.0	27726.0	1.12
103	0.02%	76294.0	48552.0	3.16
105	0.4%	106677.0	64190.0	14.75
107	0.0%	–	–	–
114	0.0%	–	–	–
115	0.03%	67237.0	65369.0	1.00
116	0.0%	–	–	–
117	0.0%	–	–	–
118	0.0%	–	–	–
119	0.46%	62123.0	83322.0	10.31
170	0.0%	–	–	–
200	1.96%	231639.0	414149.0	8.54
201	0.0%	–	–	–
205	0.0%	–	–	–
210	0.0%	–	–	–
215	0.0%	–	–	–
220	0.0%	–	–	–
360	37.16%	611504.0	439280.0	2.87
450	0.55%	803792.0	824467.0	1.11

(e) Cluster E

Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
0	45.21%	2929.0	2973.0	1.09
25	0.65%	184518.0	34096.0	2.23
100	0.0%	–	–	–
101	40.3%	8160.0	10083.0	323.86
103	0.06%	46444.0	47234.0	1.17
105	0.22%	1111530.0	1173594.0	1.55
107	0.06%	80151.0	78835.0	1.01
114	0.01%	677.0	677.0	1.00
115	3.65%	121345.0	252663.0	5.09
116	0.0%	–	–	–
117	0.0%	15875.0	15875.0	1.00
118	0.0%	30045.0	25492.0	1.00
119	31.35%	154196.0	68833.0	7.61
200	3.65%	297168.0	492372.0	5.94
201	0.0%	–	–	–
360	7.42%	963351.0	569428.0	2.17
450	0.99%	1115783.0	1113282.0	1.02

(f) Cluster F

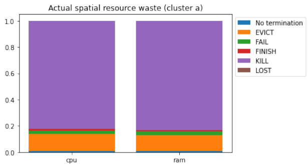
Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
0	33.61%	3010.0	3317.0	1.14
25	0.23%	61708.0	12156.0	8.69
50	0.0%	–	–	–
100	0.0%	–	–	–
101	96.47%	133953.0	7448.0	19.38
103	0.03%	118310.0	112746.0	1.27
105	0.2%	8271.0	8214.0	1.00
107	0.0%	–	–	–
114	0.0%	–	–	–
115	7.63%	34424.0	56315.0	1.80
117	0.0%	–	–	–
118	48.97%	608526.0	474729.0	3.88
119	0.09%	124006.0	231395.0	3.17
170	0.0%	–	–	–
200	26.75%	115953.0	399050.0	14.57
360	1.62%	786594.0	488025.0	2.12
450	2.74%	1204747.0	1188251.0	1.04

(g) Cluster G

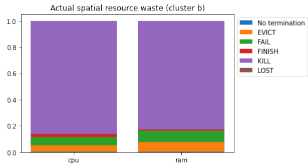
Priority	% finished	Mean resp. (last evt.)	Mean resp. (all evts.)	Mean slow-down
0	27.74%	5663.0	6211.0	1.12
19	0.0%	–	–	–
25	1.04%	304870.0	283847.0	3.06
101	100.0%	34063.0	12250.0	76.44
103	0.48%	272635.0	92894.0	1.26
105	1.43%	611763.0	393762.0	4.21
107	0.0%	–	–	–
115	5.12%	28907.0	28907.0	1.00
116	1.04%	633.0	42105.0	73.45
117	0.0%	656.0	656.0	1.00
118	1.0%	197687.0	139121.0	1.95
119	0.15%	139907.0	15558.0	7.30
200	2.7%	298799.0	470783.0	5.80
201	0.0%	–	–	–
220	0.0%	–	–	–
360	4.43%	838719.0	397301.0	2.02
450	0.54%	1470577.0	1411397.0	1.05

(h) Cluster H

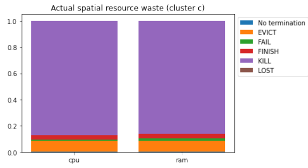
Figure 8. Mean task slowdown for each cluster and each task **Priority**



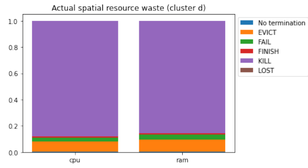
(a) Cluster A



(b) Cluster B



(c) Cluster C



(d) Cluster D

Task termination	**% CPU**	**% RAM**
No termination	0.6972%	1.0447%
Evict	13.4392%	11.8184%
Fail	2.2792%	2.8387%
Finish	1.3963%	1.1066%
Kill	82.1791%	83.1826%
Lost	0.0091%	0.0091%

(e) Cluster A (exact values)

Task termination	**% CPU**	**% RAM**
No termination	0.2582%	0.4637%
Evict	4.8340%	7.3120%
Fail	6.2950%	8.3841%
Finish	2.5877%	1.2231%
Kill	86.0215%	82.6144%
Lost	0.0036%	0.0027%

(f) Cluster B (exact values)

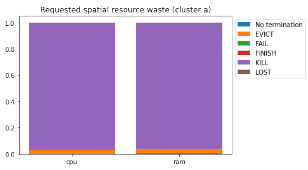
Task termination	**% CPU**	**% RAM**
No termination	0.3376%	0.3812%
Evict	8.2099%	8.0454%
Fail	1.2294%	2.0809%
Finish	2.9399%	3.3249%
Kill	87.2740%	86.1588%
Lost	0.0093%	0.0088%

(g) Cluster C (exact values)

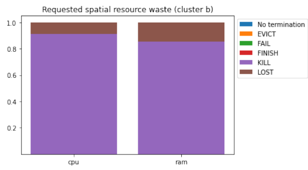
Task termination	**% CPU**	**% RAM**
No termination	0.4995%	0.4822%
Evict	7.6002%	9.0656%
Fail	3.0288%	3.9214%
Finish	0.8666%	0.8914%
Kill	88.0011%	85.6364%
Lost	0.0039%	0.0030%

(h) Cluster D (exact values)

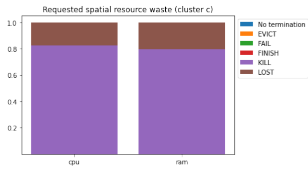
Figure 9. Relative usage of CPU and RAM resources w.r.t. final task termination.



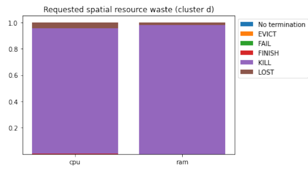
(a) Cluster A



(b) Cluster B



(c) Cluster C



(d) Cluster D

Task termination	**% CPU**	**% RAM**
No termination	0.033962%	0.193674%
Evict	2.838362%	3.399075%
Fail	0.058335%	0.069755%
Finish	0.000102%	0.000151%
Kill	96.661332%	95.799104%
Lost	0.407908%	0.538242%

(e) Cluster A (exact values)

Task termination	**% CPU**	**% RAM**
No termination	0.000094%	0.000191%
Evict	0.003365%	0.004696%
Fail	0.003061%	0.004965%
Finish	0.012696%	0.017647%
Kill	91.094839%	85.573746%
Lost	8.885947%	14.398756%

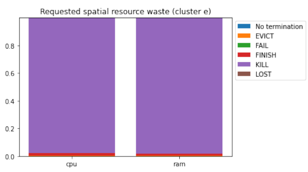
(f) Cluster B (exact values)

Task termination	**% CPU**	**% RAM**
No termination	0.000105%	0.000221%
Evict	0.008618%	0.006991%
Fail	0.001261%	0.001459%
Finish	0.015047%	0.017003%
Kill	82.483146%	79.698011%
Lost	17.491823%	20.276314%

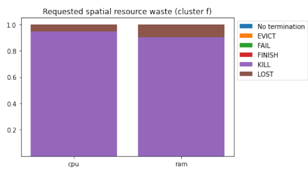
(g) Cluster C (exact values)

Task termination	**% CPU**	**% RAM**
No termination	0.000948%	0.000128%
Evict	0.046057%	0.006352%
Fail	0.023703%	0.002770%
Finish	0.095353%	0.012975%
Kill	95.468127%	97.927565%
Lost	4.365813%	2.050210%

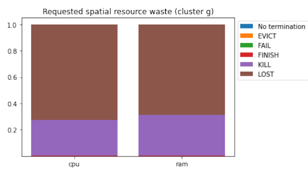
(h) Cluster D (exact values)



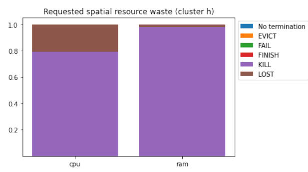
(i) Cluster E



(j) Cluster F



(k) Cluster G



(l) Cluster H

Task termination	**% CPU**	**% RAM**
No termination	0.015102%	0.016472%
Evict	0.362088%	0.321274%
Fail	0.051373%	0.047377%
Finish	1.672195%	1.310360%
Kill	97.899179%	98.304482%
Lost	0.000063%	0.000034%

(m) Cluster E (exact values)

Task termination	**% CPU**	**% RAM**
No termination	0.000114%	0.000306%
Evict	0.007986%	0.013466%
Fail	0.000913%	0.002064%
Finish	0.013296%	0.021751%
Kill	94.396548%	90.227868%
Lost	5.581144%	9.734546%

(n) Cluster F (exact values)

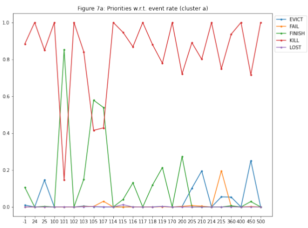
Task termination	**% CPU**	**% RAM**
No termination	0.001283%	0.000748%
Evict	0.034040%	0.025278%
Fail	0.004384%	0.003918%
Finish	0.176091%	0.166656%
Kill	27.376816%	30.954255%
Lost	72.407386%	68.849146%

(o) Cluster G (exact values)

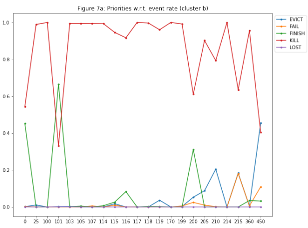
Task termination	**% CPU**	**% RAM**
No termination	0.000148%	0.000022%
Evict	0.006021%	0.000751%
Fail	0.000858%	0.000144%
Finish	0.015642%	0.001873%
Kill	78.910066%	97.713322%
Lost	21.067264%	2.283888%

(p) Cluster H (exact values)

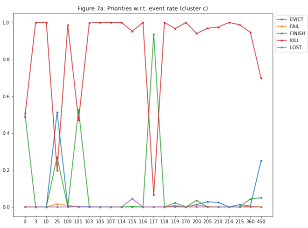
Figure 10. Relative request of CPU and RAM resources prior to tasks' execution w.r.t. final task termination.



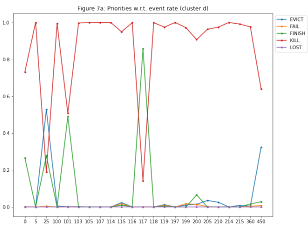
(a) Cluster A



(b) Cluster B

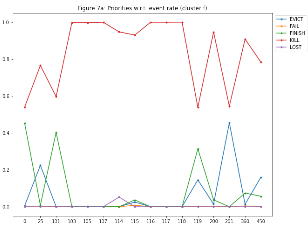


(c) Cluster C

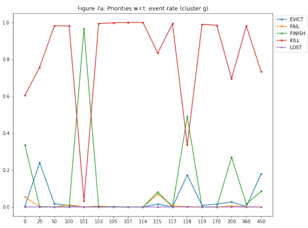


(d) Cluster D

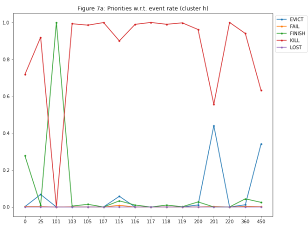
(e) Cluster E



(f) Cluster F

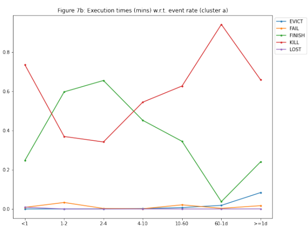


(g) Cluster G

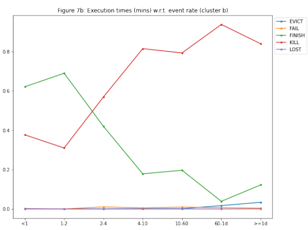


(h) Cluster H

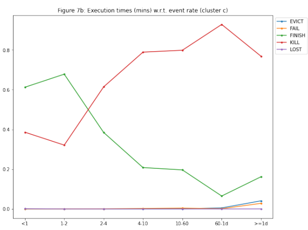
Figure 11. Task event rates vs. task priority and final task termination



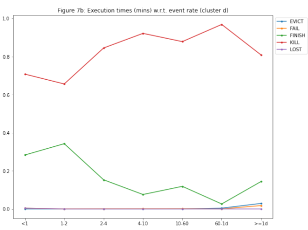
(a) Cluster A



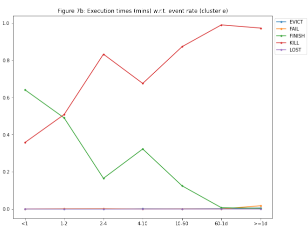
(b) Cluster B



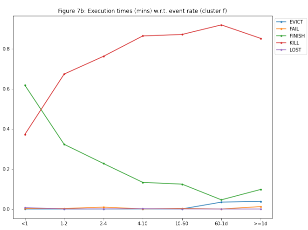
(c) Cluster C



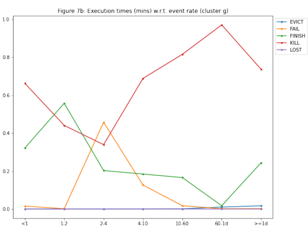
(d) Cluster D



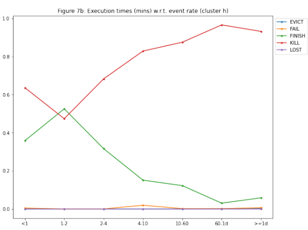
(e) Cluster E



(f) Cluster F

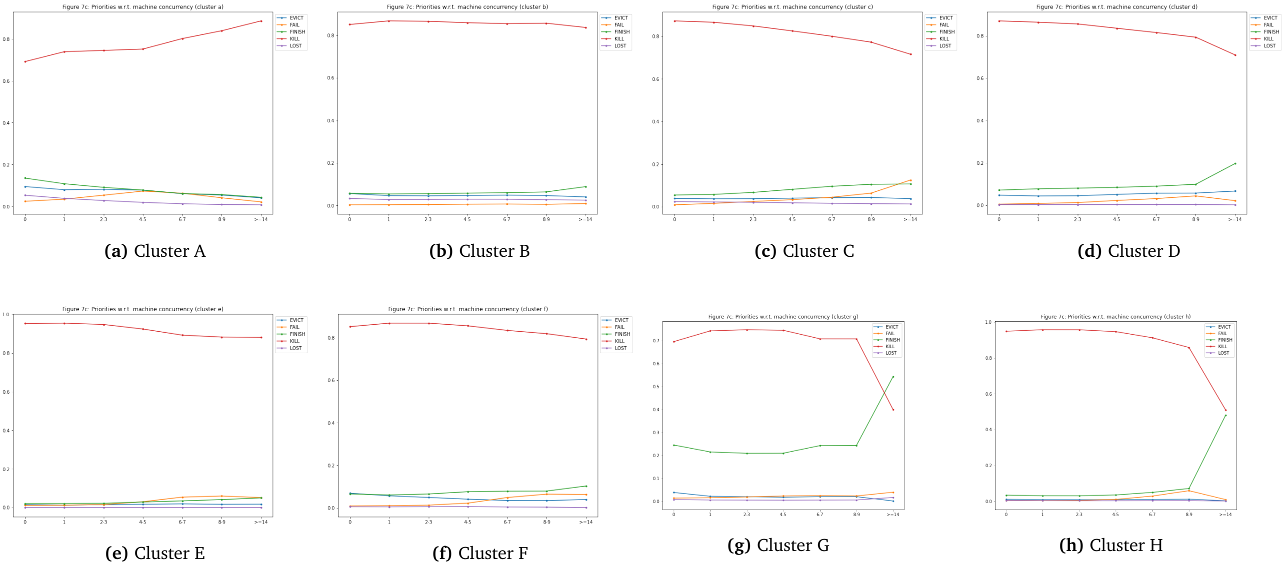


(g) Cluster G



(h) Cluster H

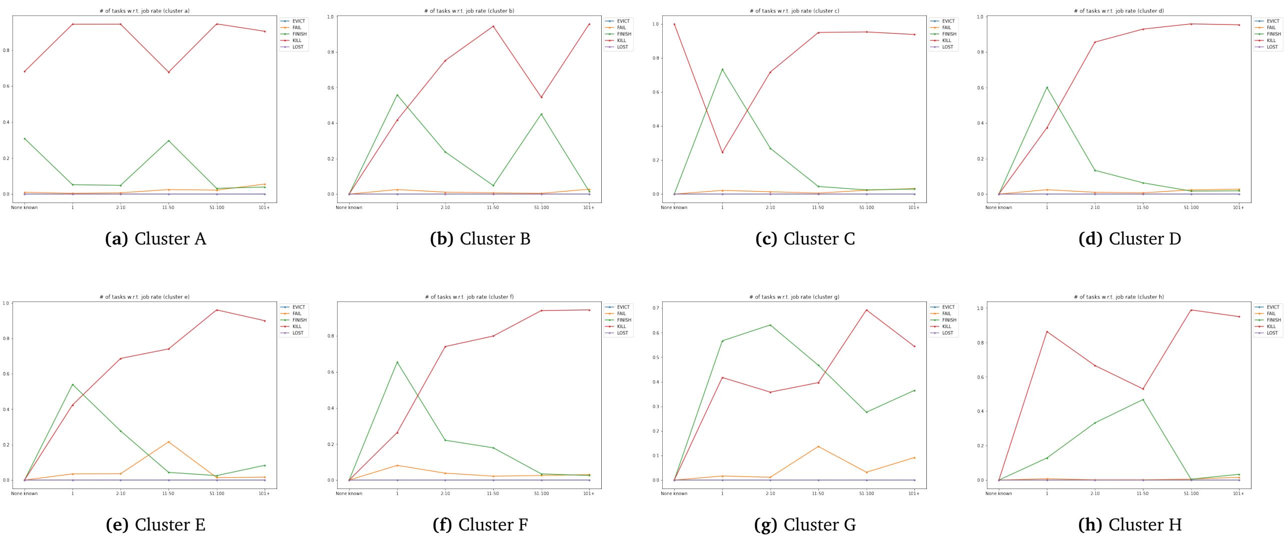
Figure 12. Task event rates vs. event execution time and final task termination



(a) Cluster A **(b)** Cluster B **(c)** Cluster C **(d)** Cluster D

(e) Cluster E **(f)** Cluster F **(g)** Cluster G **(h)** Cluster H

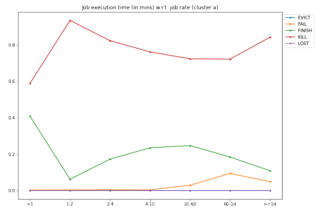
Figure 13. Task event rates vs. machine concurrency and final task termination



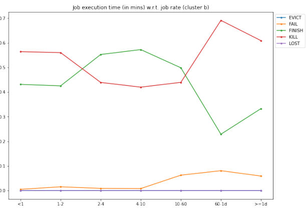
(a) Cluster A **(b)** Cluster B **(c)** Cluster C **(d)** Cluster D

(e) Cluster E **(f)** Cluster F **(g)** Cluster G **(h)** Cluster H

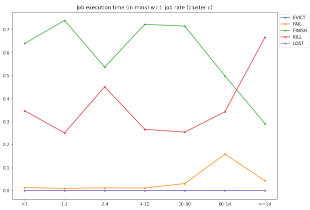
Figure 14. Job event rates vs. job size and final job termination



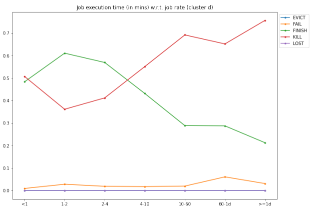
(a) Cluster A



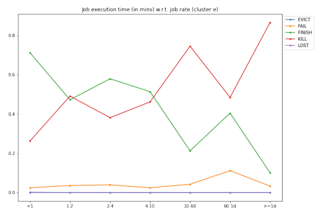
(b) Cluster B



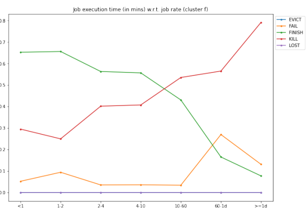
(c) Cluster C



(d) Cluster D



(e) Cluster E

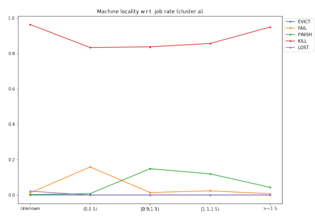


(f) Cluster F

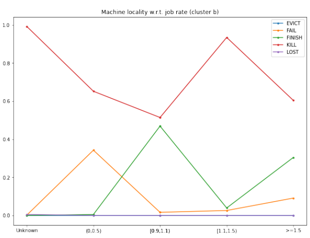
(g) Cluster G

(h) Cluster H

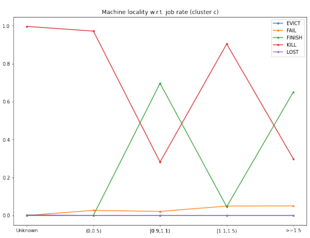
Figure 15. Job event rates vs. event execution time and final job termination



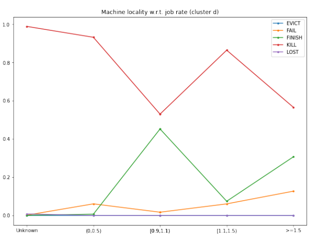
(a) Cluster A



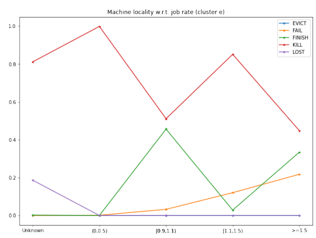
(b) Cluster B



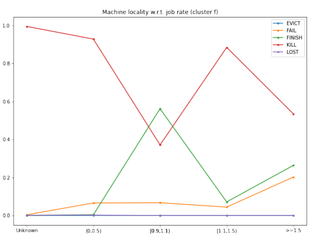
(c) Cluster C



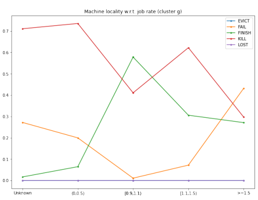
(d) Cluster D



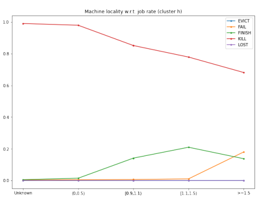
(e) Cluster E



(f) Cluster F



(g) Cluster G



(h) Cluster H

Figure 16. Job event rates vs. machine locality and final job termination

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	58.0	27.395925	2.349579	0.213859	0.003412	3.395996	0.089576
FINISH	9.0	12.405370	0.019321	0.003779	2.153432	0.008150	0.008989
FAIL	108.0	50.039556	0.287778	11.061864	0.002098	0.467656	0.053144
LOST	7.0	8.847145	0.083348	0.001821	0.384190	1.329910	1.007933
EVICT	2924.0	428.550689	73.693595	0.768553	0.000179	28.766164	0.845501
No termination	84.0	14.818523	0.000000	0.000000	0.000000	0.000000	0.000000

(a) Cluster A

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	60.0	40.901041	3.351496	0.276305	0.003656	5.541079	0.033457
FINISH	20.0	17.277596	0.020444	0.020628	2.942579	0.011640	0.016278
FAIL	260.0	86.772419	0.518061	19.656798	0.000560	0.675392	0.088523
LOST	14.0	25.690455	0.257231	0.007420	1.928351	3.515436	2.015153
EVICT	1578.0	345.705559	64.816518	0.240214	0.000000	17.961539	1.028401
No termination	32.0	13.018130	0.000000	0.000000	0.000000	0.000000	0.000000

(b) Cluster B

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	32.0	24.230887	1.533237	0.116082	0.003994	3.799111	0.013670
FINISH	18.0	15.242628	0.017929	0.012701	2.470654	0.006020	0.006414
FAIL	156.0	187.030894	0.772823	48.445773	2.035378	0.756015	0.133687
LOST	28.0	22.385446	0.411365	0.007569	1.412201	2.751353	1.998665
EVICT	1748.0	404.108669	73.715527	1.812816	0.000166	22.908022	0.546198
No termination	96.0	21.315166	0.000000	0.000000	0.000000	0.000000	0.000000

(c) Cluster C

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	32.0	29.953873	1.960134	0.150521	0.002385	4.682411	0.016156
FINISH	18.0	23.105615	0.058651	0.019051	3.789050	0.009785	0.018699
FAIL	269.0	228.004975	0.496316	58.968210	0.809520	2.040396	0.324754
LOST	20.0	17.065721	0.014760	0.003577	0.079289	4.636283	1.999794
EVICT	1478.0	323.366130	62.000510	0.700268	0.000373	14.057514	0.627592
No termination	103.0	27.867403	0.000000	0.000000	0.000000	0.000000	0.000000

(d) Cluster D

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	258.0	55.877475	1.287917	0.056909	0.000185	12.159880	0.054997
FINISH	14.0	11.976806	0.013879	0.008435	1.998677	0.008241	0.026641
FAIL	138.0	450.526937	0.457703	111.471047	0.000000	0.455705	0.187991
LOST	14.0	11.899908	0.000000	0.000000	0.033976	3.131007	1.792164
EVICT	310.0	84.645189	11.780754	0.106119	0.000090	5.790960	0.654955
No termination	34.0	7.349165	0.000000	0.000000	0.000000	0.000000	0.000000

(e) Cluster E

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	162.0	45.039557	0.384065	0.098430	0.001178	9.804287	0.037783
FINISH	20.0	19.899709	0.019381	0.003510	3.007839	0.097934	0.023707
FAIL	220.0	164.043073	0.279352	39.257407	0.000023	1.549795	0.203997
LOST	36.0	25.002219	0.011815	0.000909	0.149586	7.283534	2.000428
EVICT	510.0	302.262347	23.973621	0.192394	0.000094	45.979997	0.374789
No termination	24.0	7.784905	0.000000	0.000000	0.000000	0.000000	0.000000

(f) Cluster F

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	641.00	130.054143	6.909204	0.135073	0.000033	25.275769	0.131106
FINISH	18.00	105.240418	0.015228	0.001655	14.153775	0.004879	0.158300
FAIL	40.00	40.121553	0.016111	8.592728	0.000000	0.338883	0.011310
LOST	4602.25	576.384120	1.931330	0.360515	48.094421	35.596567	3.534335
EVICT	2015.00	555.574743	77.429054	0.303127	0.000000	58.299330	0.653819
No termination	30.00	9.503553	0.000000	0.000000	0.000000	0.000000	0.000000

(g) Cluster G

Task termination	# Evts. 95% p.tile	# Evts. mean	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
KILL	388.0	74.425542	0.633338	0.169666	0.000231	17.172624	0.062799
FINISH	22.0	23.978294	0.023700	0.014129	3.632529	0.011111	0.028482
FAIL	487.0	170.153701	0.600483	37.599942	0.000000	2.866647	0.343806
LOST	386.4	94.666667	1.493333	2.400000	0.573333	14.040000	3.480000
EVICT	206.0	75.658064	6.732544	0.837154	0.000000	7.164722	0.421745
No termination	18.0	8.123506	0.000000	0.000000	0.000000	0.000000	0.000000

(h) Cluster H

Figure 17. Mean number of tasks and event distribution per task type

- Generally speaking, the event type with higher mean is the termination event for the task
- The # evts mean is higher than the sum of all other event type means, since it appears there are a lot more non-termination events in the 2019 traces.

5.9 Mean number of tasks and event distribution per job type

Refer to figure 18.

Observations:

- Again the mean number of tasks is significantly higher than the 2011 traces, indicating a higher complexity of workloads
- Cluster A has no evicted jobs
- The number of events is however lower than the event means in the 2011 traces

5.10 Probability of task successful termination given its unsuccesful events

Refer to figure 19.

Observations:

- Behaviour is very different from cluster to cluster
- There is no easy conclusion, unlike in 2011, on the correlation between succesful probability and # of events of a specific type.
- Clusters B, C and D in particular have very unsmooth lines that vary a lot for small # evts differences. This may be due to an uneven distribution of # evts in the traces.

5.11 Potential causes of unsuccesful executions

TBD

6 Implementation issues – Analysis limitations

6.1 Discussion on unknown fields

TBD

6.2 Limitation on computation resources required for the analysis

TBD

6.3 Other limitations …

TBD

7 Conclusions and future work or possible developments

TBD

References

[1] Abhishek Verma, Luis Pedrosa, Madhukar R. Korupolu, et al. "Large-scale cluster management at Google with Borg". In: *Proceedings of the European Conference on Computer Systems (EuroSys)*. Bordeaux, France, 2015.

[2] Andrea Rosà, Lydia Y. Chen, and Walter Binder. "Understanding the Dark Side of Big Data Clusters: An Analysis beyond Failures". In: *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. 2015, pp. 207–218. DOI: 10.1109/DSN.2015.37.

[3] Muhammad Tirmazi, Adam Barker, Nan Deng, et al. "Borg: the Next Generation". In: *EuroSys'20*. Heraklion, Crete, 2020.

Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	92.359436	174.3	23.263951	3.454474	23.047597	34.565608	0.707709
EVICT	-1.000000	-1.0	NaN	NaN	NaN	NaN	NaN
FAIL	90.792728	499.0	0.694942	0.683556	0.085957	1.849587	0.009730
FINISH	1.187092	1.0	0.004696	0.001341	1.072623	0.024396	0.000952
KILL	16.533171	10.0	1.045419	0.073867	0.461387	1.188720	0.044610
LOST	223.206593	1689.6	0.000000	0.000000	0.000000	1.034082	0.974598

(a) Cluster A

Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	112.422759	169.8	34.681161	0.711242	13.379533	38.794188	0.780483
EVICT	1.000000	1.0	1.000000	0.000000	0.000000	0.000000	0.000000
FAIL	74.367804	374.0	2.003355	1.993765	0.266584	4.944145	0.034526
FINISH	6.304299	10.0	0.022380	0.008476	2.349304	0.012729	0.006484
KILL	69.853370	234.0	1.696449	0.157833	0.613748	3.008678	0.012092
LOST	320.020202	459.8	0.000000	0.000000	0.000000	2.959946	1.996875

(b) Cluster B

Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	96.399561	100.0	55.276973	7.552906	23.848867	41.578669	0.664107
EVICT	1.000000	1.0	1.000829	0.000000	0.000000	0.000415	0.000000
FAIL	41.982301	200.0	3.483606	0.997592	0.376438	3.998369	0.046439
FINISH	1.991485	1.0	0.021806	0.016914	1.565034	0.017401	0.001803
KILL	110.680808	652.0	0.627334	0.059076	0.656426	2.266794	0.006258
LOST	38.870091	48.6	0.000031	0.000311	0.000000	2.620721	1.833872

(c) Cluster C

Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	103.889987	120.00	41.421532	7.604808	18.179476	47.603502	0.661826
EVICT	1.000000	1.00	1.000000	0.000000	0.000000	0.000000	0.000000
FAIL	43.355682	250.00	6.111993	0.948602	0.531390	6.497784	0.041077
FINISH	2.109260	2.00	0.268375	0.012614	1.723392	0.018567	0.005052
KILL	89.647948	283.00	1.013114	0.054374	0.283313	3.255675	0.006664
LOST	271.441748	2620.75	0.000000	0.000000	0.000000	5.938069	1.647084

(d) Cluster D

Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	350.929407	596.0	7.204391	2.074423	0.126290	46.646065	0.378274
EVICT	1.000000	1.0	1.000000	0.000000	0.000000	0.000000	0.000000
FAIL	23.081125	25.0	0.246529	0.665546	0.716720	1.588119	0.066467
FINISH	7.776085	2.0	0.018677	0.029073	1.934488	0.020929	0.064920
KILL	88.790215	309.0	0.706293	0.028618	0.461084	7.572301	0.029122
LOST	5.374150	5.0	0.000000	0.000000	0.000000	3.234494	1.813924

(e) Cluster E

Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	217.718640	379.4	4.304676	1.315021	4.971122	48.118465	0.464429
EVICT	1.000000	1.0	1.000000	0.000000	0.000000	0.000000	0.000000
FAIL	17.161251	8.0	0.621327	0.546356	0.426265	7.559244	0.034773
FINISH	2.940843	2.0	0.014704	0.051014	1.669860	0.162042	0.002623
KILL	103.888843	361.0	0.182630	0.063914	0.416684	5.824311	0.014161
LOST	3736.500000	18823.4	0.001491	0.000038	0.000000	6.298140	1.429604

(f) Cluster F

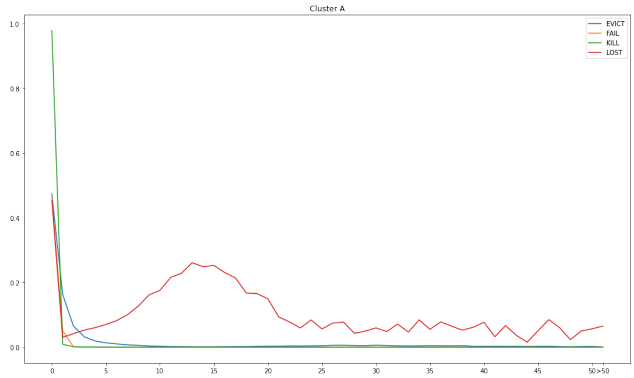
Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	342.090034	599.10	14.184405	0.626186	23.836017	46.002917	0.735801
EVICT	1.000000	1.00	1.000000	0.000000	0.000000	0.000000	0.000000
FAIL	51.834803	250.00	0.555532	3.334848	0.607560	20.351992	0.176242
FINISH	8.519166	36.00	0.001733	0.629809	1.759677	0.005452	0.004575
KILL	37.054914	100.00	5.687172	0.064640	0.080370	19.166260	0.059132
LOST	190.500000	358.35	0.000000	0.000000	0.000000	1.994751	1.994751

(g) Cluster G

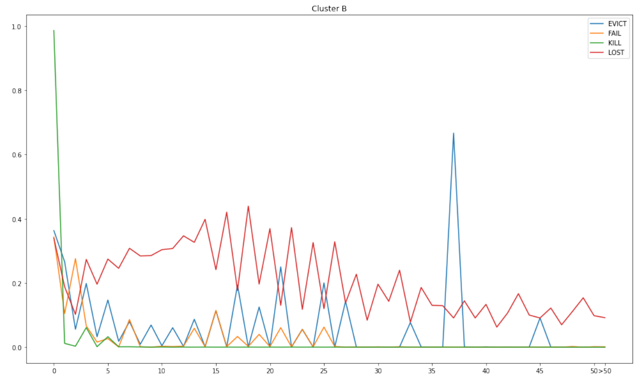
Job termination	# Tasks mean	# Tasks 95% p.tile	# EVICT Evts. mean	# FAIL Evts. mean	# FINISH Evts. mean	# KILL Evts. mean	# LOST Evts. mean
No termination	321.133053	546.9	3.470078	0.907801	3.316902	44.535824	0.315120
EVICT	1.000000	1.0	1.000000	0.000000	0.000000	0.000000	0.000000
FAIL	20.504293	1.0	0.114090	2.300036	0.980635	12.833466	0.046833
FINISH	4.278193	14.0	0.005406	0.152814	1.778038	0.013567	0.012663
KILL	11.022705	3.0	0.235500	0.102899	0.287701	11.336956	0.031148
LOST	3.400000	10.6	0.000000	0.000000	0.000000	0.235294	1.705882

(h) Cluster H

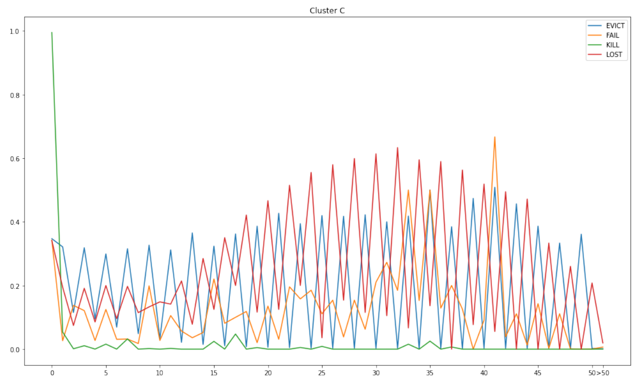
Figure 18. Mean number of tasks and event distribution per job type



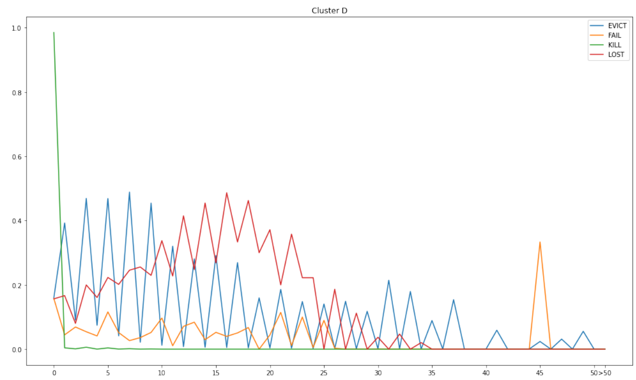
(a) Cluster A



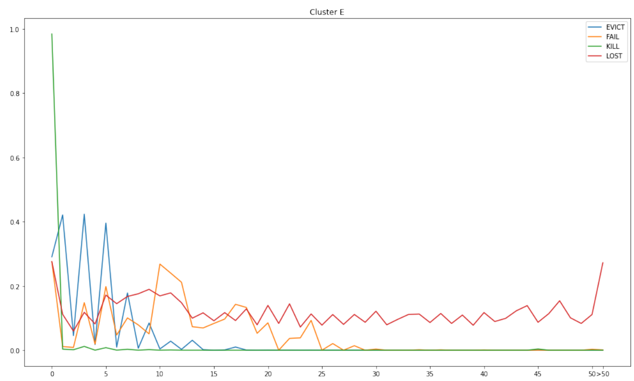
(b) Cluster B



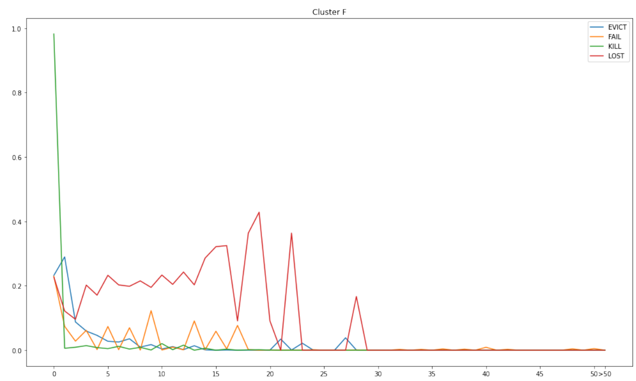
(c) Cluster C



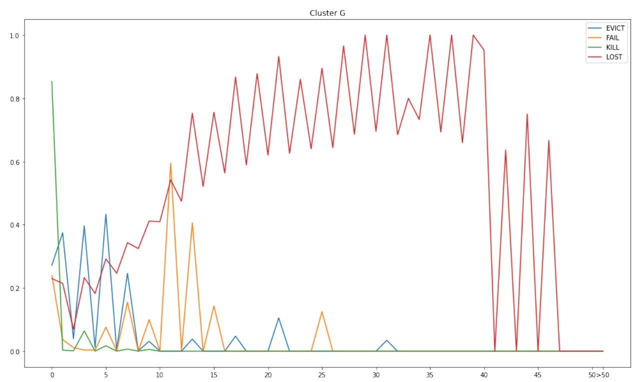
(d) Cluster D



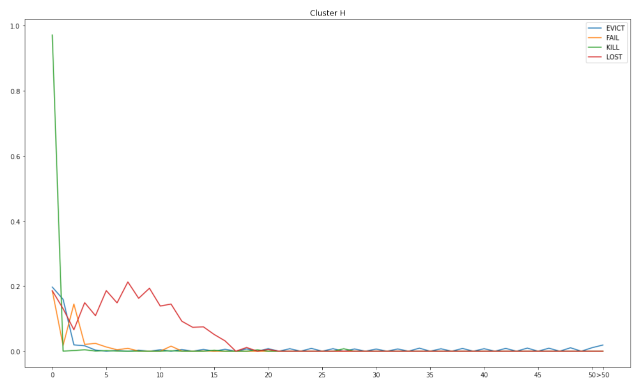
(e) Cluster E



(f) Cluster F



(g) Cluster G



(h) Cluster H

Figure 19. Conditional probability of task success given a number of specific unsuccesful events observed, i.e. eviction, fail, kill or lost.

[4] John Wilkes. *Google cluster-usage traces v3.pdf*. Aug. 2020. URL: https://drive.google.com/file/d/10r6cnJ5cJ89fPWCgj7j4LtLBqYN9RiI9/view.

[5] Nan Deng. *Google 2019 Borg traces protobuffer specification*. Aug. 2020. URL: https://github.com/google/cluster-data/blob/master/clusterdata_trace_format_v3.proto.