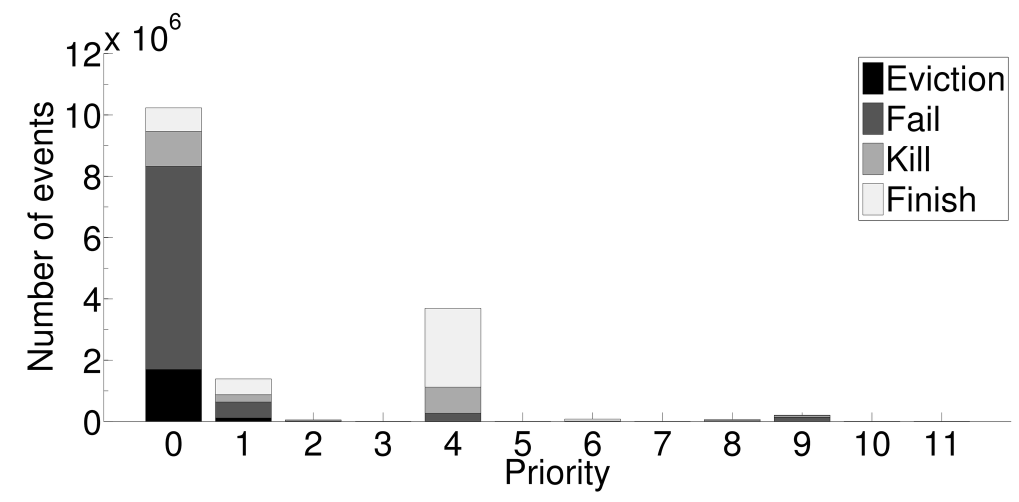
When Things Turn Sour at Big Data Clusters: Understanding Unsuccessful Executions

Andrea Rosà*, Lydia Y. Chen†, Walter Binder*
*Università della Svizzera Italiana, Faculty of Informatics, Lugano, Switzerland
†IBM Research Zurich Lab, Rueschlikon, Switzerland

Goal

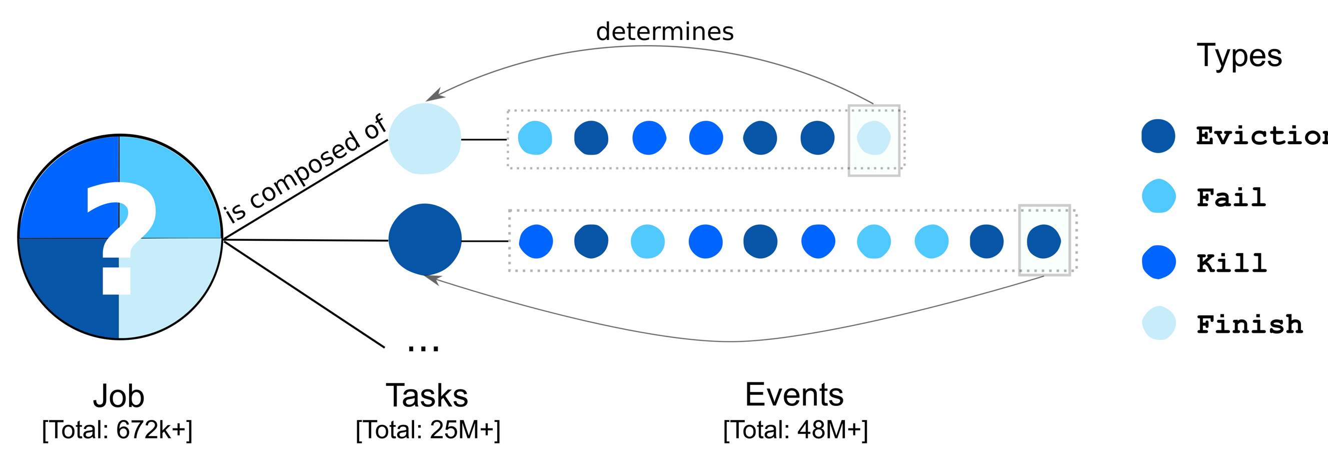
- Provide a better understanding of **unsuccessful executions**:
  - their **performance impact**
  - their **characteristics**
  - their **relationship** with **application** and **machine attributes**
- In **multi-purpose** and **multi-tenancy datacenters**

Motivations



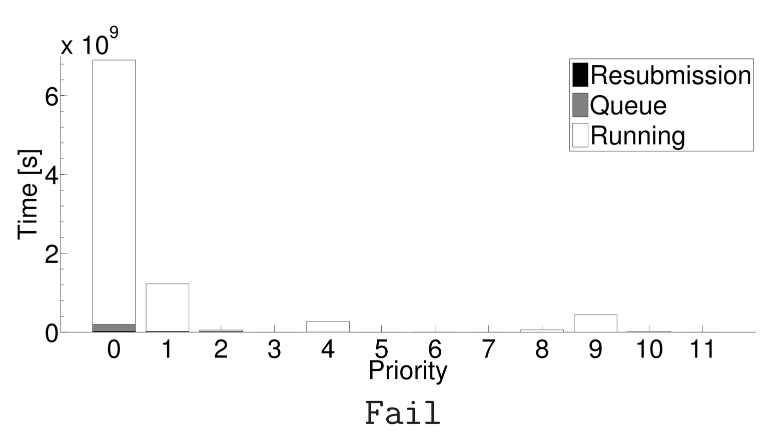
- **Failures** are **very frequent** in large-scale datacenters [1]
  - **Software** and **hardware** failures turn into critical **performance impediment**
- Big clusters are **complex**:
  - Jobs with **high number** of **tasks fanout**; tasks subjected to **multiple events**
  - Tasks have **different priority** and **resource constraints**
  - **Several types** of unsuccessful executions: `eviction`, `fail`, `kill`
  - Resulting analysis is **challenging**
- At **all priorities**, a lot of **unsuccessful executions**:
  - `Eviction`, `fail` and `kill` happen at all priorities
  - Both jobs and tasks have **high probability to fail**
  - Non negligible **resource waste** and **slowdown** of the application performance

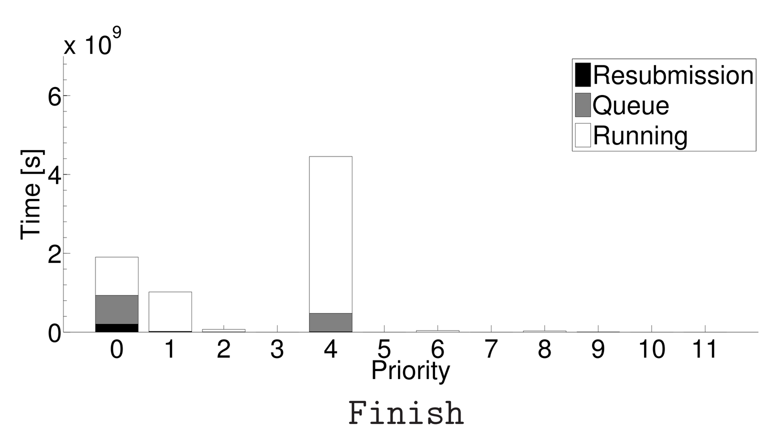
Data set



- **Google cluster trace** [2]
  - **Jobs** composed of multiple **tasks**, which experience multiple **events**
  - Jobs, tasks, and events are classified into different **types**, depending on their outcome:
    - **Unsuccessful**: `eviction`, `fail`, `kill`
    - **Successful**: `finish`
    - Tasks are classified based on their final event
    - Job type is given directly by the trace
    - We focus on **unsuccessful jobs, tasks, and events**
  - **Sizeable** datacenter:
    - 12k+ machines; 672k+ jobs; 25M+ tasks; 48M+ events
  - A lot of information provided:
    - Task **priority**
    - **Arrival**, **scheduling** and **ending** timestamps
    - **Machine** equipped resources
    - Task **requested** and **used** resources
  - **Heterogeneous** and **dynamic** workload [3]

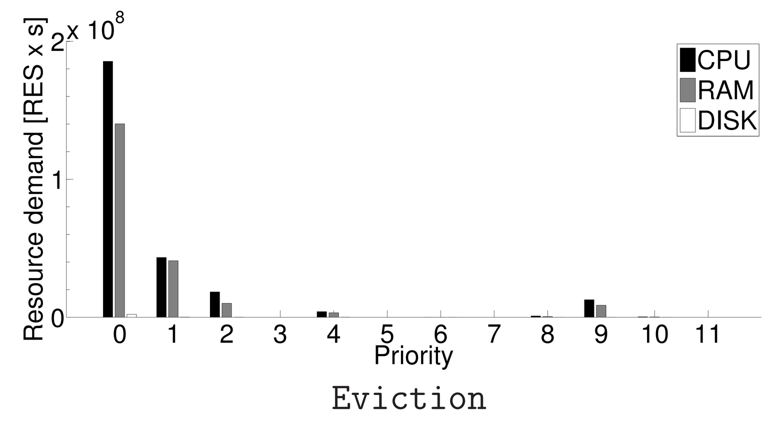
Time waste

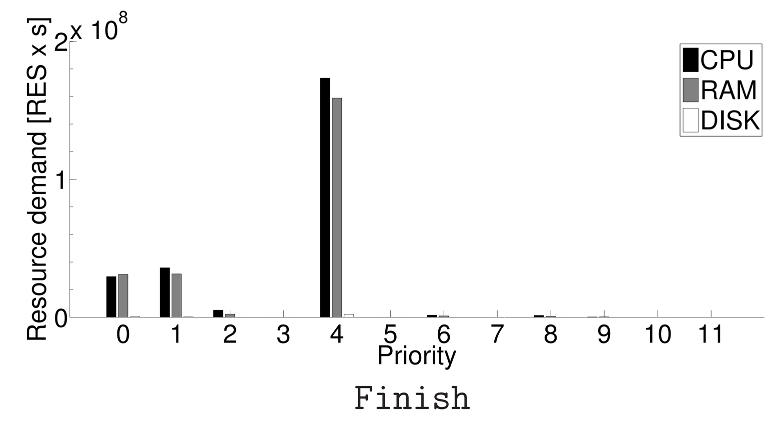




- Task execution time divided into three time intervals:
  - **Resubmission** time (from previous failure to arrival)
  - **Queue** time (from arrival to scheduling)
  - **Running** time (from scheduling to ending)
- Large amount of **wasted time**:
  - Successful tasks only use **a portion** of computational time
  - A lot of time is spent into **useless operations**

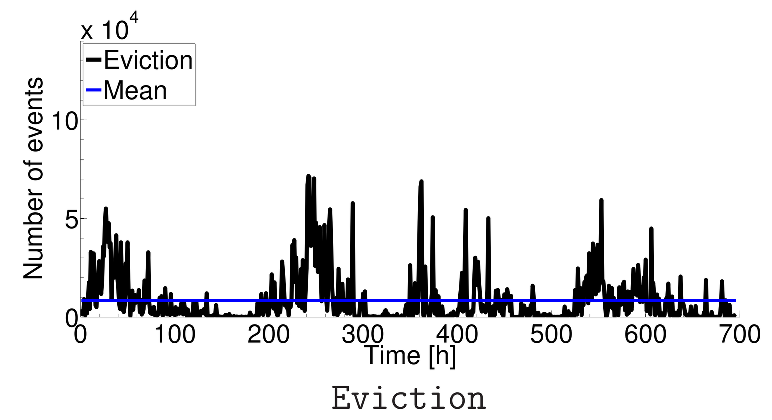
Resource waste

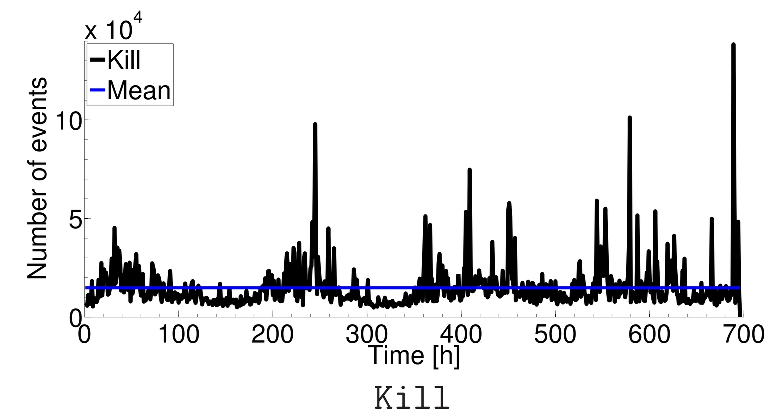




- Analysis on **resource demand**:
  - Definition: average amount of used or requested resources × running time
  - Two kinds of resource demand:
    - **Requested** demand: how many resources have been **allocated** to tasks, and how long
    - **Used** demand: how many resources have been **used** by tasks, and how long
  - Three types of resources: `CPU`, `RAM`, `DISK`
  - At **all priorities**, high **wasted resource demand**

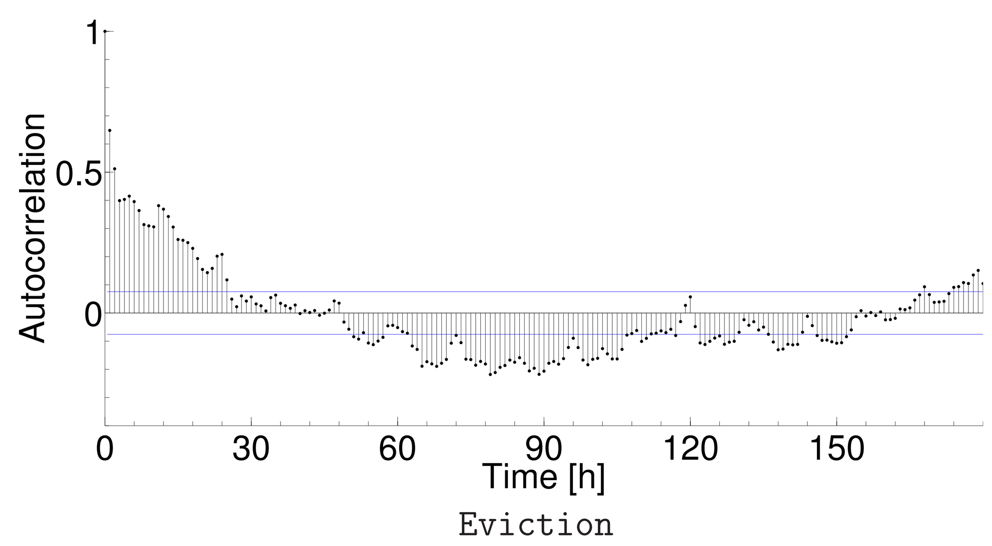
Time-varying behavior





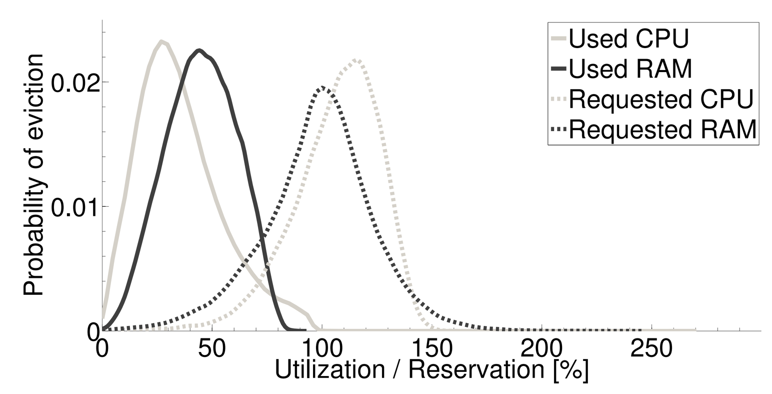
- Unsuccessful events **over time**
  - **How often** and **when** jobs and tasks fail
  - **Unstable** trends of unsuccessful executions
  - **Fitting** of **inter-type** times shows that unsuccessful executions can be approximated by heavy-tailed theoretical distributions

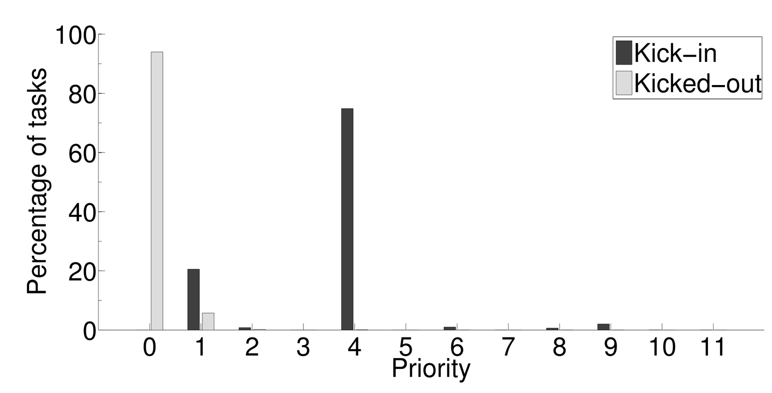
Autocorrelation



- **Dependency** of unsuccessful executions **with themselves**
  - Analysis of **autocorrelation functions** for each type of unsuccessful executions
  - **Strong time dependencies** in first few hours
    - Unsuccessful events tend to happen **repetitively** on a subset of tasks
    - Similar frequencies of events in adjacent hours
  - Unsuccessful executions could be described by **Moving Average** models

Root causes of evictions





- **Machine saturation level** vs. `eviction`
  - Identification of **concurrent** tasks (running on the same machine at `eviction` time)
  - Computation of two different saturation levels:
    - **Reservation** level: total amount of requested resources / machine equipped resources
    - **Utilization** level: total amount of used resources / machine equipped resources
  - Peak of `eviction` events when machines are **near saturation**
- **Task priority** vs. `eviction`
  - Identification of kick-in and kicked-out tasks:
    - **Kick-in**: tasks whose scheduling caused the `eviction` process
    - **Kicked-out**: tasks descheduled to free resources for the kick-in ones
  - **Key role** of priority in the `eviction` process

Acknowledgments

This work has been supported by the Swiss National Science Foundation (project 200021 141002).

Università della Svizzera italiana | Faculty of Informatics

References

[1] L. Barroso, J. Dean, and U. Hölzle. Web Search for a Planet: The Google Cluster Architecture. *IEEE Micro*, 23(2):22–28, Mar. 2003.

[2] J. Wilkes. More Google cluster data. Google research blog. `https://code.google.com/p/googleclusterdata/wiki/ClusterData2011_1`, Nov 2011.

[3] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In *ACM SoCC*, pages 7:1–7:13, 2012.