Predicting and Mitigating Job Failures in Big Data Clusters

Andrea Rosà[*], **Lydia Y. Chen**[†], **Walter Binder**[*]

[*]Università della Svizzera italiana, Faculty of Informatics, Switzerland
[†]IBM Research Zurich Lab, Cloud Server Technologies Group, Switzerland

IEEE/ACM CCGrid 2015

May 6, 2015

This work has been supported by the Swiss National Science Foundation (project 200021_141002) and EU commission under FP7 GENiC project (608826).

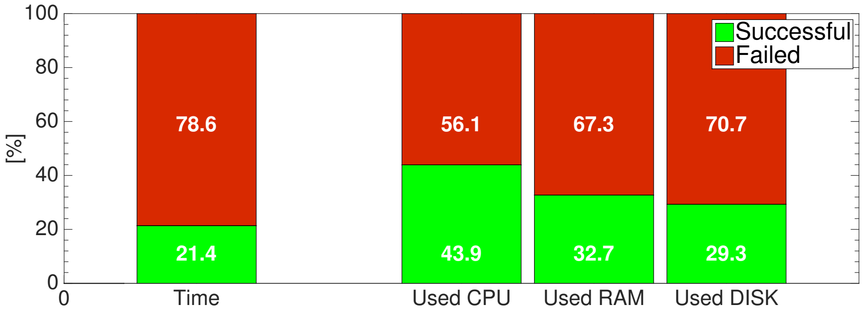
Big-Data Systems

- **Big-data** is becoming a key requirement for many applications
  - Used for large-scale simulations, scientific computations, web indexing, sensor networks, ...
- Workload **greatly diversified**
  - High degree of **heterogeneity** and **dynamicity**
- Systems have large **scale** and are very **complex**

Problem Statement

- A lot of job **failures**!
- Potentially turn into critical performance impediments
  - Resource waste
  - Job slowdown
- It is essential to **predict** job outcomes and **mitigate** resource waste due to job failures.

Motivations



Field data: Google cluster trace [1]

- A lot of **wasted resources**
- Used for a lot of time
- May block the execution of other jobs

[1] J. Wilkes, More Google cluster data, Google research blog. Nov 2011.

Challenges

- Intricate **dependencies** among jobs and on the underlying hardware
- Jobs composed of **a lot** of tasks with different **requirements**
- Jobs exhibit strong **time-variability**

Contibutions

- Development of **on-line prediction model** for **job outcomes**
  - Using machine learning techniques
  - Employing on-line training based on historical data
  - Based on information about past jobs and system load
  - Prediction upon job arrival
- Proposal of **delay-based mitigation policy**
  - Terminates failed jobs after a grace period
  - Idea: misclassified jobs still have chance to complete successfully
- Goal: minimize **resource waste** and **harmful terminations**

Data Description

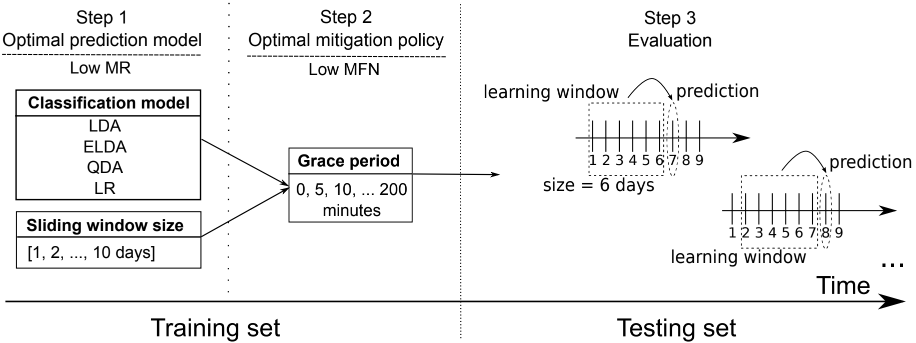
- Google cluster trace [1]
  - 29 days of workload
  - **Jobs** contain multiple **tasks**
  - Final types: `finish`, `eviction`, `fail`, `kill`
  - Two **classes** considered: `successful`, `failed`
  - **Task** attributes:
    - Specify by users at arrival time
    - Requested resources (CPU, RAM, DISK)
    - Priority $\in [0, 11]$
  - **Job** attributes: AVG/STD of task attributes

[1] J. Wilkes, More Google cluster data, Google research blog. Nov 2011.

Metrics of interests

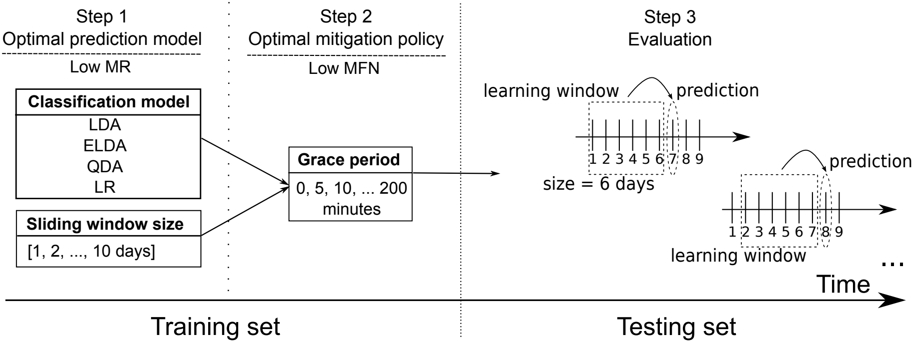
- Prediction model
  - **False negative rate**: $FN = \frac{\#\ \texttt{successful}\ \text{jobs classified as}\ \texttt{failed}}{\#\ \text{jobs}}$
  - **Misclassification rate**: $MR = \frac{\#\text{misclassified jobs}}{\#\ \text{jobs}}$
- Mitigation policy
  - **Mitigated false negative rate**:
    $MFN = \frac{\#\ \texttt{successful}\ \text{jobs terminated by policy}}{\#\ \text{jobs}}$
  - **Reduction of resource waste**:
    $RRW = 1 - \frac{\text{resources consumed applying policy}}{\text{resources consumed not applying policy}}$
    - Job resource consumption $=$
      $\#$ tasks $\cdot$ AVG task requested resources $\cdot$ job execution time

Methodology



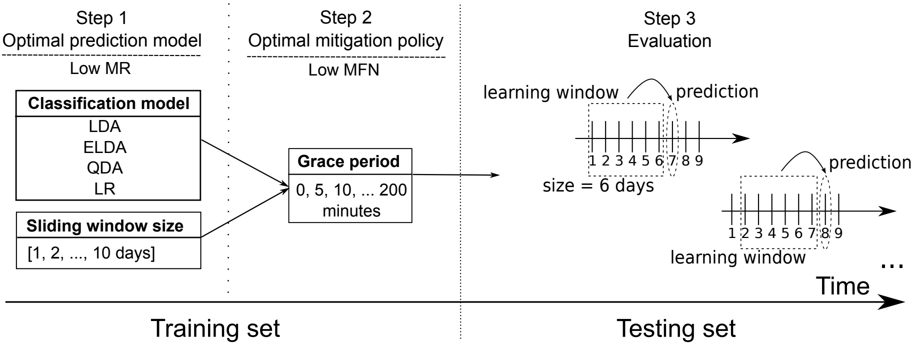
- Prediction model
  - Classification model
    - Known from machine learning theory
  - New model built every day
    - Uses attributes of past jobs in a sliding learning window

Methodology



- Mitigation policy
  - Only on predicted to fail jobs
  - Grace period length

Methodology



- Several prediction models and mitigation policies
- Derive the optimal ones
  - Optimal prediction model: low MR
  - Optimal mitigation policy: low MFN
- Training set vs testing set

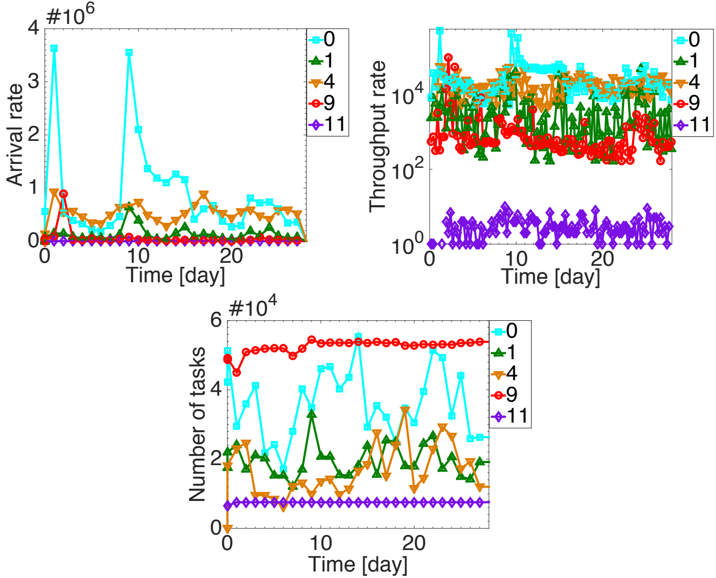
Feature Sets

- Two feature sets assigned to each job:
  - **Static** features: static information about jobs
  - **System** features: related to system load at job arrival time
- Static features:
  - Job requested CPU, RAM, DISK
  - Job priority
  - Number of tasks
  - Total: 9 static features

System Features (1)

- Idea: job outcome also depends on system load
- System load indicators:
  - (Sampling window = 5 minutes)
  - Arrival rate
  - Throughput rate
  - Number of tasks
- Assigned to each job at arrival time

System Features (2)



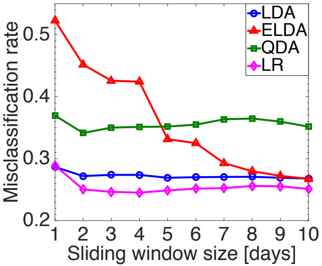
System Features (3)

- Priority matters!
  - Consider 3 different values for each system load indicator:
    - Same priority
    - Lower priorities
    - Higher priorities
- Consider instantaneous fluctuations of system state
  - Difference between most two recent sampling windows
- Total: 36 system features

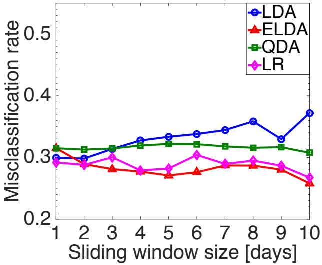
Classification Models

- Four classification models:
  - **Linear Discriminant Analysis (LDA)**
  - **Linear Discriminant Analysis on expanded basis (ELDA)**
    - Expanded feature sets: original, product, squared value
    - Total: 54 static features, 702 system features
  - **Quadratic Discriminant Analysis (QDA)**
  - **Logistic Regression (LR)**

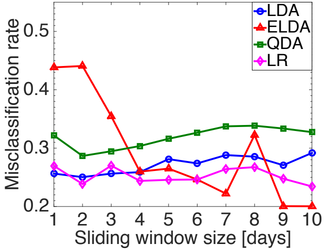
Evaluation - Prediction Model



Static features

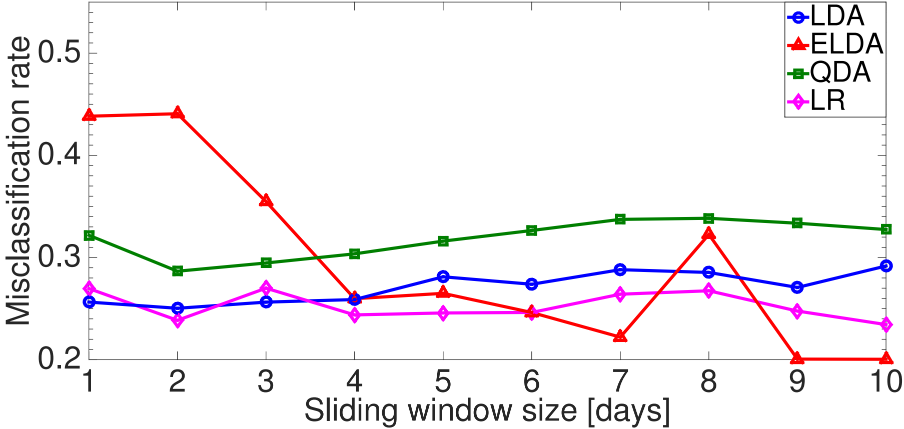


System features



Static + System

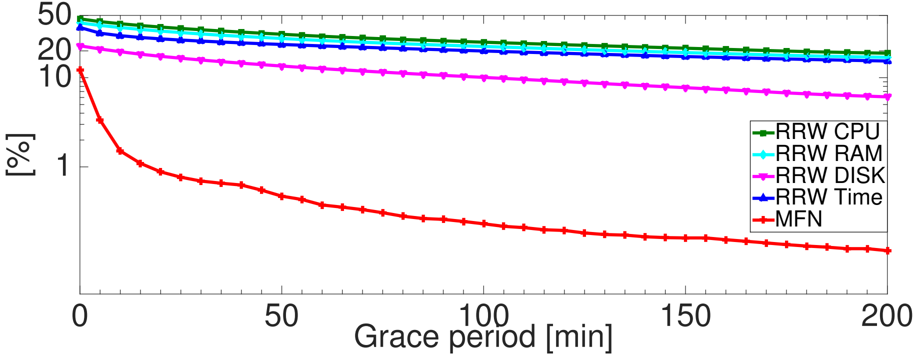
Evaluation - Prediction Model



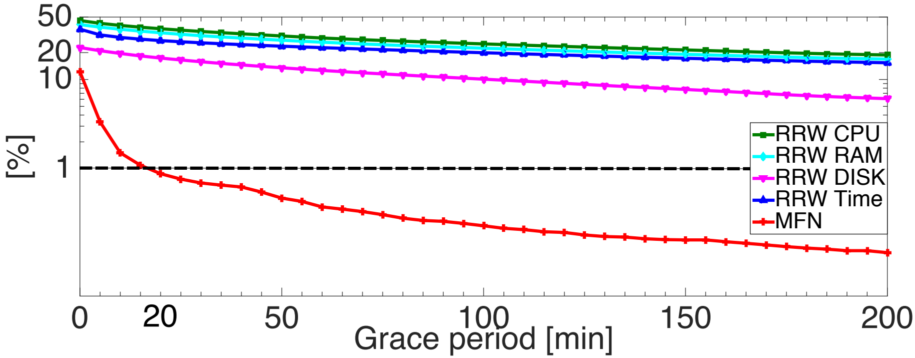
Static + System

- Optimal prediction model:
  - Uses **both** static and system features
  - Uses a long learning window (**10 days**)
  - Uses **ELDA**

Evaluation - Mitigation Policy (1)

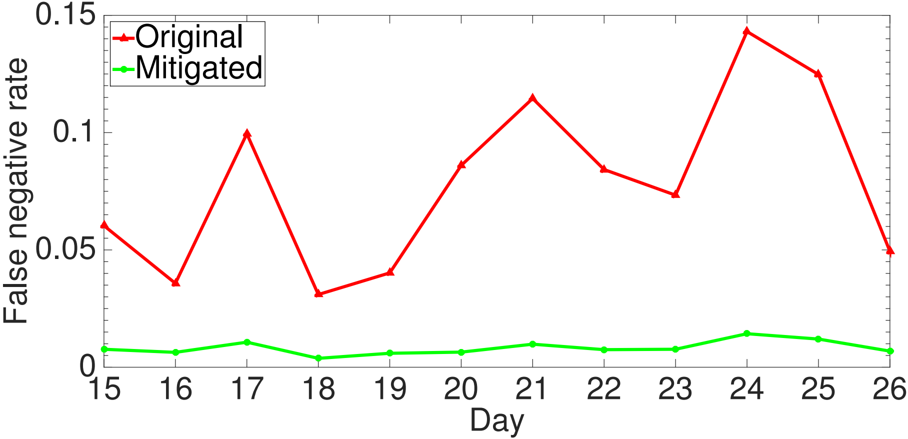


Evaluation - Mitigation Policy (2)



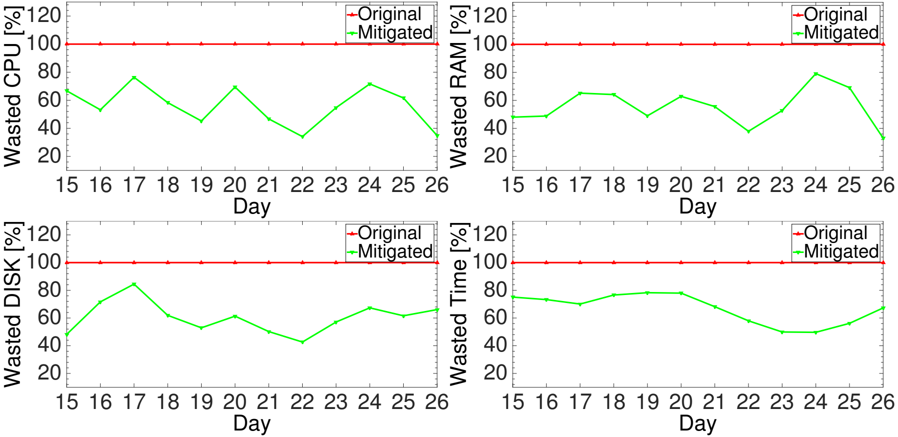
- Optimal mitigation policy:
  - Keep $MFN \leq 1\%$
  - Grace period = **20 minutes**

Evaluation - Testing Set



- AVG $MFN = 1.05\%$

Evaluation - Testing Set



- AVG RRW = 47% (CPU), 47% (RAM), 41% (DISK), 33% (time)

Conclusion

- We developed an **on-line prediction model** for **job outcomes**
  - Classification model: ELDA
  - Learning window: 10 days
- We developed a **delay-based mitigation policy**
  - Grace period: 20 minutes
- Good balance between **resource conservation** and **harmful job terminations**
- Future work:
  - Further improve classification accuracy
  - Extend prediction to tasks
  - Extend prediction **classes** (`finish`/`eviction`/`fail`/`kill`)

Conclusion

- We developed an **on-line prediction model** for **job outcomes**
  - Classification model: ELDA
  - Learning window: 10 days
- We developed a **delay-based mitigation policy**
  - Grace period: 20 minutes
- Good balance between **resource conservation** and **harmful job terminations**

Andrea Rosà

andrea.rosa@usi.ch

http://www.inf.usi.ch/phd/rosaa

Università della Svizzera italiana (USI)

Faculty of Informatics

Università della Svizzera italiana

Faculty of Informatics