Stragglers' Detection in Big Data Analytic Systems: On the Role of Heartbeat Arrival

Thomas Lambert[1], Shadi Ibrahim[2], Twinkle Jain[3], and David Guyon[4]

[1]Universite de Lorraine, Inria, LORIA, Nancy, France. [2]Inria, Univ. Rennes, CNRS, IRISA, Rennes, France.
[3]Northeastern University, Boston, USA. [4]Inria, IMT Atlantique, LS2N, Nantes, France.

Stragglers in Big Data Analytic Systems

— Big Data analytic systems is now deployed in large-scale environments to meet the ever growing size and velocity of data.
— A major challenge when running Big Data applications at large-scale infrastructures is **Stragglers**.
— Stragglers are tasks performing relatively slower than other tasks in a job.
  ▸ Can be 7-8× slower than the median task [1].
— Stragglers result in a severe degradation in performance.
  ▸ Can increase the average job duration by up to 47% [1].

Stragglers Mitigation: Handling

Big Data analytic systems (Google MapReduce [2], Apache Hadoop [3], Apache Spark [4]) employ **speculative execution** to mitigate stragglers at large scale.

— New copies of detected stragglers are launched on available machines.
— When one copy of them (original or speculative copy) completes, it is marked as successful and the other is killed.
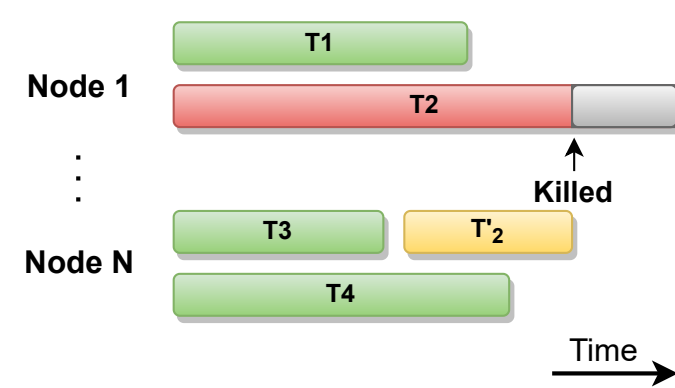

Stragglers Mitigation: Detection

— The effectiveness of speculative execution depends in part on how well they detect stragglers.
— Stragglers detection: by comparing (evaluating) tasks using the information extracted from the **last received heartbeats**.
  ▸ Heartbeats are periodic signals that report the progress/status of running tasks.
  ▸ Heartbeat window is 6s in Hadoop and 10s in Spark (by default).

The Role of Heartbeat Arrivals

— Having *up-to-date* information about the running tasks is essential when detecting stragglers.
— The **freshness of the information** reported by last received heartbeats **vary between tasks** depending on the heartbeat window, heartbeat latency, and the task starting times.
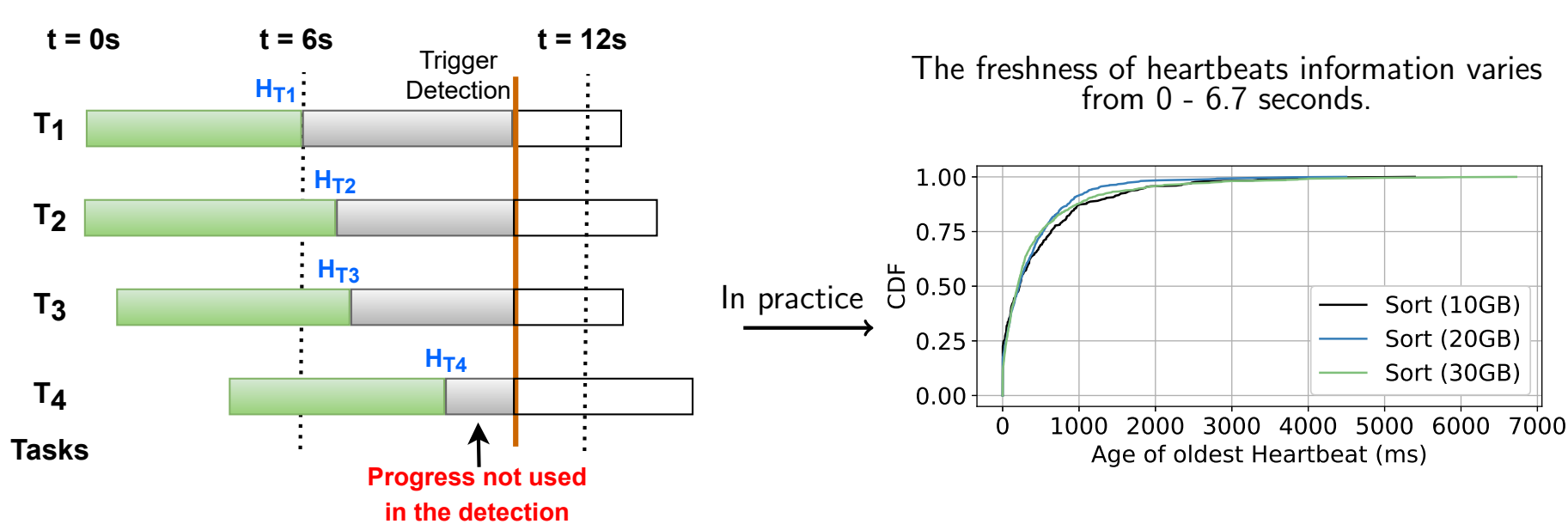
— This can mislead Big Data analytic systems to make wrong detection with high inaccuracy.

Quantifying the Impact of Heartbeat Arrival on the Inaccuracy of Stragglers Detection

— A simulator written in Java.
— Stragglers are detected every second after a task in the job is finished.
— The number of hosts is equal to the number of tasks in a job.
— Starting times of the tasks in a job:
  ▸ UNIFORMSTARTS: All starting times are equal to 0.
  ▸ SKEWEDSTARTS: Finishing times of UNIFORMSTARTS as starting times.
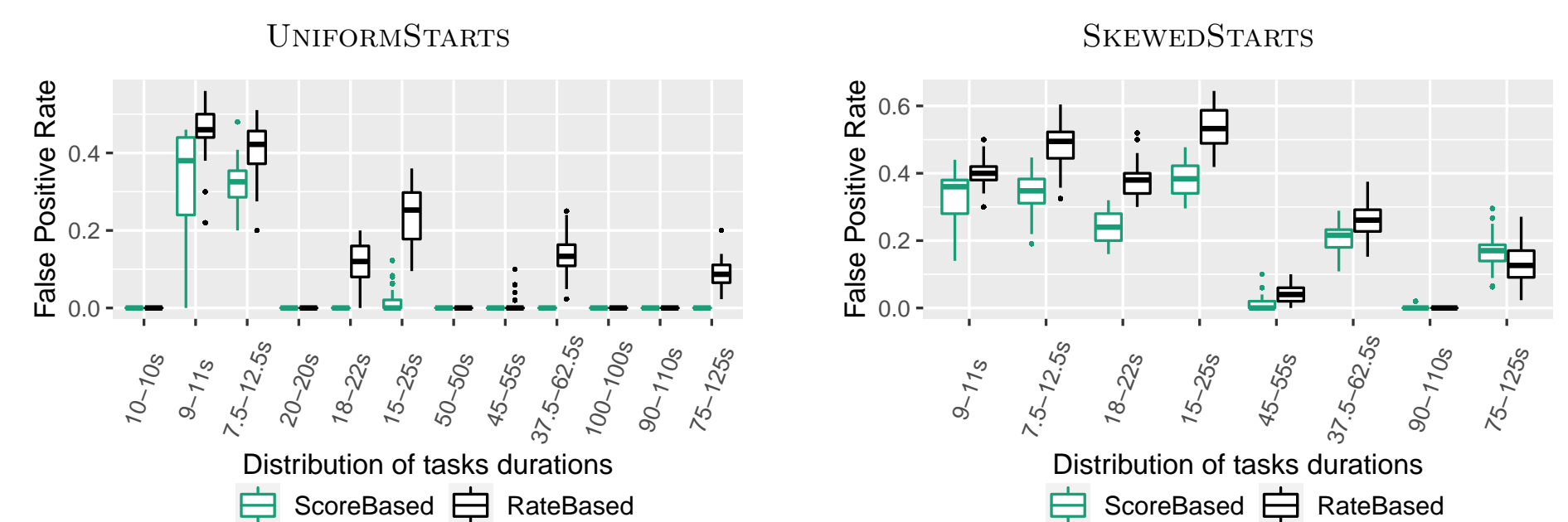
Detection Methods

ScoreBased: Score based detection.
— Uses the progress score.
  ▸ The fraction of already processed data.
— Label a task T_i as a straggler if its progress score is smaller than the average progress score of all running tasks minus a 20% (as in Hadoop).
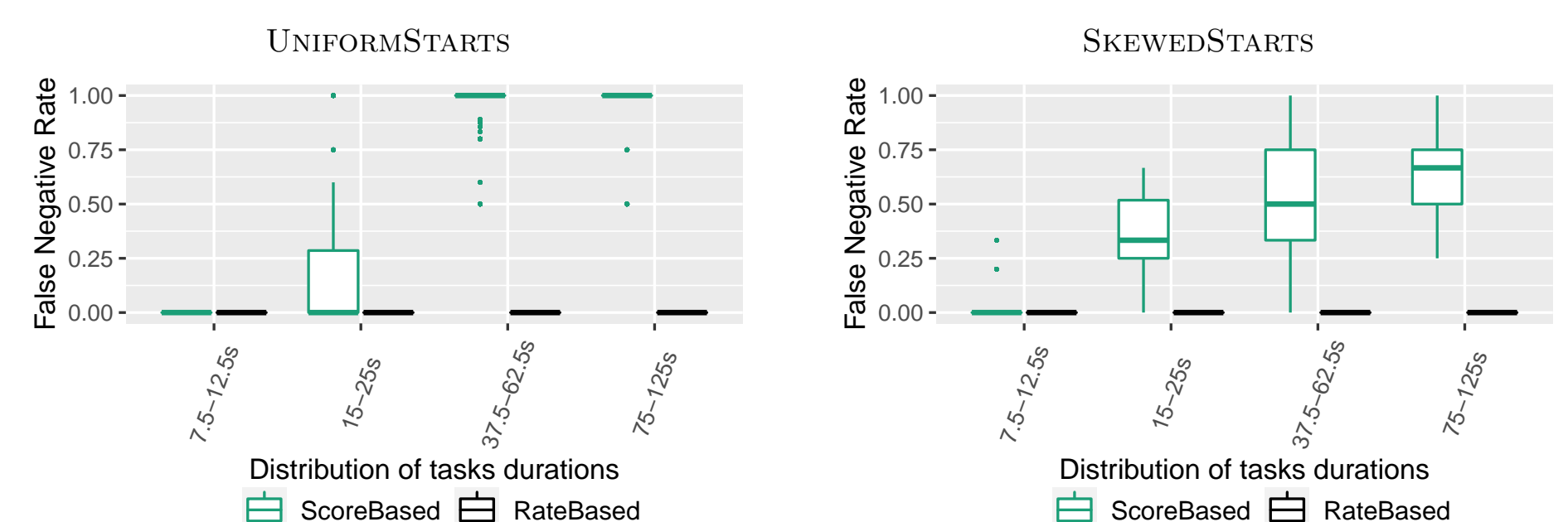
RateBased: Rate based detection.
— Uses the progress rate.
  ▸ The rate at which progress score increases.
— Mark a task T_i as a straggler if it is 20% slower than the average progress rate (or if it has 20% longer "estimated" elapsed time compared to the mean or median).

False Positive Rate

— False positives appear under both *ScoreBased* and *RateBased* detection methods.
— The false positive rate decreases when the average task duration increases.
  ▸ More obvious with *RateBased* method.
— The false positive rate increases when the starting times of tasks are skewed.

False Negative Rate

— *RateBased* method does not result in false negatives.
— *ScoreBased* method does not detect any stragglers for jobs with average task durations of 50s and 100s.
— Surprisingly, the false negative rate decreases under skewed starting times.

Conclusion

— Not considering the freshness of information during straggler detection can lead to a large amount of false positives (mainly noticeable with *RateBased*).
— This phenomenon is greatly amplified when the heartbeats are asynchronous, either because of skewed starting times or because of late heartbeats (because of latency).
— False negative is mainly an issue with *ScoreBased* methods.

Ongoing Work

— Consider the time-stamp of heartbeats in detection methods.
  ▸ Use the time-stamps to estimate the progress scores and progress rates when the detection is triggered.
— Evaluate the performance and energy efficiency of the detection methods.

References

[1] Ananthanarayanan, G. et al., "Effective Straggler Mitigation: Attack of the Clones," in *NSDI'13*, 2013.
[2] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Communications of the ACM*, vol. 51, no. 1, 2008.
[3] The Apache Hadoop Project, *http://hadoop.apache.org*, 2022.
[4] Zaharia, M. et al., "Spark: Cluster Computing with Working Sets," in *HotCloud'10*, 2010.

Acknowledgement