GRASS: Trimming Stragglers in Approximation Analytics

Ganesh Ananthanarayanan, Michael Hung, Xiaoqi Ren, Ion Stoica, Adam Wierman, Minlan Yu

Microsoft Research
Berkeley University of California
USC Viterbi School of Engineering
California Institute of Technology 1891

Next Generation of Analytics

- **Timely** results, even if **approximate**
  - Data deluge makes this necessary

Approximation Dimensions

- **Deadline**: Maximize accuracy within deadline

  "Pick the best ad to display within 2s"

- **Error**: Minimize time to get desired accuracy

  "#cars sold to the nearest thousand"

Optimal Scheduler

Improve accuracy by **48%**

Speedup by **40%**

**w.r.t.* *<u>state-of-the-art</u> schedulers* (production workloads from Facebook and Bing)

Scheduling Challenge

- **Prioritize** tasks
  - Subset of *tasks* to complete
  - #tasks » #slots (*multi-waved* jobs)

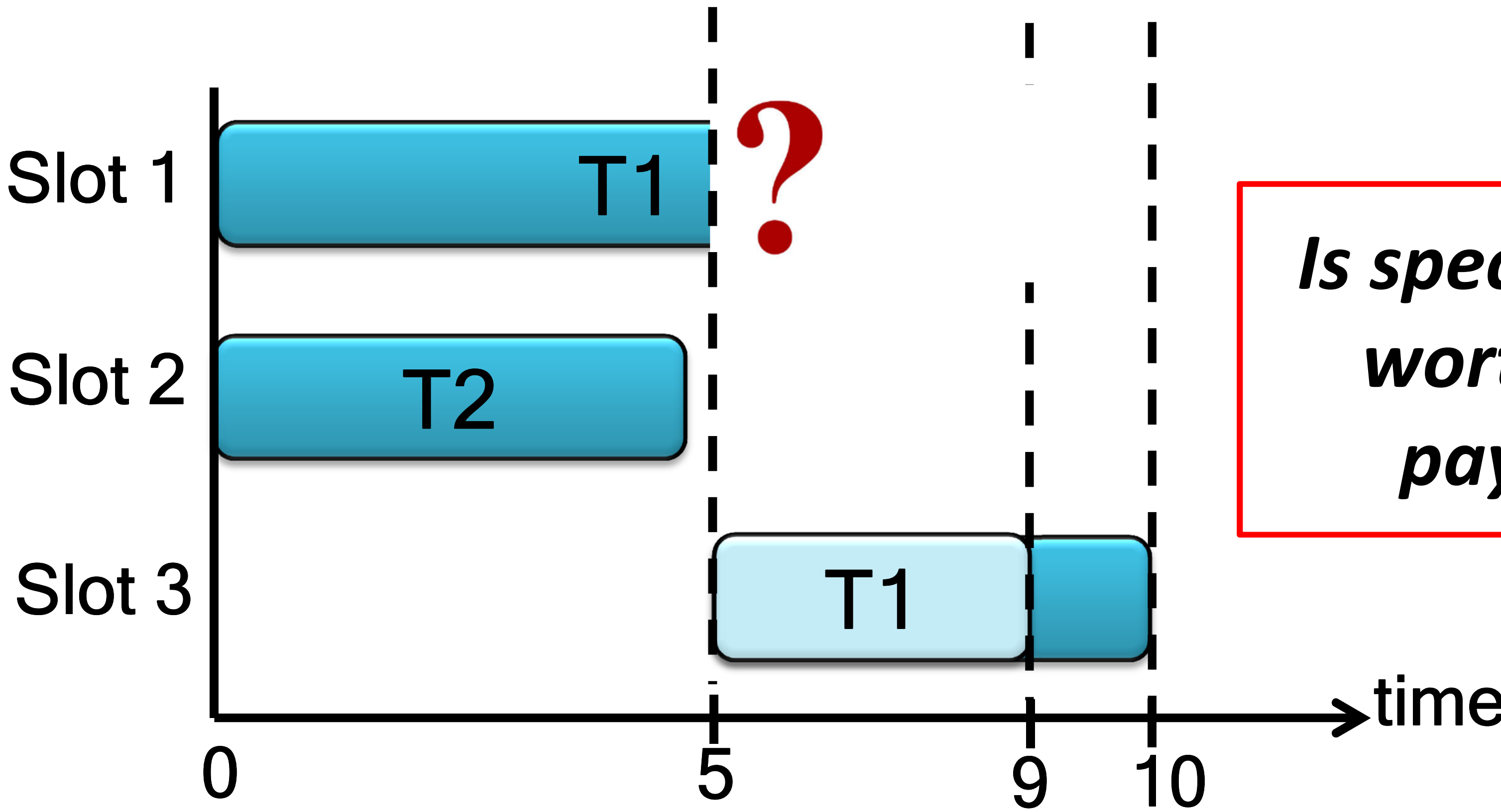
(*NP-Hard but many known heuristics…*)

- **Straggler** tasks
  - Slowest task can be **8x** slower than median task
  - **Speculation**: Spawn a duplicate, earliest wins
    - Google[OSDI'04], FB[OSDI'08], Microsoft[OSDI'10]

Challenge: *dynamically prioritize between speculative & unscheduled tasks to meet deadline/error bound*

Opportunity Cost

Speculative copies consume *extra* resources



Is speculation worth the payoff?

Roadmap

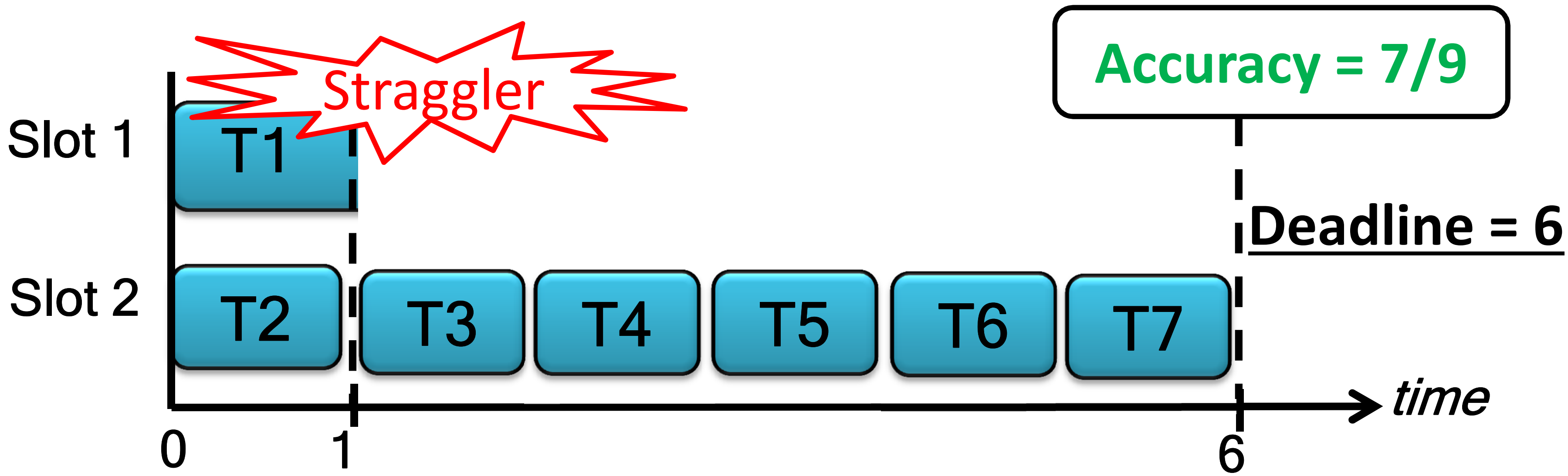
1. Two natural scheduling designs

2. **GRASS**: Combining the two designs

3. Evaluation of **GRASS**

Greedy Scheduling (GS)

Greedily improve accuracy, i.e., earliest finishing task

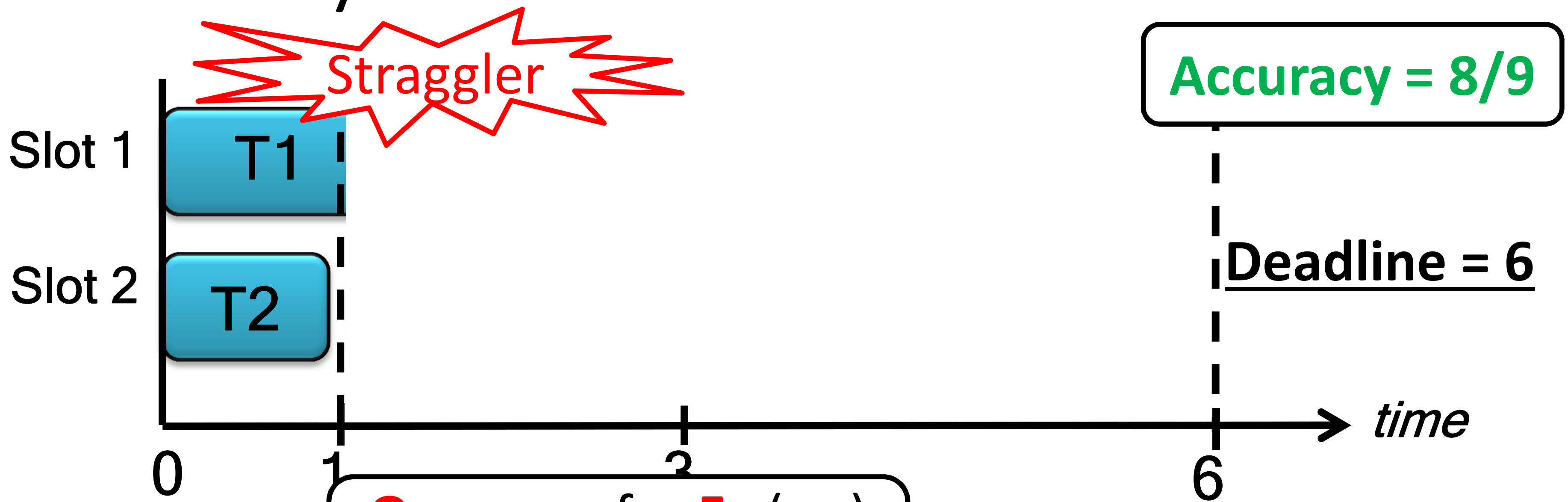


(at time =1)

Task ID	T1	T2	T3	T4	T5	T6	T7	T8	T9
Time remaining	5	---	---	---	---	---	---	---	---
New copy	2	---	1	1	1	1	1	1	3

Resource Aware Scheduling (RAS)

Speculate only if it saves time *and* resources

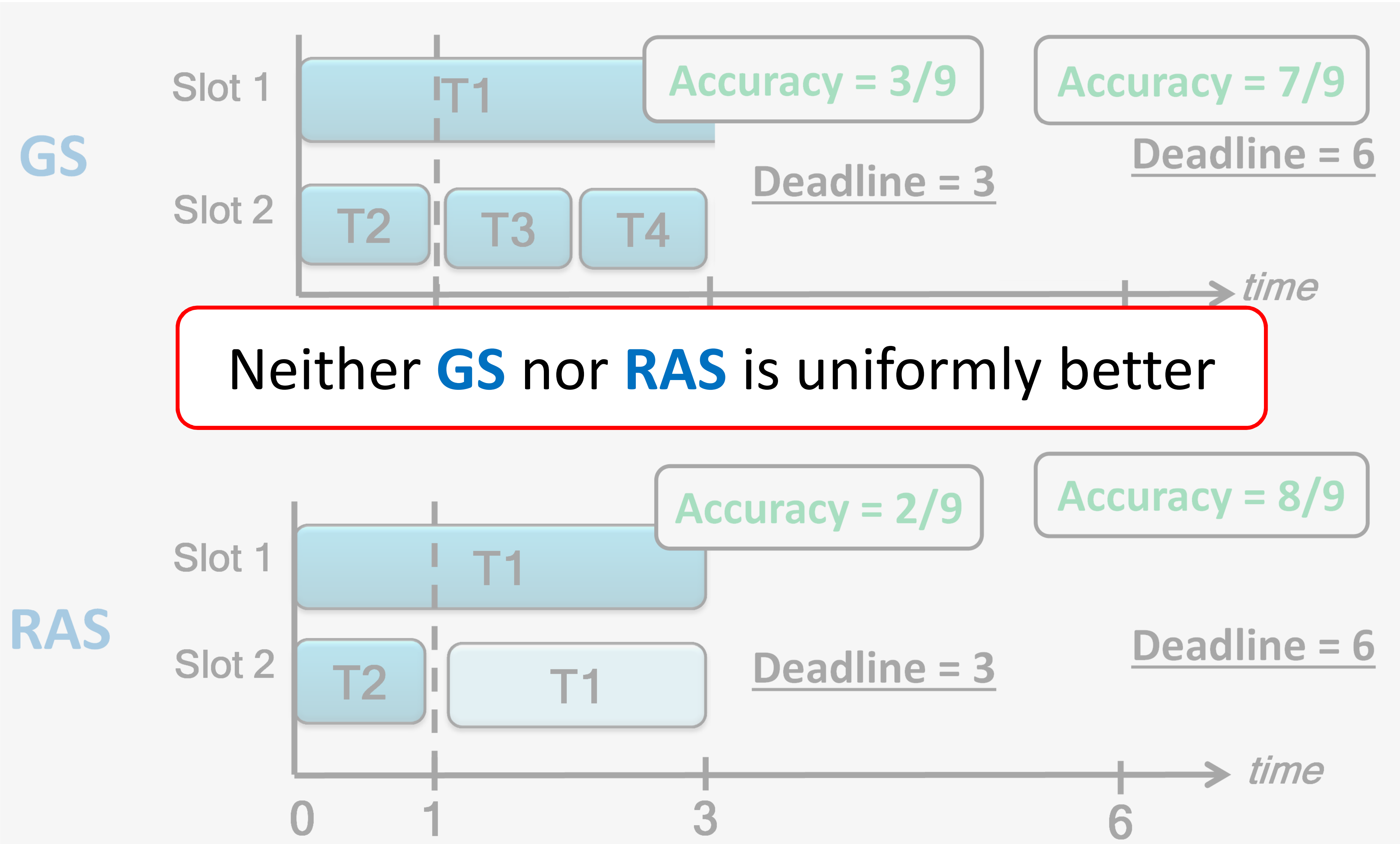


One copy for **5s** (vs.)
Two copies for **2s**

(at time =1)

Task ID	T1	T2	T3	T4	T5	T6	T7	T8	T9
Time remaining	5	---	---	---	---	---	---	---	---
New copy	2	---	1	1	1	1	1	1	3

GS vs. RAS



Neither **GS** nor **RAS** is uniformly better

Intuition:

Use **RAS** early in the job (be “conservative”), switch to **GS** towards the end (be “aggressive”)

Theoretical Scheduling Model

- Multi-waved scheduling of tasks
  - Constant wave-width
  - Agnostic to fairness policies
  - Heavy-tailed (Pareto) distribution of task durations
- Speculation: GS, RAS, Switching, Optimal

Theorem:

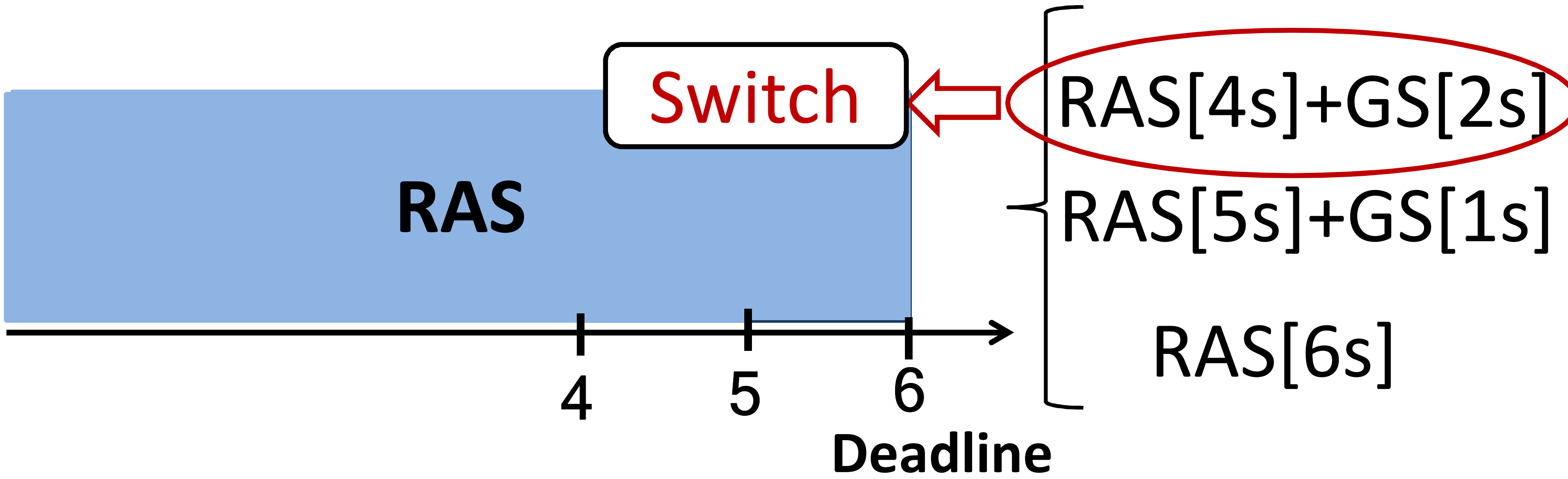
Using RAS when >2 waves of tasks remain, and GS when ≤2 waves of tasks remain is "near-optimal"

How to estimate two remaining waves?

- Wave boundaries are not strict
  - Non-uniform task durations
- Wave-width is not constant

Start with **RAS** and switch to **GS** *close* to the deadline/error-bound

Learning the switching point



- **GS**-only and **RAS**-only job samples
  - "Exploration vs. Exploitation"
  - Multi-armed bandit solution, $\varepsilon = 0.1$

GRASS (= GS + RAS) Scheduler

- **Opportunity Cost** in speculation for stragglers
  - **GS** → Greedy Scheduling
  - **RAS** → Resource Aware Scheduling
- *Switch* **RAS**→**GS** *close* to deadline/error-bound
  - Learn switching point empirically from job samples
- Provably **near-optimal** in theoretical model

Implementation

- Hadoop 0.20.2 and Spark 0.7.3
  - Modified Fair Scheduler
  - Job bins with **GS**-only and **RAS**-only samples

- Task Estimators
  - Remaining time is extrapolated from data-to-process
    - progress reports at 5% intervals
  - New copy's time is sampled from completed tasks

How well does GRASS perform?

- Workload from Facebook and Bing traces
  - Hadoop and Dryad production jobs
  - Added deadlines and error bounds

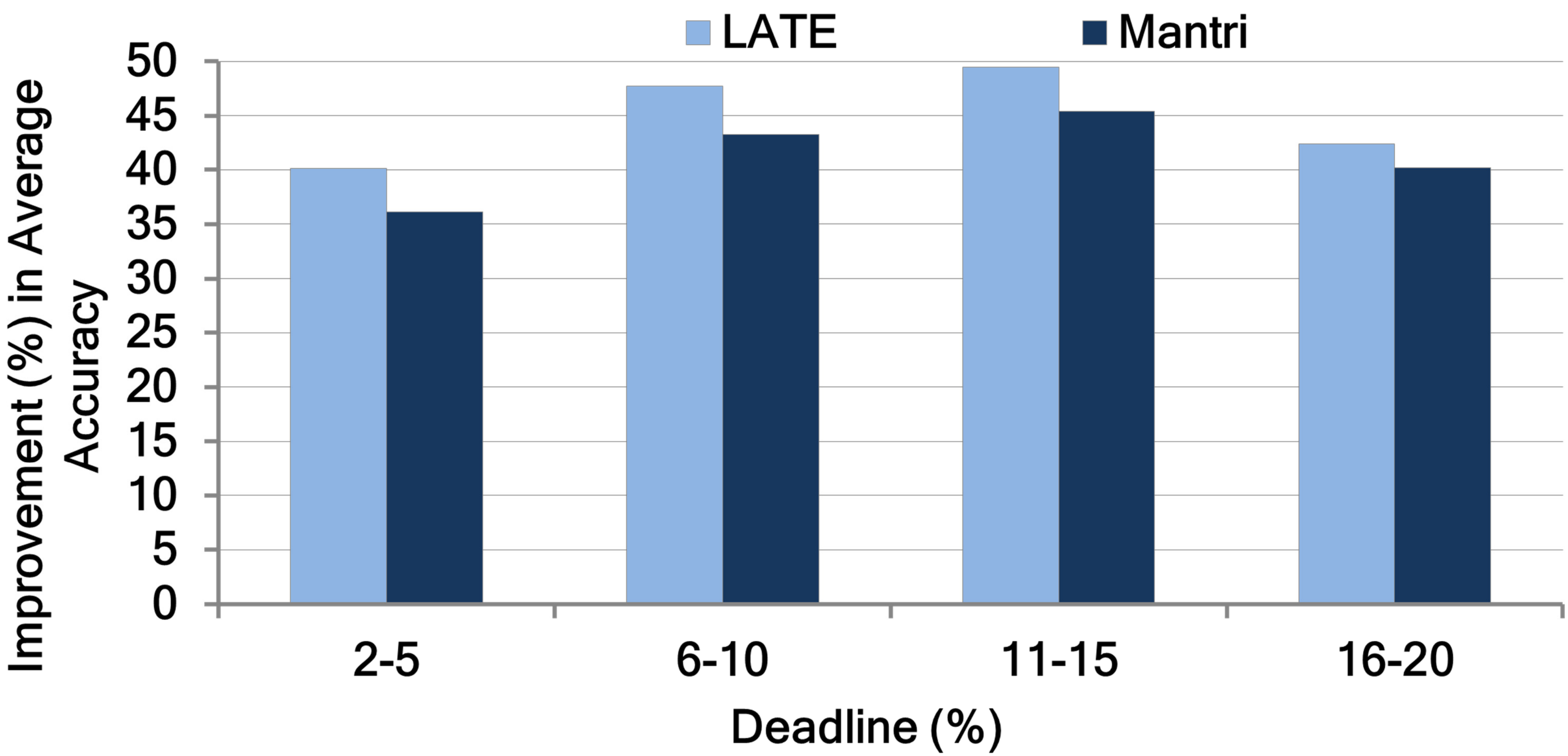
- Baselines: **LATE** & **Mantri**

  facebook bing

- 200 node EC2 deployment (m2.2xlarge instances)

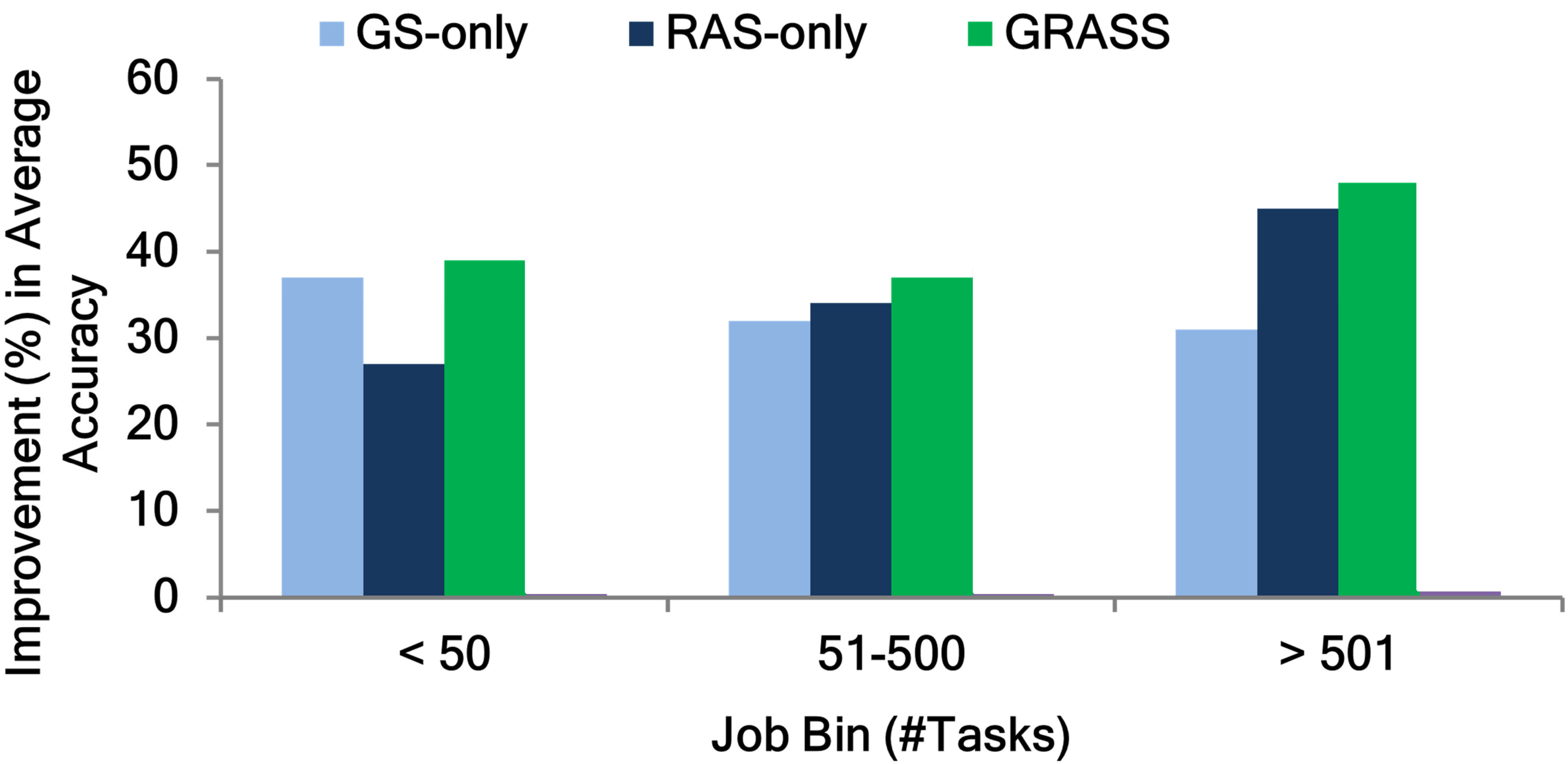
Accuracy of deadline-bound jobs improve by 47%

Gains hold across deadlines (lenient and stringent)



GRASS is 22% better than statically picking GS or RAS

… and is near-optimal



Error-bound Jobs

- Overall speedup of **38%** (optimal is 40%)
  - Gains hold across all error bounds

- Exact jobs (0% error-bound) speed up by **34%**

Unified Straggler Mitigation

Conclusion

- Next gen. of analytics: ***Approximate*** but timely results
- Challenge: Dynamic and unpredictable stragglers

- **GRASS** **–** Conservative *speculation* early in the job; aggressive towards its end

- Evaluation with Hadoop & Spark
  - Accuracy of deadline-bound jobs improve by **47%**
  - Error-bound jobs speed up by **38%**