CloudSense

Continuous Fine-Grain Cloud Monitoring with Compressive Sensing

H. T. Kung, **Chit-Kwan Lin**, Dario Vlah

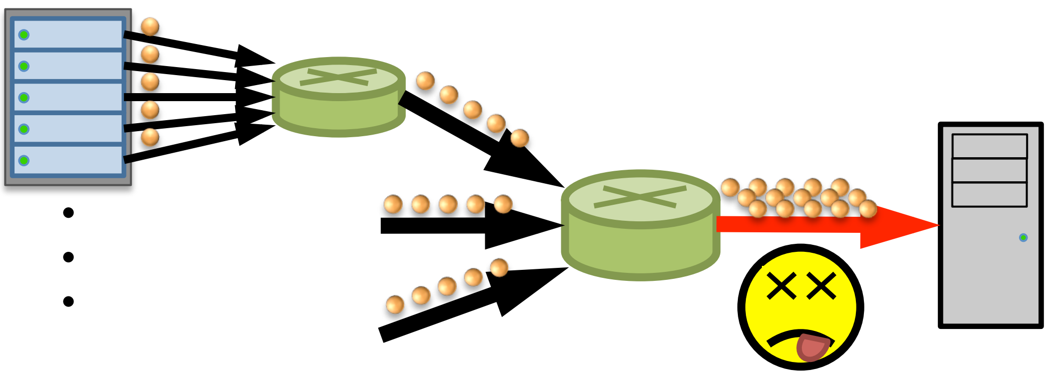
HARVARD
School of Engineering
and Applied Sciences

Monitoring and Performance

- Cloud monitoring and performance can be intimately related
- Finer-grain cloud state info ⇒ better performance
  - Schedulers: improve resource utilization
  - Apps: improve responsiveness
- Important to providers and customers alike

Challenges of Fine-Grain Monitoring

- Prohibitive network overhead
  - (125 bytes/1s) × 100 streams × 10 apps × 10K nodes = **10Gbps**!
  - Significant fraction of bisection bandwidth
- Bottleneck at collection point limits granularity



- **Global relative comparisons** require **global status collection**
  - Important class of monitoring task
  - Summarization/aggregation/filtering can't help here

Example: MapReduce Straggler Detection

- **Requires global relative comparisons**
  - Master identifies a task as straggling if its progress is some factor *d* slower than median task progress
- **Requires global status collection**
  - Every node periodically reports task progress to master

- The sooner you detect and mitigate a straggler, the earlier your job completes

A Solution?

- Monitoring is often for anomaly detection
- Status stream may be high volume, but we only care about anomalies in the stream, which are by definition sparse (i.e., compressible)
- Compress the status stream in-network, before it reaches the bottleneck

A Solution?

- Monitoring is often for anomaly detection
- Status stream may be high volume, but we only care about anomalies in the stream, which are by definition sparse (i.e., compressible)
- Compress the status stream in-network, before it reaches the bottleneck

But distributed compression is hard!

The Secret Sauce

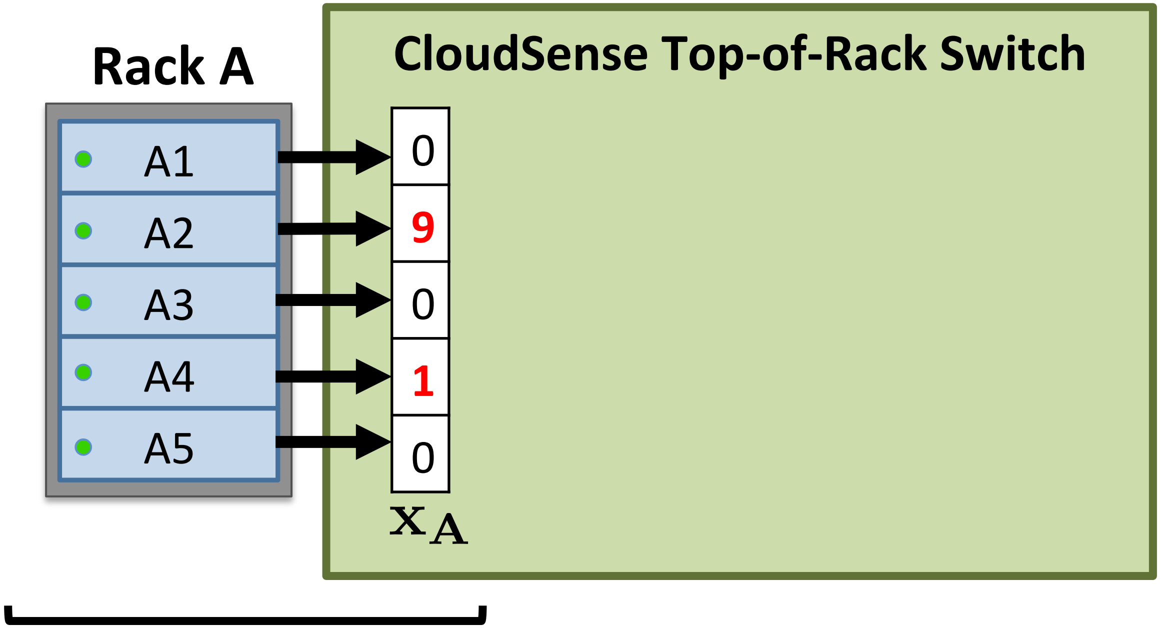
- **Compressive sensing**, a signal processing technique for distributed compression without coordination/side information

Advantages of Compressive Sensing in Cloud Monitoring

- For cloud app monitoring, compressive sensing
  - **Increases reporting granularity**, given same bandwidth budget
  - Is **simple to implement** inside a switch
  - Has a useful property that the **largest anomalies are identified first** (i.e., with just a few reports) Bonus
- *CloudSense* is a switch design that realizes these gains

HARVARD School of Engineering and Applied Sciences

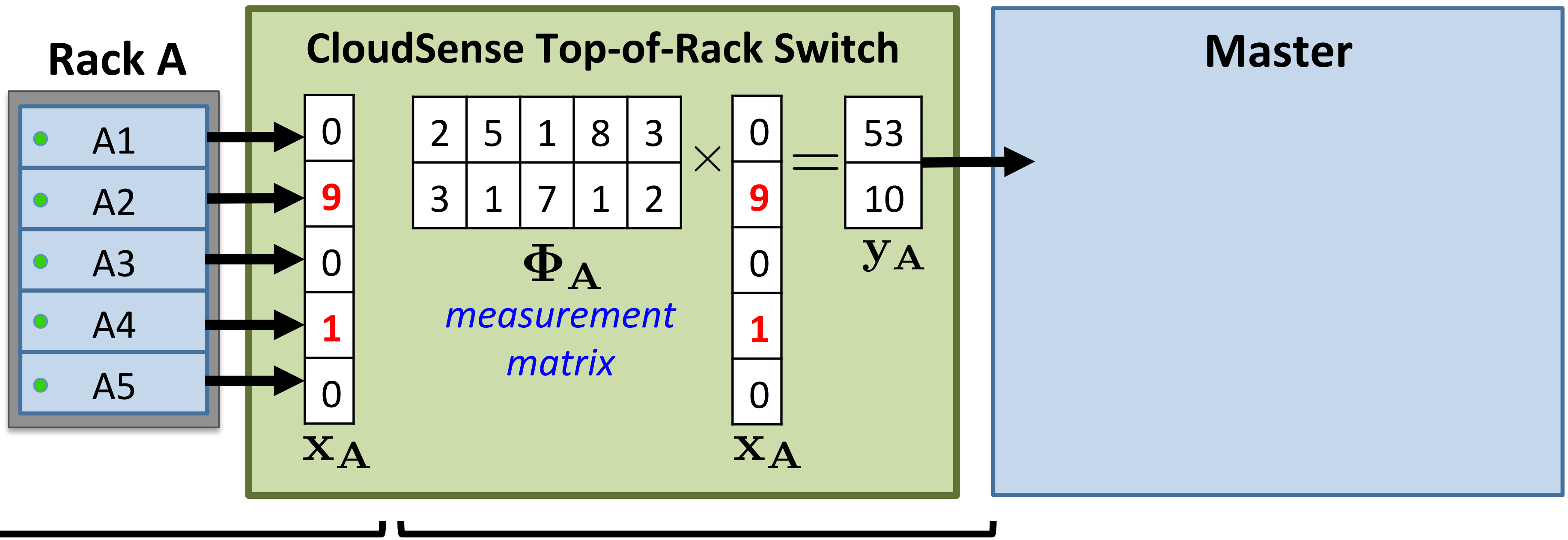
CloudSense Compressive Sensing Basics



Step 1: Collect
Switch collects status of each node ($N = 5$) into a *signal vector* x_{A}.

x_{A} is *sparse* and has sparsity $K = 2$.

CloudSense Compressive Sensing Basics



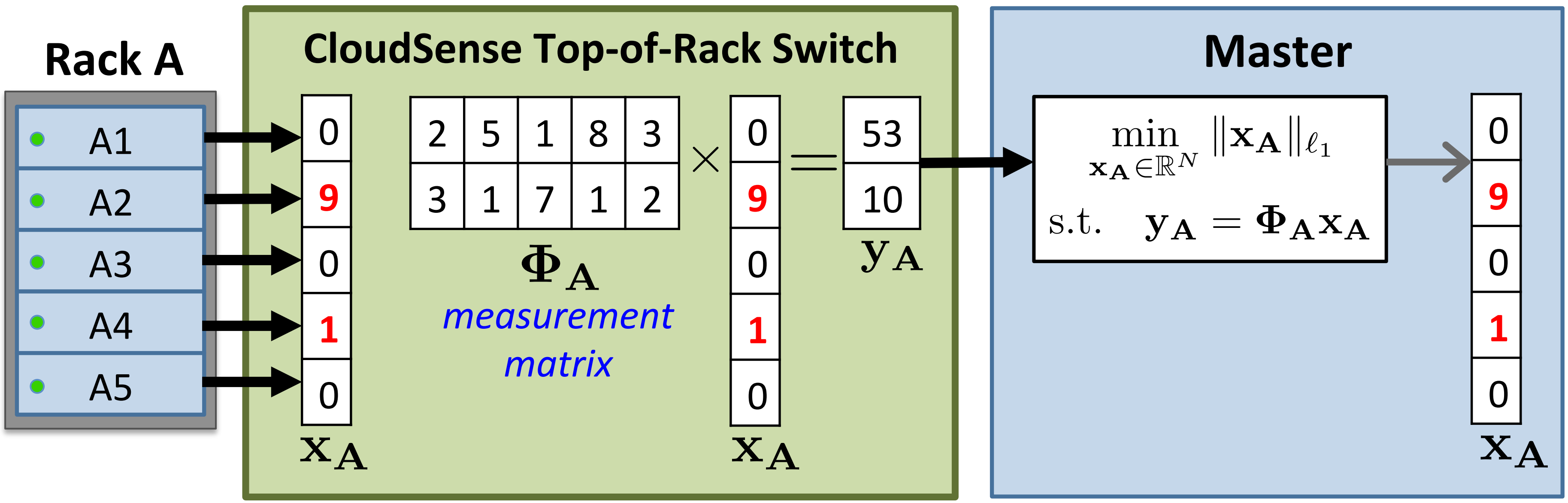
Step 1: Collect
Switch collects status of each node ($N = 5$) into a *signal vector* $\mathrm{x_A}$.

$\mathrm{x_A}$ is *sparse* and has sparsity $K = 2$.

Step 2: Encode & Send
Switch computes random projections of $\mathrm{x_A}$ onto low-D space, generating *measurement vector* $\mathrm{y_A}$, and sends it to Master.

Compression occurs because Φ_A is $M \times N$, $M << N$.

CloudSense Compressive Sensing Basics



Step 1: Collect

Switch collects status of each node ($N = 5$) into a *signal vector* $\mathrm{x_A}$.

$\mathrm{x_A}$ is *sparse* and has sparsity $K = 2$.

Step 2: Encode & Send

Switch computes random projections of $\mathrm{x_A}$ onto low-D space, generating *measurement vector* $\mathrm{y_A}$, and sends it to Master.

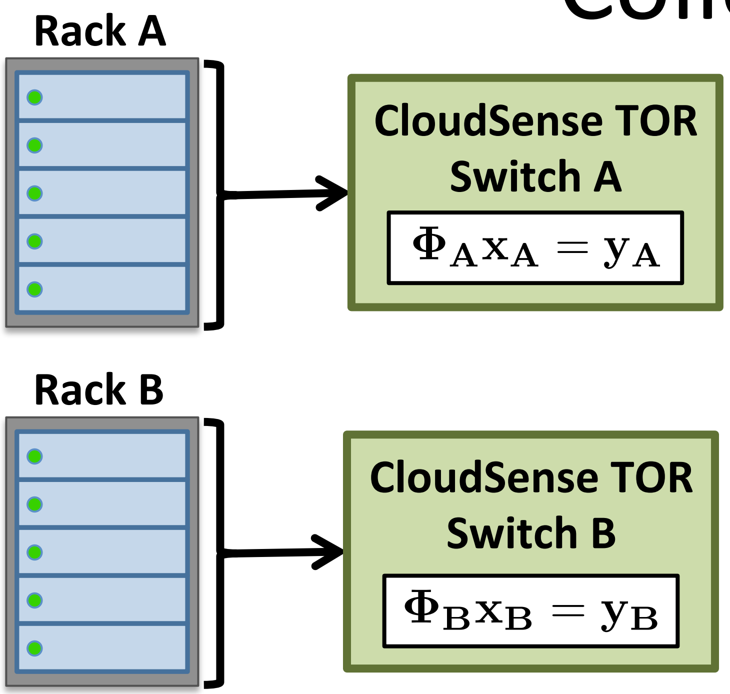
Compression occurs because Φ_A is $M \times N$, $M << N$.

Step 3: Decode

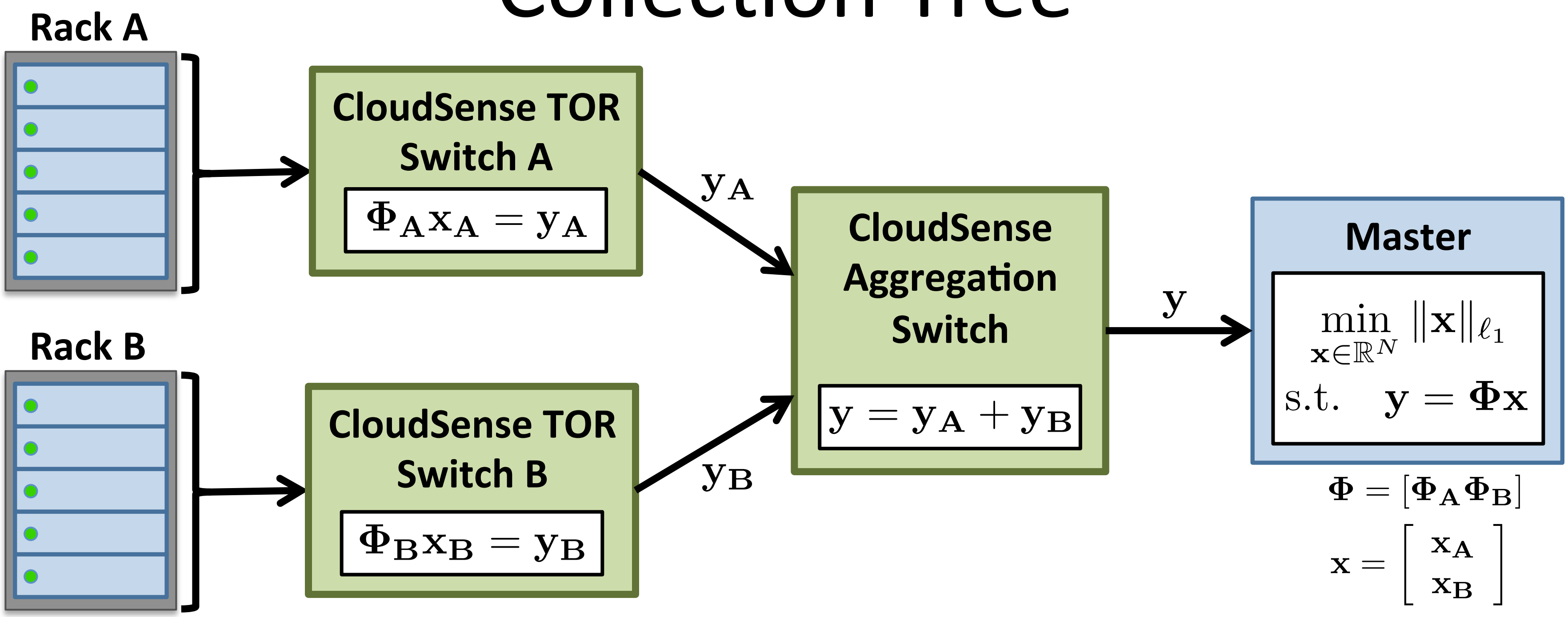
On receiving $\mathrm{y_A}$, and since it knows Φ_A, Master solves ℓ_1-minimization problem via linear programming to recover $\mathrm{x_A}$.

High probability of success if $M = O(K \log N/K)$.

A Simplified *CloudSense* Measurement Collection Tree



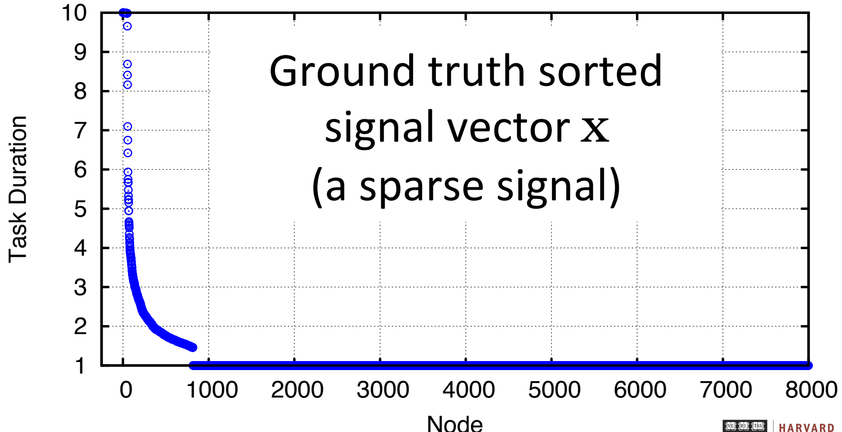
A Simplified *CloudSense* Measurement Collection Tree



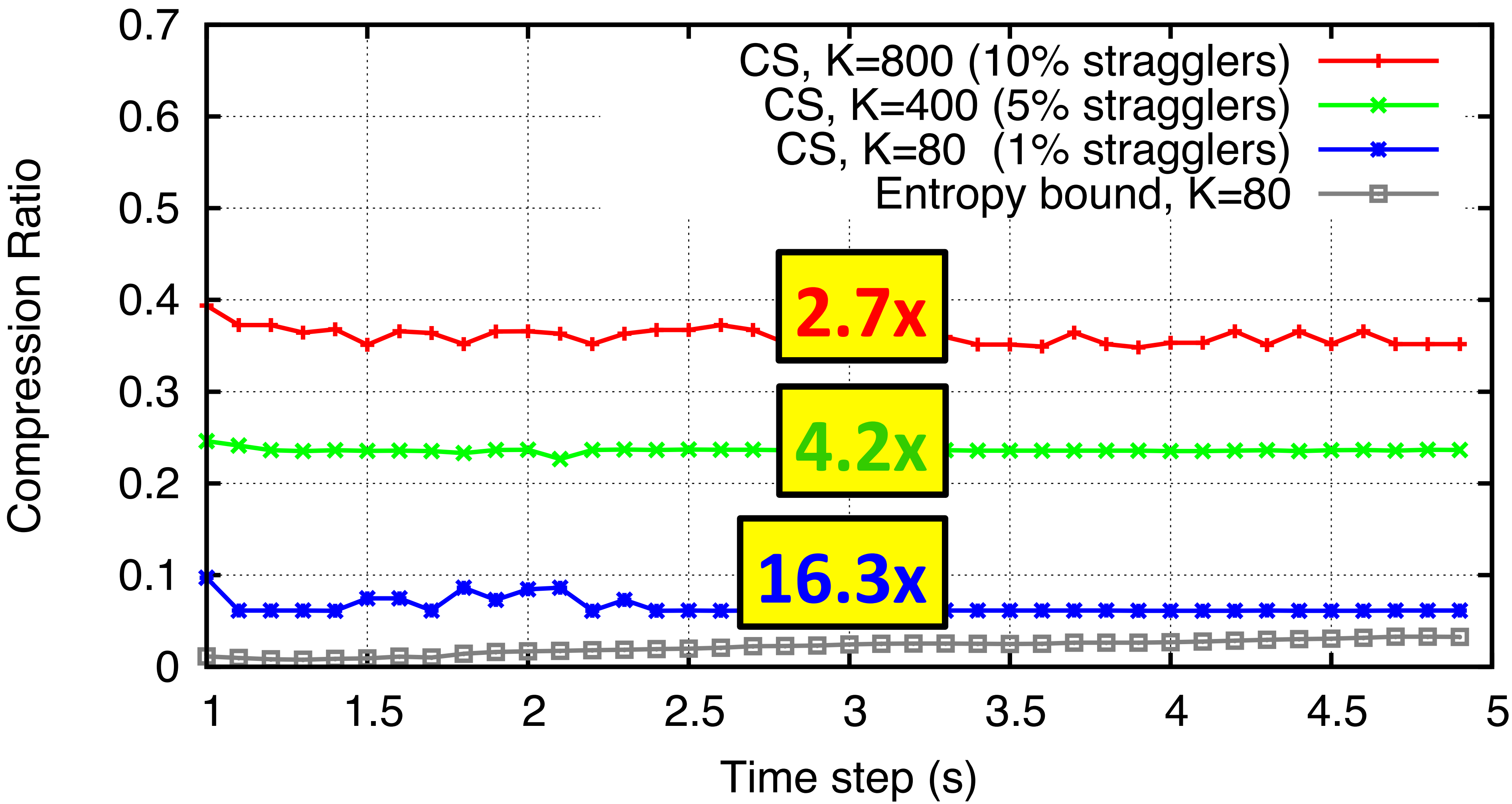
- At fan-in point (aggregation switch), measurement vectors are simply summed
- **No increase in outgoing data size** over each link

CloudSense Improves Straggler Detection

- CloudSense
  - Allows earlier detection due to finer-grain reports
  - Detects the worst stragglers first
- Simulation
  - Synthetic task progress traces derived from Bing straggler statistics (*Mantri* system, OSDI'10)
  - Each time step in the trace constitutes a signal vector $\mathbf{x}$

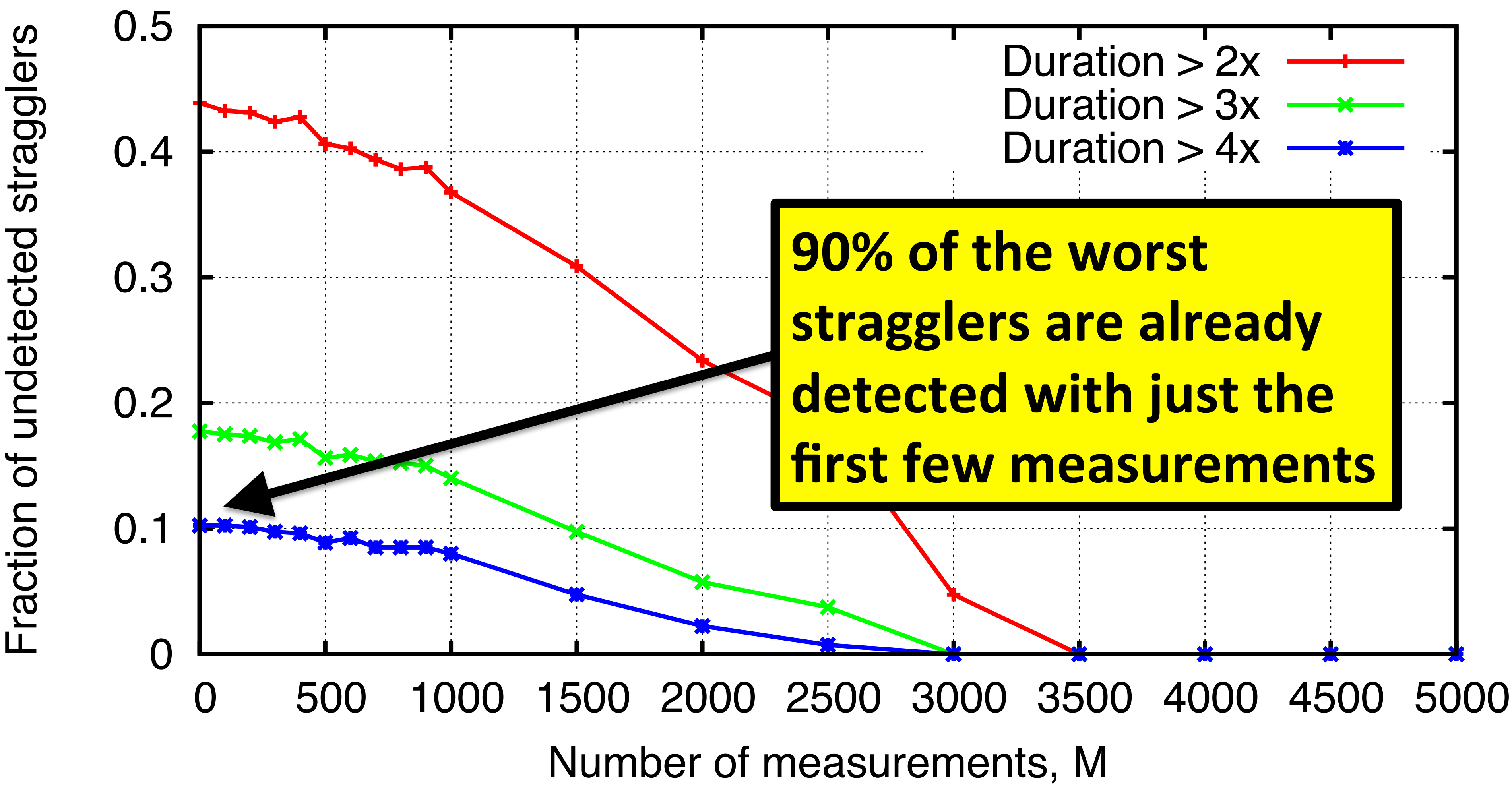


Compression Performance



HARVARD
School of Engineering and Applied Sciences

Largest-First Anomaly Detection



CloudSense Summary

- **Improves granularity** of status reports via in-network distributed compression
- Conveniently **provides largest-first** anomaly detection
- **Easy to implement** due to simple encoding
- Permits a new network service for monitoring cloud apps

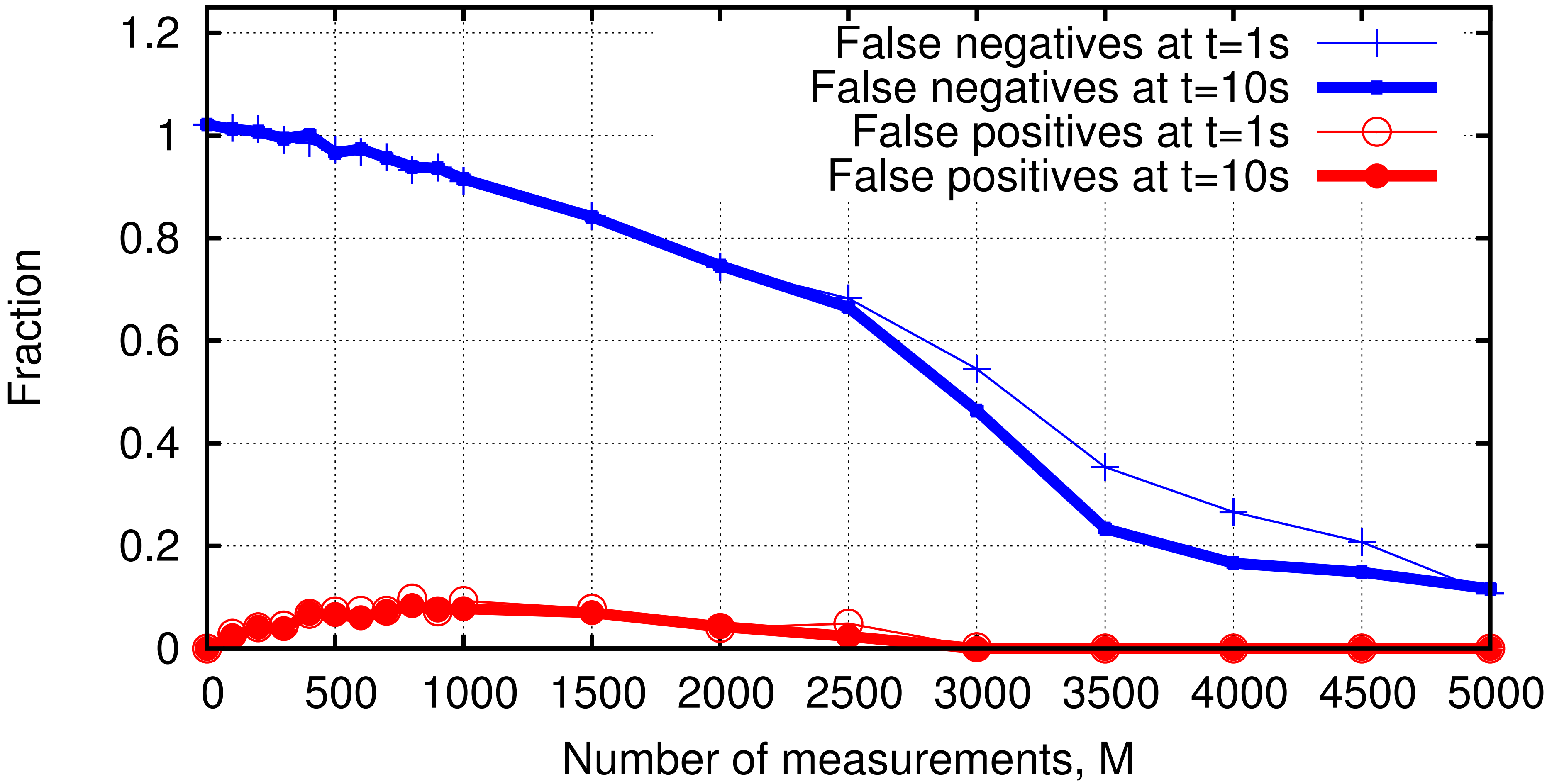
Questions & Discussion Teasers

- How sparse are various application-related status streams in production data centers?

- What are the major application opportunities that might benefit from *CloudSense*, and especially from largest-first anomaly detection?

- What are the next steps for further validation?

CloudSense Decoding Accuracy



CloudSense Decoding Accuracy