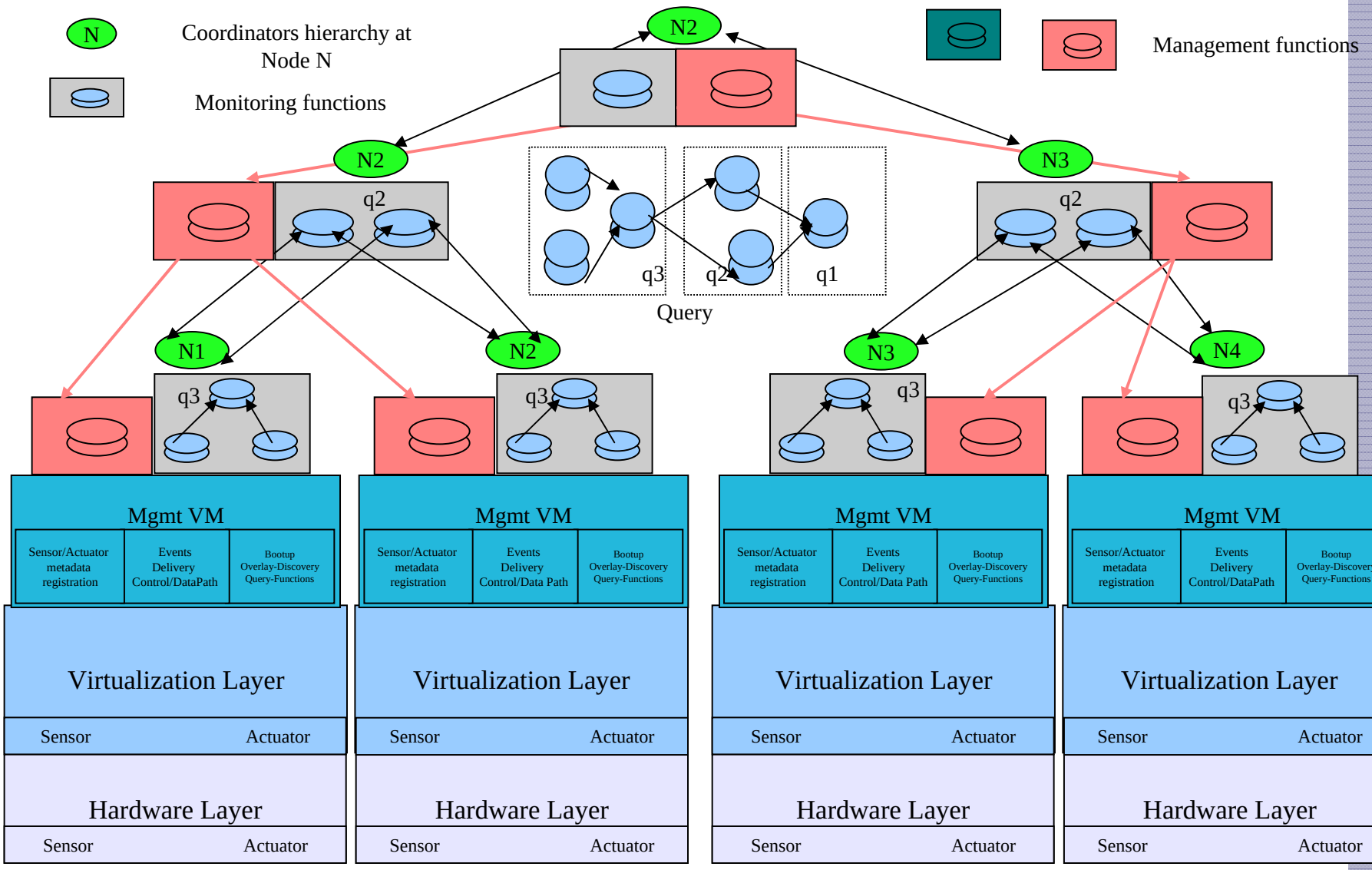
Automated Behavior Detection
(for Utility Clouds)

Karsten Schwan (schwan@cc.gatech.edu)

Vanish Talwar (HP Labs)

Partha Ranganathan (HP Labs)

Current Monalytics Architecture



Monalytics - Scalability

Dynamics I: Local analysis and filtering	Request Trace Records Size
Without Local Analysis	1.41. MB
With Local Analysis (via filtering)	60.45 KB

Dynamics II: Zoom-in Analysis	Monalytics Run(3hr)	Offline Analysis(10hr)
Centralized: Data Transferred	394.08 KB	1.15 MB
Local Analysis: Data Transferred	123.32 KB	345.6 KB

RUBiS Testcase – details below

Monalytics – Behavior Detection

Aggregation Problem

1. **Scalability**: reduce the data volume in communication and analysis
2. **Retain valuable information** for anomaly detection and identification.
3. **'horizontal crossing' and 'vertical crossing':** metrics in different levels/components are collectively considered

Detection Problem

1. **Online** designating **when** the utility cloud is experiencing anomalies.
2. High **Detection Rate** and Low **False Alarm Rate**
3. Unsupervised method with **minimal pre-knowledge** about normal or abnormal behaviors.

Zoom-In Problem

Localizing anomalies so as to **narrow** the search scopes for further diagnosing the causes of those anomalies.

Monalytics – Behavior Detection

EbAT: *Entropy based Anomaly Testing*

or

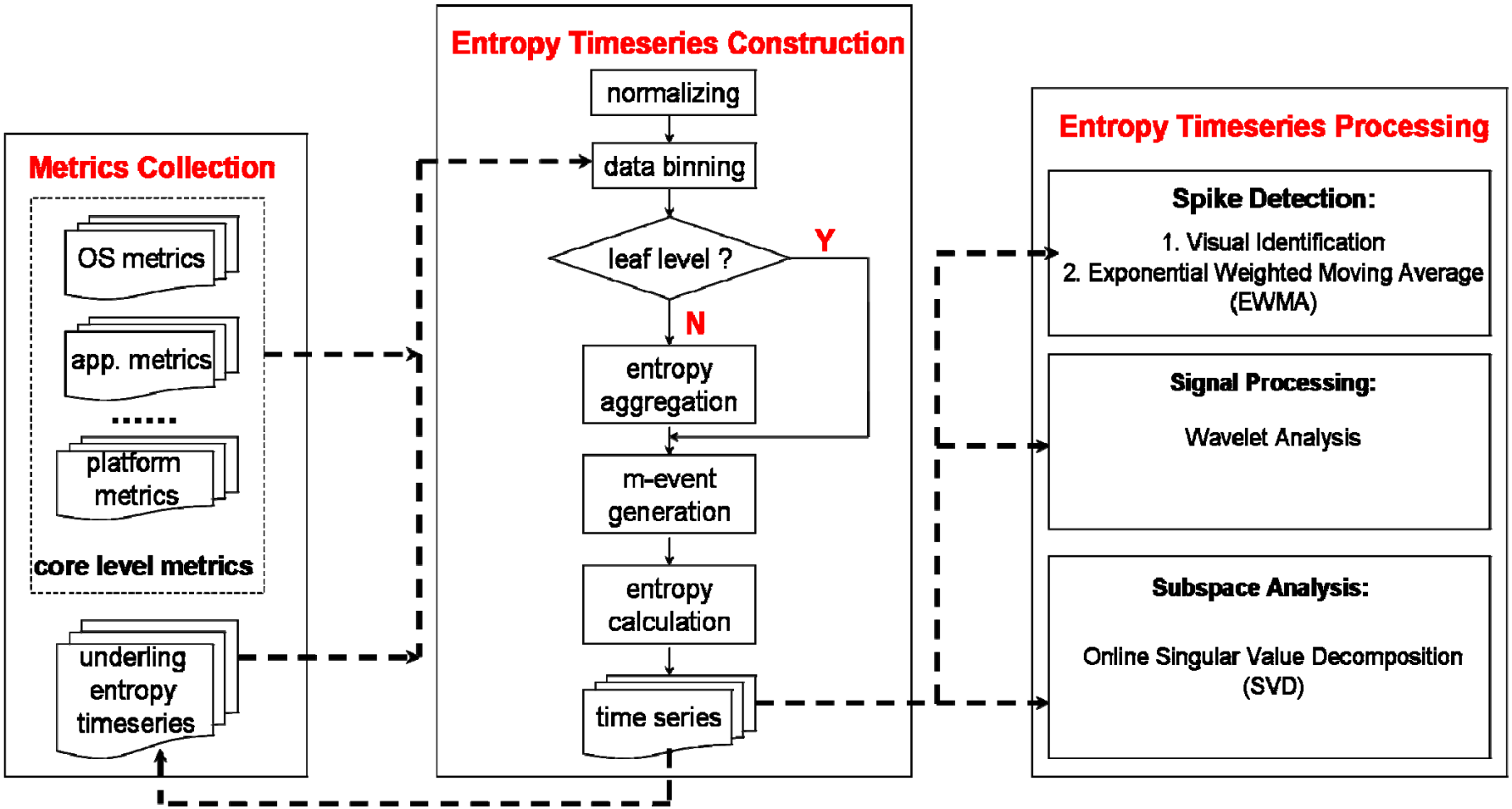
Scalable Online Anomaly Detection Using Metric Distributions

1. **1. A lightweight online approach to scalable monitoring, capable of**
2. **raising alarms and zooming in on potential problem area.**
3. **2. Detects changes in aggregated metric distributions rather than**
4. **individual metric values (like threshold-based approaches)**
5. **3. Unsupervised, minimal prior knowledge**

[1] Chengwei Wang, Karsten Schwan, Matthew Wolf, "Ebat: An entropy based online anomaly tester for data center management", poster, In BDIM 2009: 4th IEEE/IFIP International Workshop on Business-driven IT Management

[2] Chengwei Wang, Vanish Talwar, Karsten Schwan, Parthasarathy Ranganathan, "Online Detection of Utility Cloud Anomalies Using Metric Distributions", to appear in NOMS, 2010.

EbAT Overview



Aggregating metric distributions

EbAT Results – RUBiS Failure Injection

Methods	*Description*	*# Alarms*	*# Successful Detections*	*Recall*	*Precision*	*Accuracy(F_1)*	*FAR*
Entropy I	**Global Entropy Using Entropy of Child Entropies**	**45**	**43**	**0.86**	**0.96**	**0.91**	**0.04**
Entropy II	**Global Entropy Using Sum of Child Entropies**	**56**	**45**	**0.90**	**0.80**	**0.85**	**0.20**
Threshold I	**Near-Optimum**	**46**	**33**	**0.66**	**0.72**	**0.69**	**0.28**
Threshold II	**Static >0.9 or <0.05**	**18**	**16**	**0.32**	**0.89**	**0.47**	**0.11**

$$Recall = \frac{\#\ of\ successful\ detections}{\#\ of\ total\ anomalies} \qquad Precision = \frac{\#\ of\ successful\ detections}{\#\ of\ total\ alarms}$$

$$Accuracy(F_1) = \frac{2 * precision * recall}{precision + recall} \qquad False\ Alarm\ Rate\ (FAR) = \frac{\#\ of\ false\ alarms}{\#\ of\ total\ alarms}$$

On Average 57.4% Improvement on Accuracy, 59.3% Reduction on FAR

Additional Results: Hadoop 80 node OpenCirrus runs + add. metrics

Many Remaining Issues

- Scaling to Exascale:
  - dynamics: needed: scalable control, including:
    - automation in deployment and use (e.g., monalytics QoS)
  - ease of use: higher level abstractions, including
    - **linking abnormal behavior detection to problem diagnosis and prevention**
  - real-life systems: distributed utility clouds

Conclusions and Future Work

- Monitoring for effective management: Monalytics
  - rich domain for future work – software architectures, systems and platform support, methods and techniques
  - close linkage with business analytics and real-time data analysis (for us: linkage to HPC)
  - encouraging cooperation and joint research: no one system does it all, need for many methods and structures, many unsolved problems, …
- From Monalytics to Large-scale Management:
  - managing over time (simple example in vManage) to react to system and application changes over time (little understood)
- Dynamic change in management purpose and different/multiple simultaneous management goals
- Can we make systems more manageable?
  - constructing more robust (performance robustness and reliability) virtualization infrastructures – runtime learning?
  - platform support for monitoring, management, and coordination