香港中文大學
The Chinese University of Hong Kong

ARISE
Automated Reliable Intelligent Software Engineering

Intelligent Reliability Monitoring and Engineering for Online Service Systems

CHEN, Zhuangbin

Ph.D. Oral Defense

Supervisor: Prof. Michael R. Lyu

30 November, 2022

Online Services are Everywhere

Web search

Office apps

Social network

Online shopping

And many others…

Service Reliability is Crucial

Service reliability is vital for both service providers and users

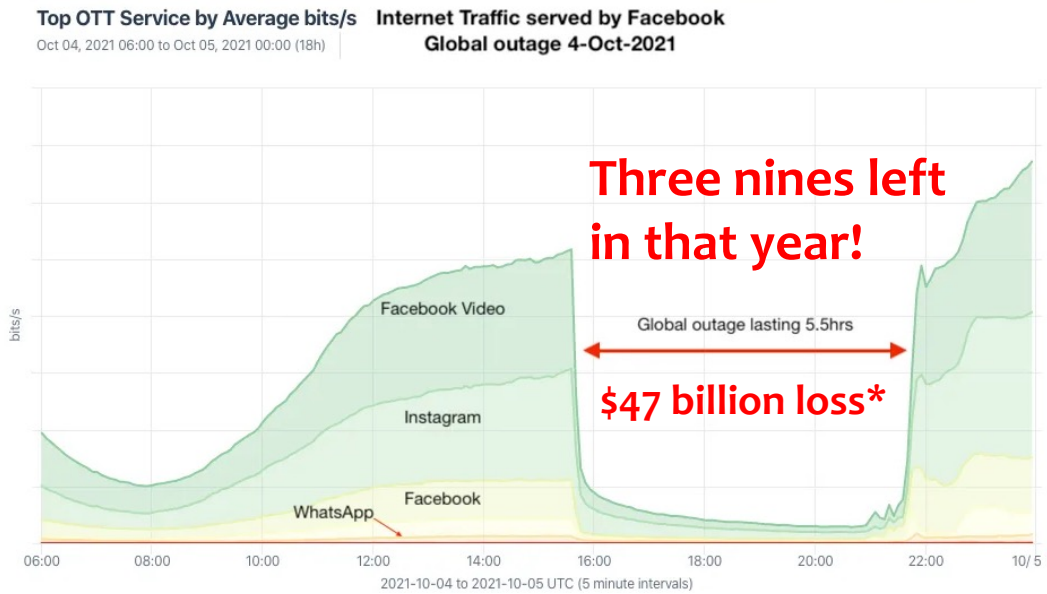
Revenue loss

Service issues

User dissatisfaction

2021 Facebook Outage

State-of-the-art service reliability: 5-6 9s (99.9999% up time)



Facebook service traffic during 2021 outage**

*Data from: https://www.datacenterdynamics.com/en/opinions/too-big-to-fail-facebooks-global-outage/
**Image from: https://en.wikipedia.org/wiki/2021_Facebook_outage

Reliability monitoring for online service systems is **crucial**, but **challenging**

Service Reliability is Challenging

Challenge 1: Large scale and complexity

MOTHERBOARD
TECH BY VICE

21 Terabytes of Open Source Code Is Now Stored in an Arctic Vault

Other artifacts stored in the archive include manuscripts from the Vatican Library and masterpieces from the National Museum of Norway.

KT By Kevin Truong

July 17, 2020, 10:08pm Share Tweet Snap

Image from: https://www.vice.com/en/article/m7jpab/21-terabytes-of-open-source-code-is-now-stored-in-an-arctic-vault

Service Reliability is Challenging

Challenge 2: Fast development iteration

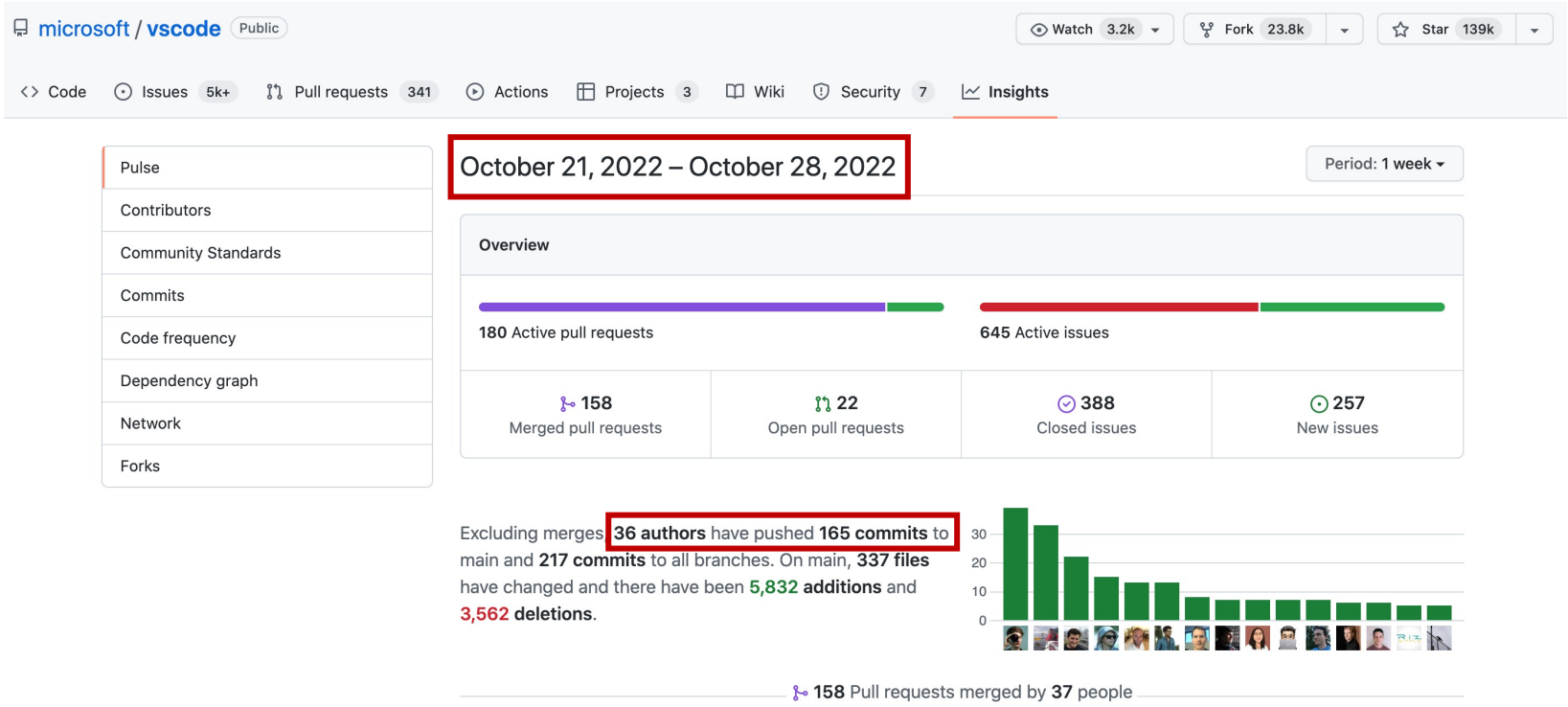
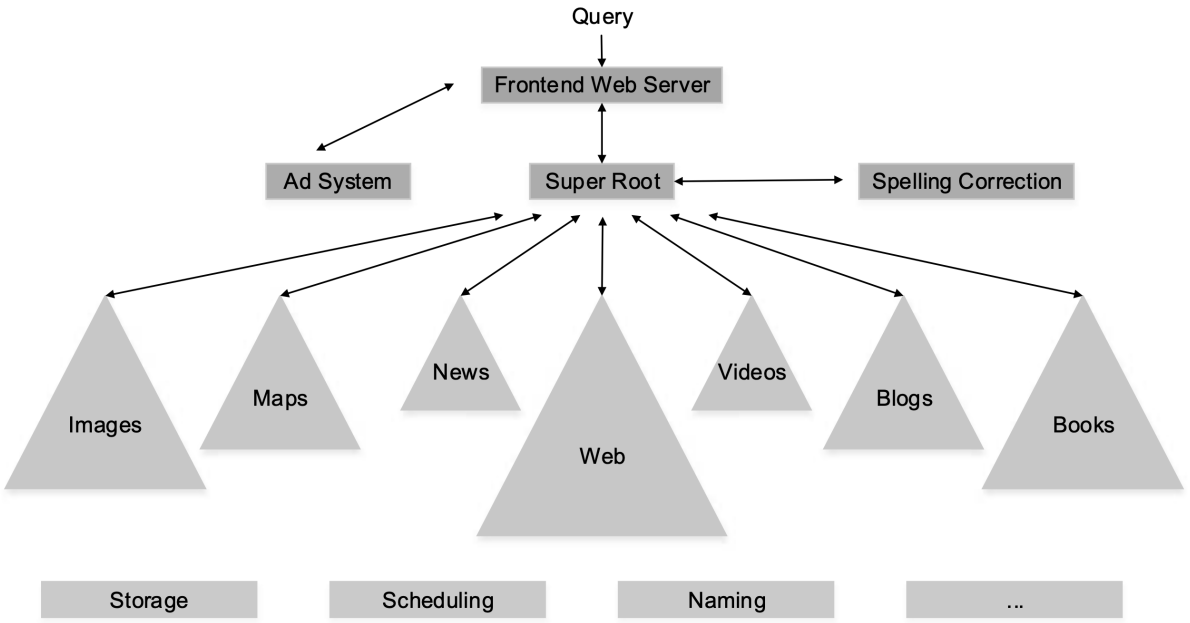


Image from: https://github.com/microsoft/vscode/pulse

Service Reliability is Challenging

Challenge 3: Complicated service dependencies



A prototype of Google search service

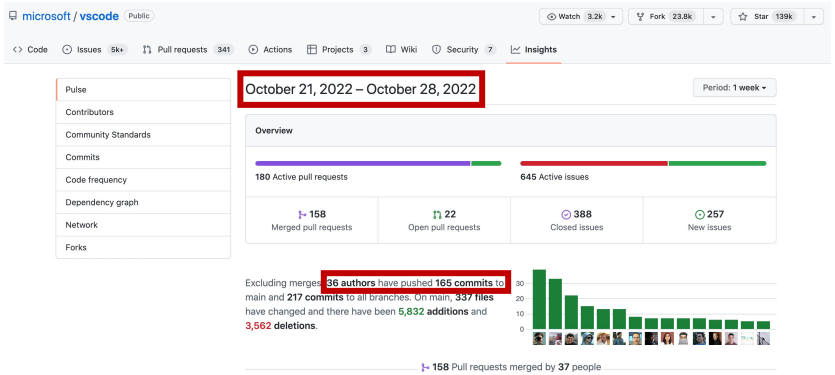
Image from: Zhu, Jieming. Data-Driven Quality Management of Online Service Systems. 2016.

Traditional engineering techniques are often **insufficient**

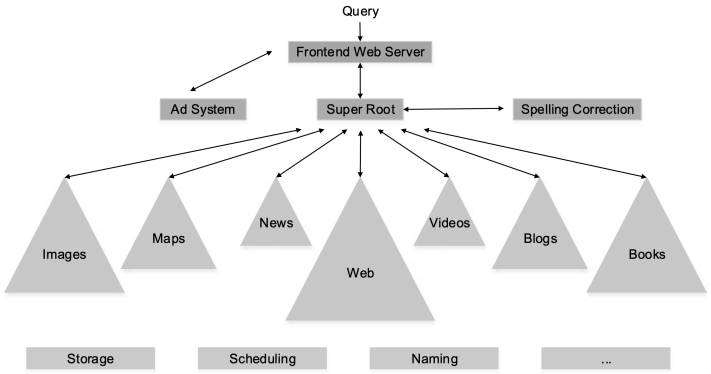
Intelligent service monitoring is **in need**

Key Qualities of Intelligent Service Monitoring

Large scale and complexity



Fast development iteration



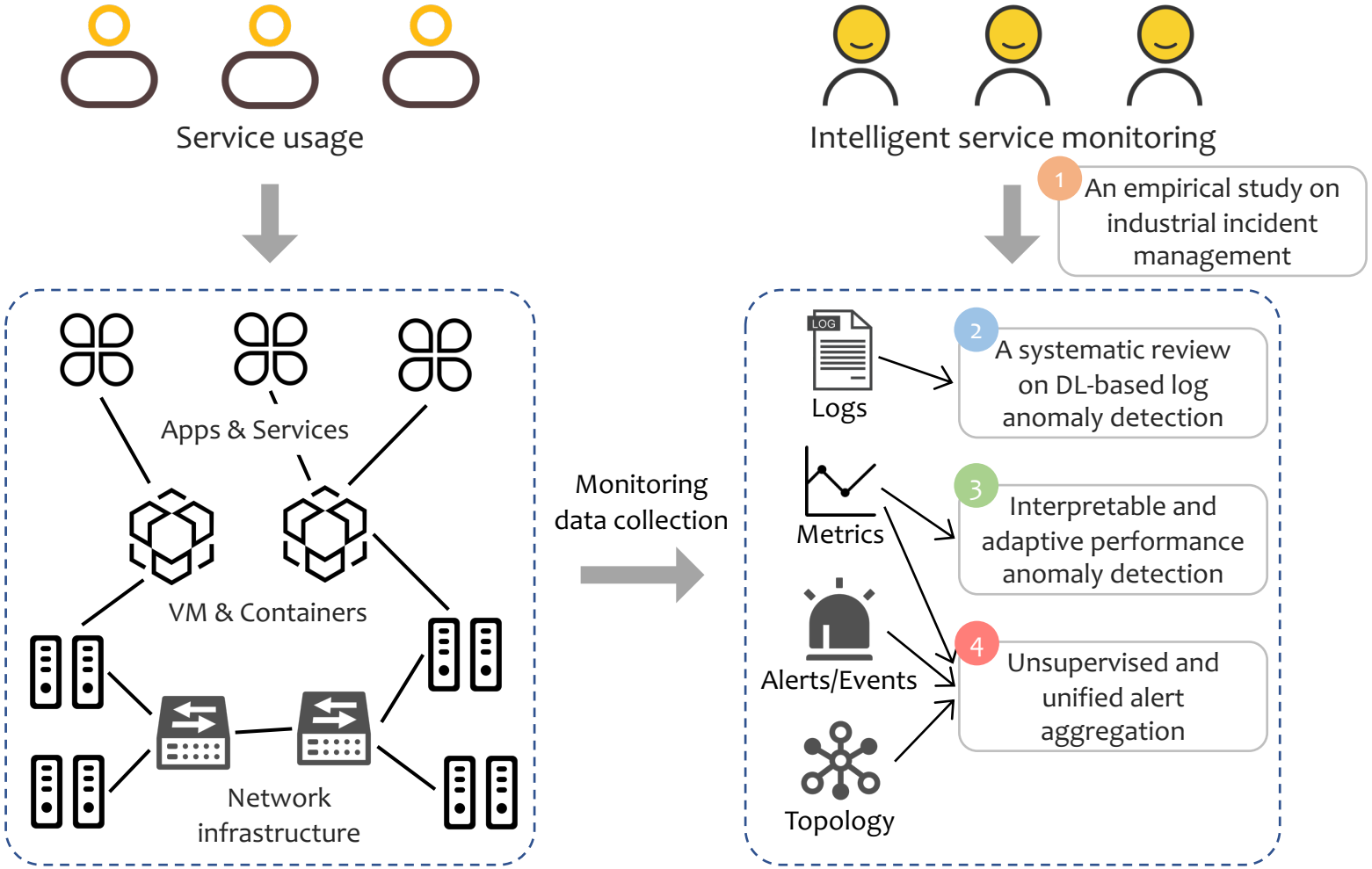
Complicated service dependencies

Good performance: accurate, fast, and high-coverage

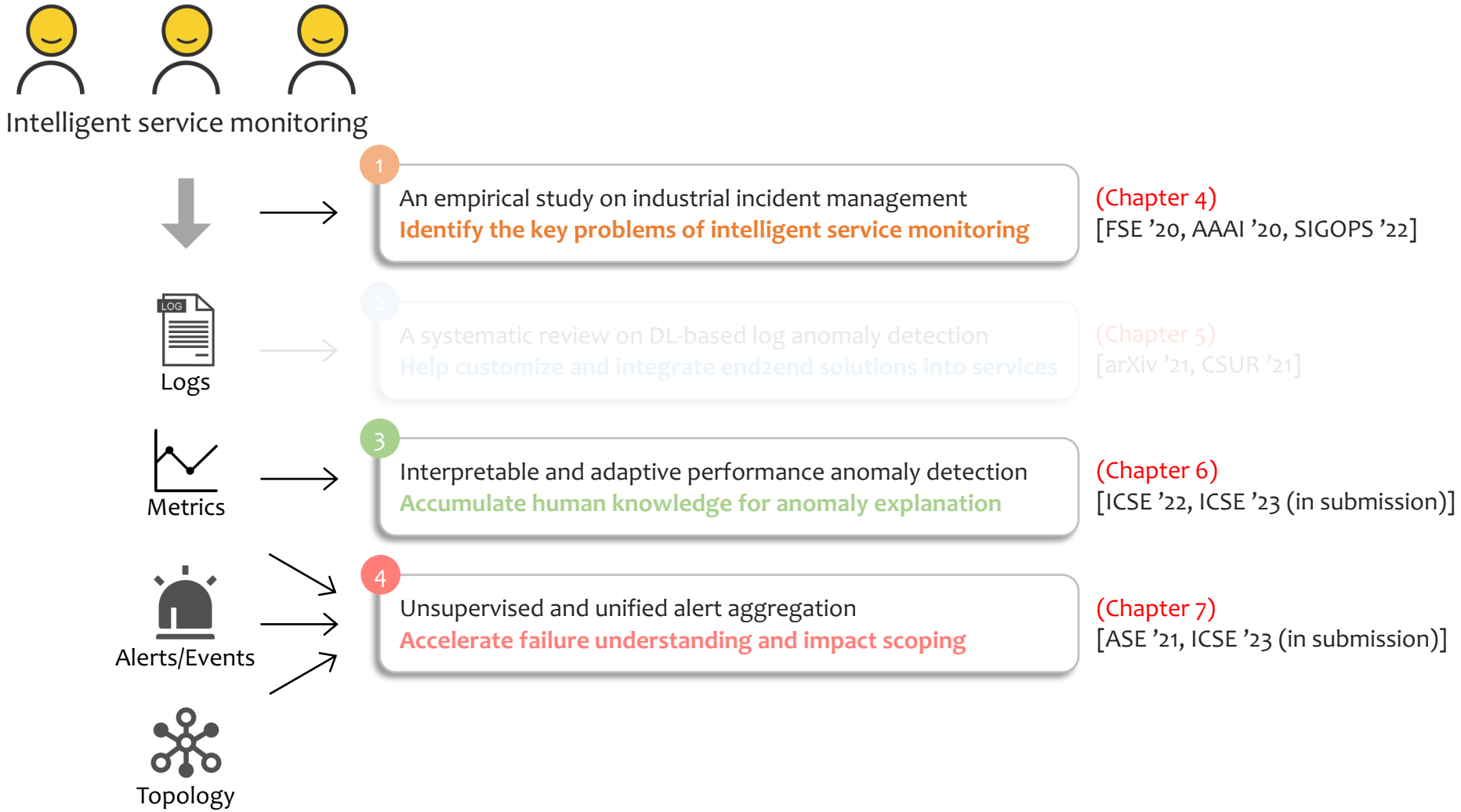
Adaptivity and interpretability

Impact scope estimation

Intelligent Service Monitoring



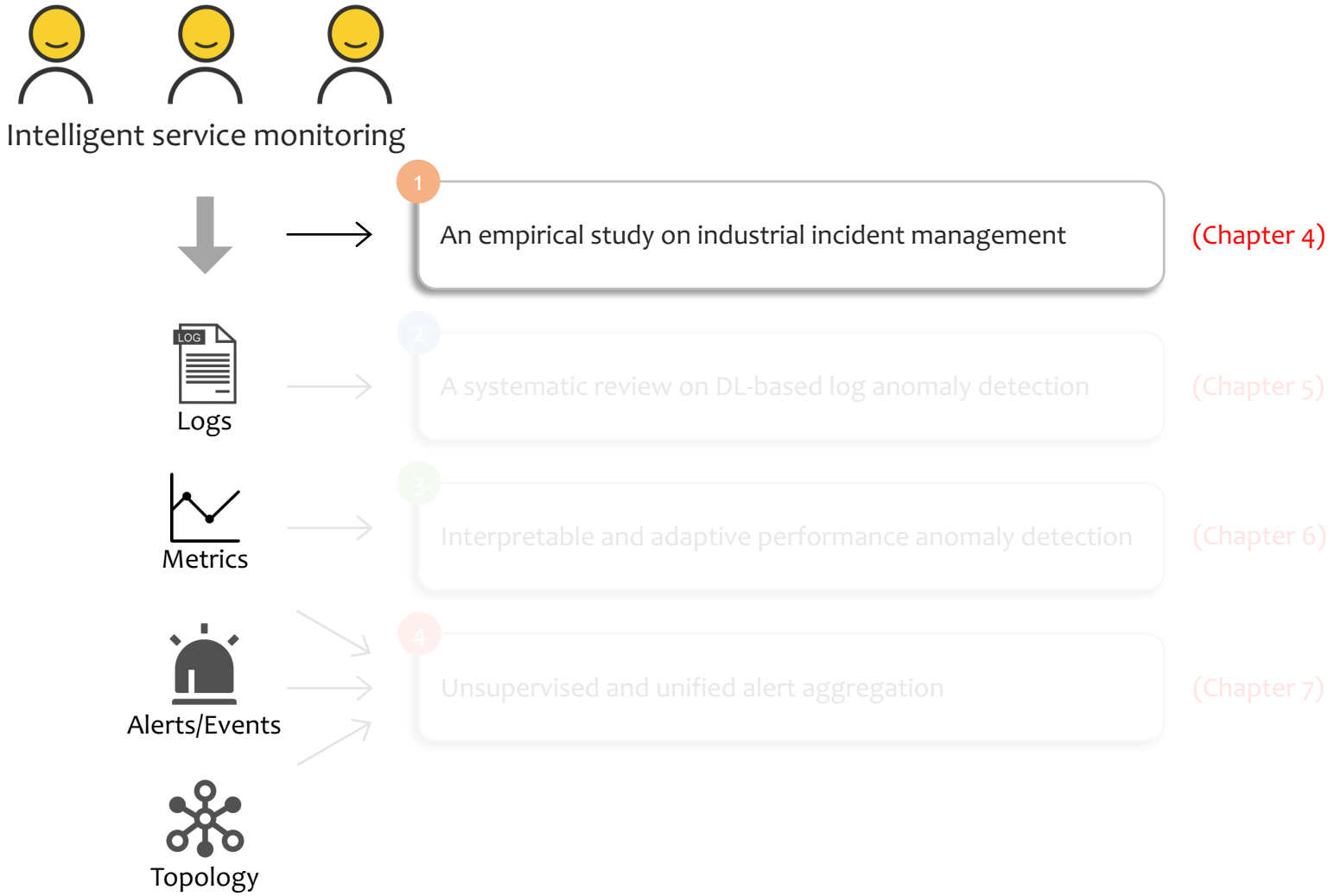
Thesis Contributions



Outline

- Topic 1: An empirical study on industrial incident management
- Topic 2: Interpretable and adaptive performance anomaly detection
- Topic 3: Unsupervised and unified alert aggregation
- Conclusion and Future work

Outline


Intelligent service monitoring
Logs
Metrics
Alerts/Events
Topology
1
An empirical study on industrial incident management
(Chapter 4)
2
A systematic review on DL-based log anomaly detection
(Chapter 5)
3
Interpretable and adaptive performance anomaly detection
(Chapter 6)
4
Unsupervised and unified alert aggregation
(Chapter 7)

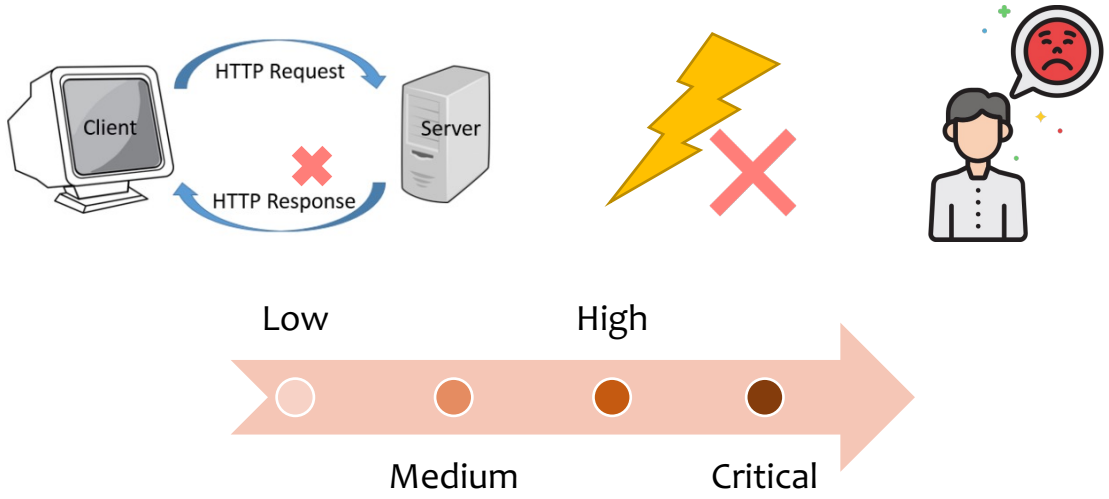
Content

- Topic 1: An empirical study on industrial incident management
  - ✓ Motivation & methodology
  - ✓ Incident characteristics
  - ✓ Key challenges of incident management
  - ✓ Summary

What is a Service Incident?

Service interruption or performance degradation

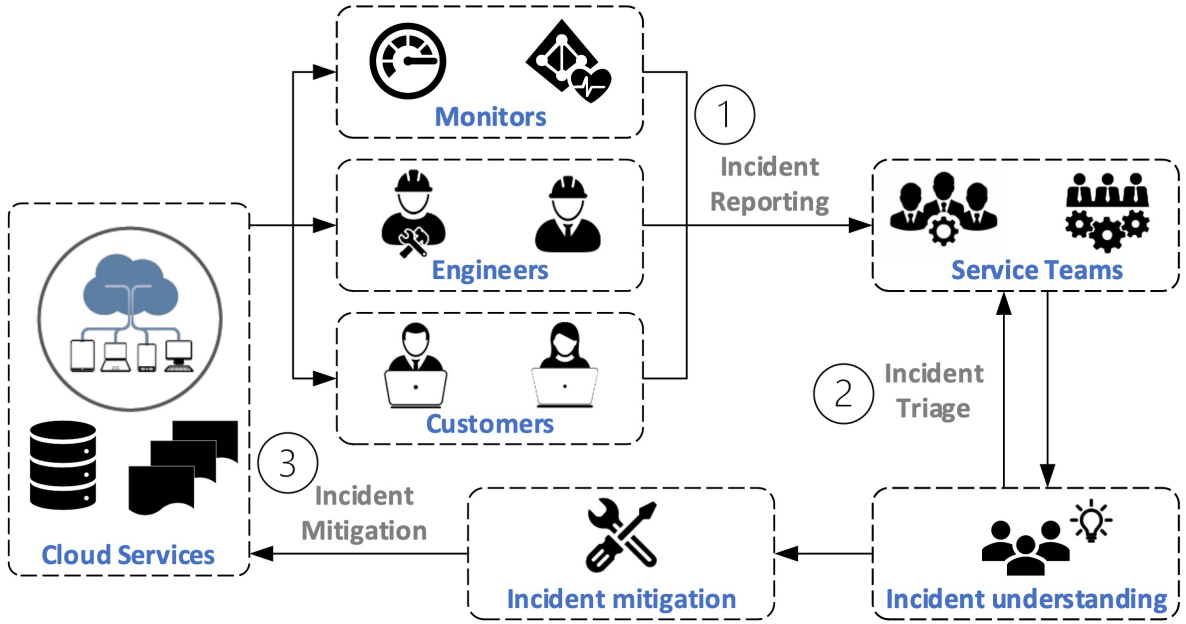
- Is or will be affecting user experience
- Can be referred to as *failure*
- Examples
  - ✓ Bad HTTP requests
  - ✓ Power outages
  - ✓ Customer-reported errors

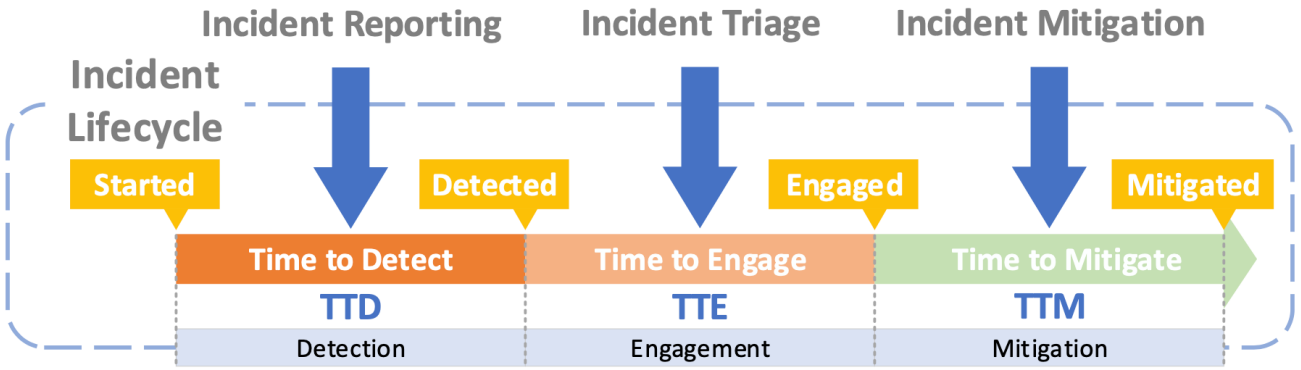


Incident Management Procedure

Incident management procedure

- Incident reporting
- Incident triage
- Incident mitigation





Motivation

- A lack of comprehensive study of incident management
- Understand the key challenges of incident handling
- Identify the unaddressed problems of service monitoring

Methodology

Raw dataset
- Two years of incident tickets at Microsoft

Six core services
- Datacenter Management (DCM)
- Networking
- Storage
- Compute
- Database
- Web Service (WS)

Study approaches
- Incident ticket analysis
- Field studies
- Validation through quantitative experiments

Incident ID Resolved Critical	***Disk firmware update disabled disk cache***	
	Service: Storage	# of impacted requests: ~100,000
	Datacenter: DC #4	# of impacted accounts: ~10,000

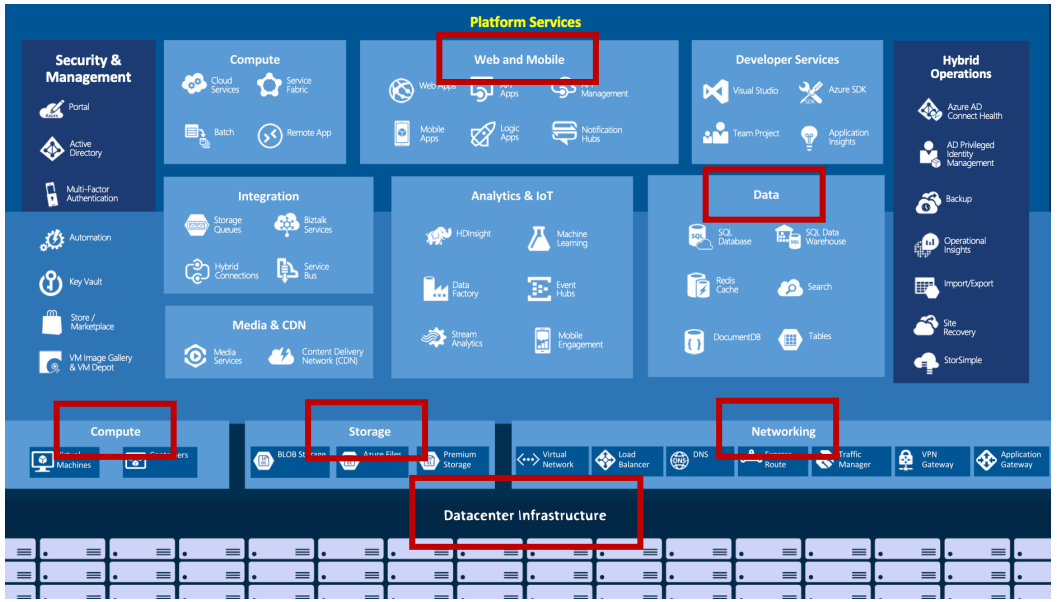
Summary

Writing to a big data storage platform experienced high failure counts.

Diagnosis

Firmware upgrade to a game drive service inadvertently disabled write cache. At the beginning, there was no direct impact on the service because the number of machines getting into bad state was small and the system was built to tolerate such instances. However, as more and more machines were getting upgraded, the overall latency of the service stack was slowly accumulating and at some point got tipped. It took quite some time to detect the incident which unfortunately deteriorated into a critical issue.

An example of incident ticket



The cloud stack of Microsoft Azure

Content

- Topic 1: An empirical study on industrial incident management
  - ✓ Motivation & Study methodology
  - ✓ Incident characteristics
  - ✓ Key challenges of incident management
  - ✓ Summary

Incident Characteristics

Incident root causes

- Human Errors
- Network Issues
- Deployment Issues
- External Issues
- Capacity Issues
- Others

30.6%

Root Cause	Dist.	Root Cause	Dist.
Network (Hardware)	**22.95%**	Human Error (Code Defect)	**19.23%**
Network (Connectivity)	2.24%	Human Error (Config.)	**7.45%**
Network (Config.)	0.89%	Human Error (Design Flaw)	5.66%
Network (Other)	4.47%	Human Error (Integration)	2.09%
Deployment (Upgrade)	5.22%	Human Error (Other)	2.83%
Deployment (Config.)	3.87%	External Issue (Partner)	2.83%
Deployment (Other)	1.19%	External Issue (Other)	1.64%
Capacity Issue	**6.56%**	Others	10.88%

37.3%

Distribution of incident root causes

Incident Characteristics

Incident severity

- Low + Medium incidents > 90%
- Critical incidents [0.01%, 0.4%]

	DCM	Network	Storage	Compute	Database	WS
Critical	0.01%	0.01%	0.01%	0.31%	0.40%	0.07%
High	5.48%	1.21%	2.57%	5.27%	4.32%	3.33%
Medium	86.65%	46.90%	43.32%	74.19%	63.93%	84.52%
Low	7.86%	51.88%	54.10%	20.23%	31.35%	12.08%

Distribution of incident severity

Incident fixing time

- In many cases, the time Critical incidents take is larger than the sum of others

	DCM	Network	Storage	Compute	Database	WS
Critical	**38.33x**	8.46x	**10.06x**	**142.05x**	**209.97x**	**286.6x**
High	19.25x	3.18x	2.52x	2.56x	5.75x	3.56x
Medium	1x	**9.8x**	7.09x	2.95x	25.28x	12.93x
Low	3.01x	5.49x	1.09x	11.65x	2.41x	144.79x

Distribution of incident fixing time

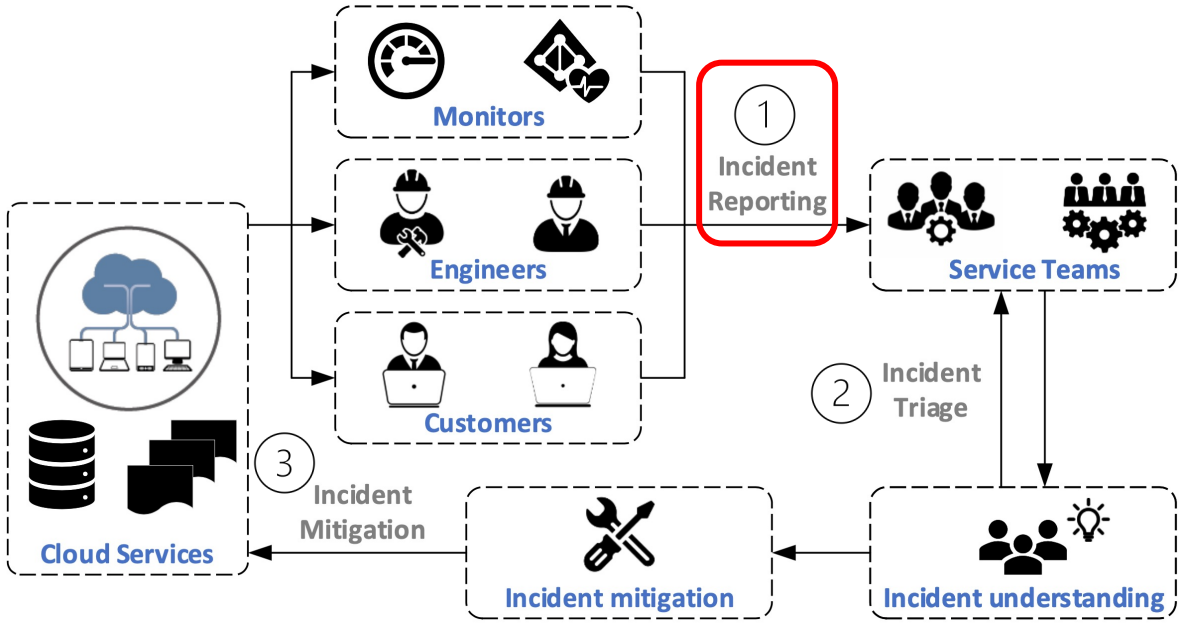
Content

- Topic 1: An empirical study on industrial incident management
  - ✓ Motivation & Study methodology
  - ✓ Incident characteristics
  - ✓ Key challenges of incident management
  - ✓ Summary

Key Challenges of Incident Management

Challenge 1: Resource health assessment

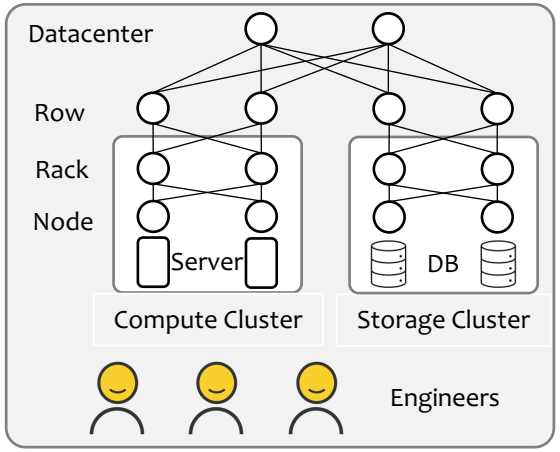
- Problem detection based on various signals (metrics, logs, etc.)
- Hard-to-understand problems with complex and changing patterns



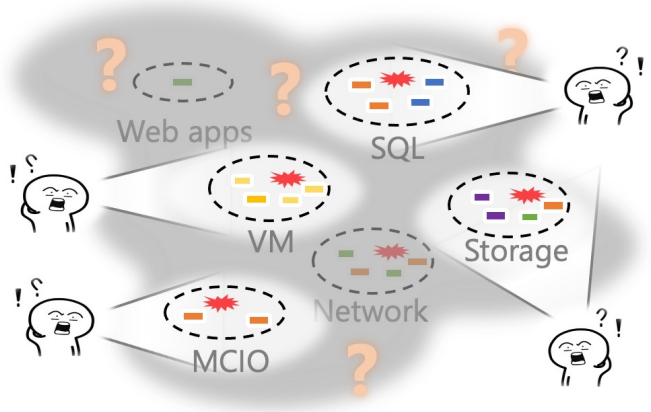
Key Challenges of Incident Management

Challenge 1: Resource health assessment

- Problem detection based on various signals (metrics, logs, etc.)
- Hard-to-understand problems with complex and changing patterns



Flooding alarms



Gray failures

Subtle failures that defy quick and definitive detection [1].

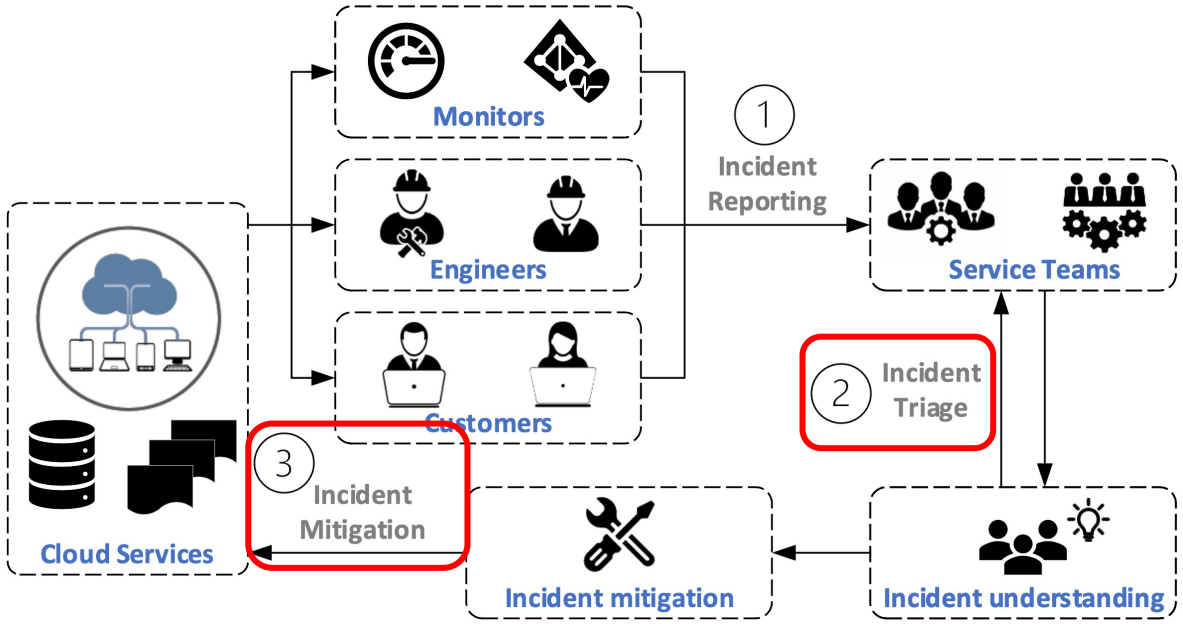
Accurate, adaptive, and interpretable anomaly detection alleviates flooding alarms and gray failures [Topic 2]

[1] Huang et al. Gray Failure: The Achilles' Heel of Cloud-Scale Systems. HotOS '17.

Key Challenges of Incident Management

Challenge 2: Resource dependency discovery

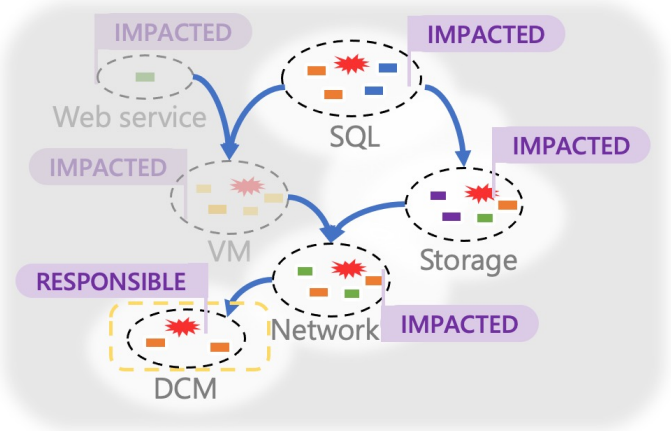
- Services rely on each other (microservices)
- Incomplete, outdated, and human-dependent



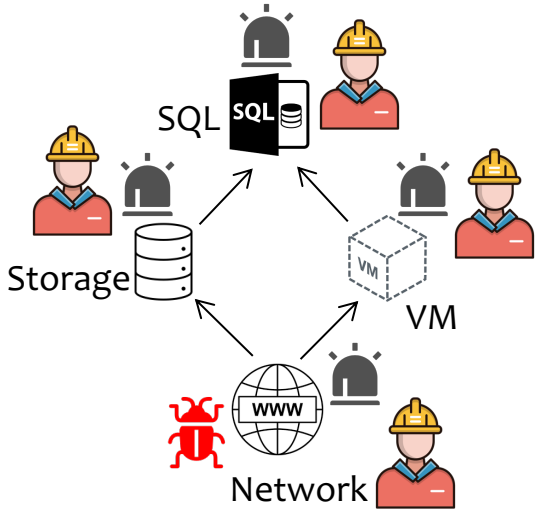
Key Challenges of Incident Management

Challenge 2: Resource dependency discovery

- Services rely on each other (microservices)
- Incomplete, outdated, and human-dependent



Imprecise impact estimation



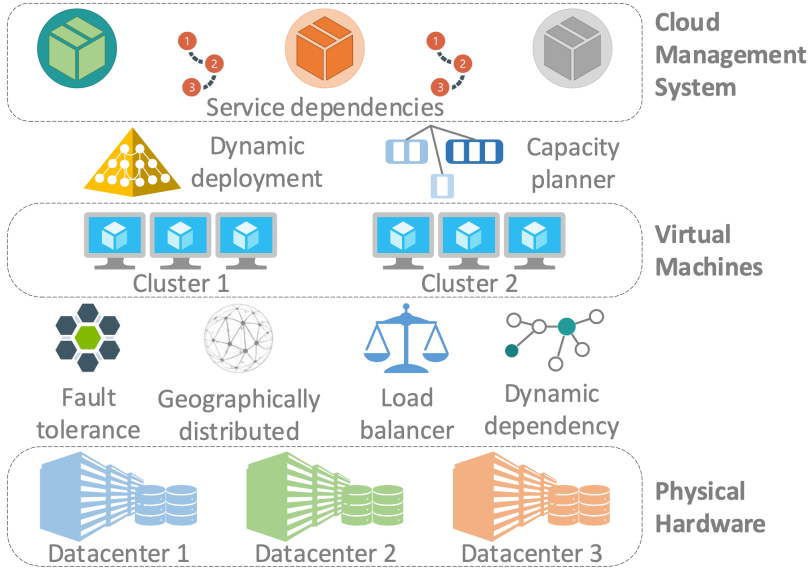
Redundant engineering efforts

Identifying related problems facilitates failure impact estimation and duplicate effort saving [Topic 3]

Understanding the Key Challenges

Challenge 1: Resource health assessment
- System fault tolerance
- Monitor design and distribution
- …

Challenge 2: Resource dependency discovery
- Software system modularity
- Physical infrastructure virtualization
- Dynamic deployment
- Load balancing
- …



A typical cloud computing architecture

Incident ID Resolved Critical	*Disk firmware update disabled disk cache*	
	Service: Storage	# of impacted requests: ~100,000
	Datacenter: DC #4	# of impacted accounts: ~10,000

Summary
Writing to a big data storage platform experienced high failure counts.

Diagnosis
Firmware upgrade to a game drive service inadvertently disabled write cache. At the beginning, there was no direct impact on the service because the number of machines getting into bad state was small and the system was built to tolerate such instances. However, as more and more machines were getting upgraded, the overall latency of the service stack was slowly accumulating and at some point got tipped. It took quite some time to detect the incident which unfortunately deteriorated into a critical issue.

An incident showing Challenge 1

Incident ID Resolved Critical	*A high error rate of operation [API] has been seen*	
	Service: CRM	# of impacted requests: ~1,000,000
	Datacenter: DC #2	# of impacted accounts: ~10,000

Summary
Monitor has detected multiple VMs and web applications unavailable.

Diagnosis
Some operations of Cloud Resource Management (CRM) service suffered from a high error rate. Engineering team found the frontend web service was in a loop of crash and reboot. This resulted in customer requests being held for an extended period of time in web server request queue, leading to slow responses and request timeouts. More than five other services suffered from different failures such as login failures, request timeout errors, etc. The cascading effects and implicit service dependencies made the engineering team hard to know and notify all impacted service teams, especially during busy bug fixing time. Therefore, many impacted services received failure reports and diagnosed their services independently. Particularly, an IT Management Software (ITMS) service attributed the failures to DNS service due to the direct dependency. However, the DNS service was managed by the CRM service (the true root cause), which took ITMS team some time to figure out.

An incident showing Challenge 2

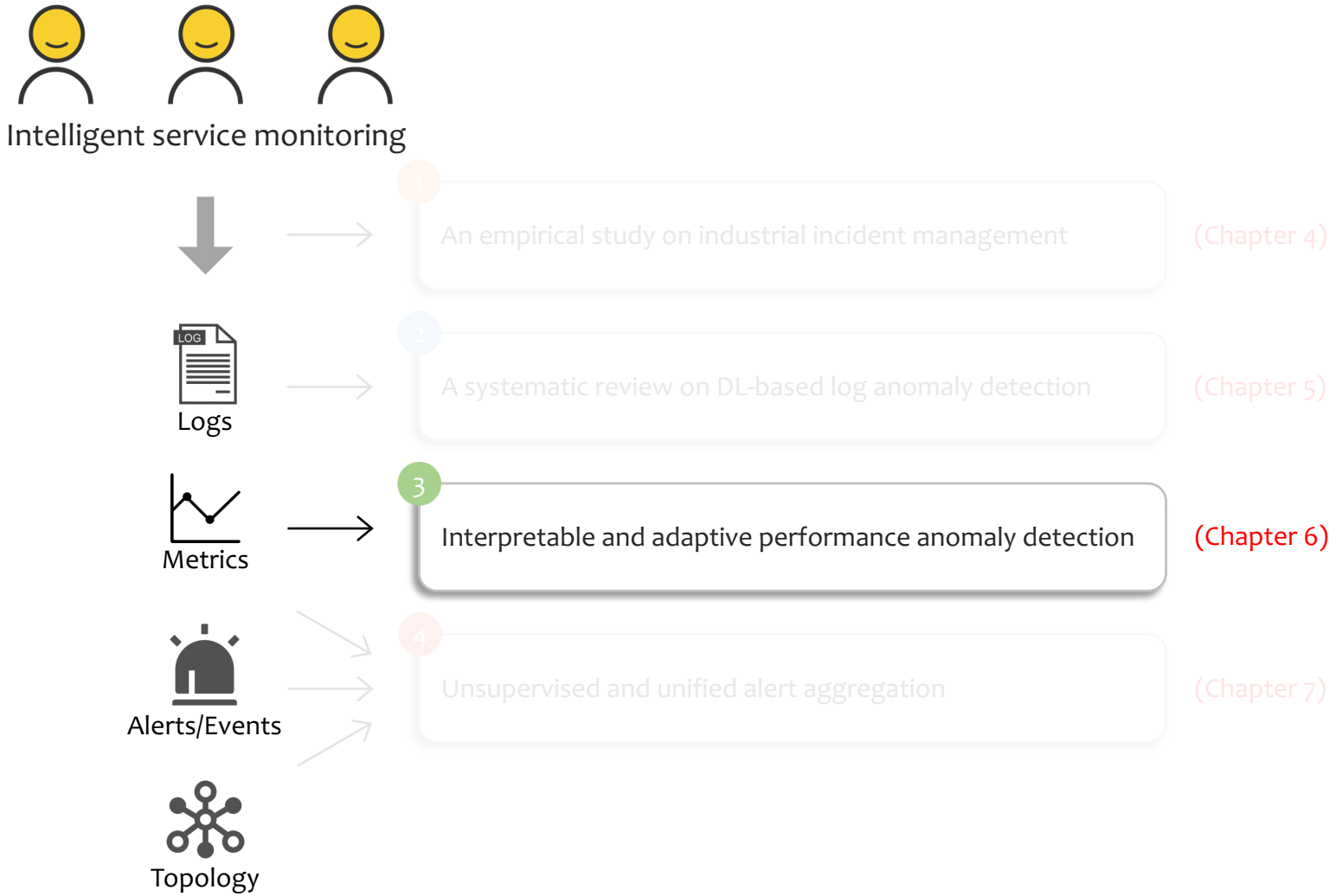
Content

- Topic 1: An empirical study on industrial incident management
  - ✓ Motivation & Study methodology
  - ✓ Incident characteristics
  - ✓ Key challenges of incident management
  - ✓ Summary

Summary of Topic 1

- A comprehensive study of industrial incident management
- The general management procedure of incidents and their characteristics
- Study the key challenges of incident handling and the underlying reasons
- Findings motivate the studies in Topic 2 and Topic 3

Outline


Intelligent service monitoring
Logs
Metrics
Alerts/Events
Topology
1
An empirical study on industrial incident management
(Chapter 4)
2
A systematic review on DL-based log anomaly detection
(Chapter 5)
3
Interpretable and adaptive performance anomaly detection
(Chapter 6)
4
Unsupervised and unified alert aggregation
(Chapter 7)

Content

- Topic 2: Interpretable and adaptive performance anomaly detection
  - ✓ Motivation
  - ✓ Anomaly detection based on pattern sketching
  - ✓ Evaluation
  - ✓ Summary

Performance Anomaly Detection

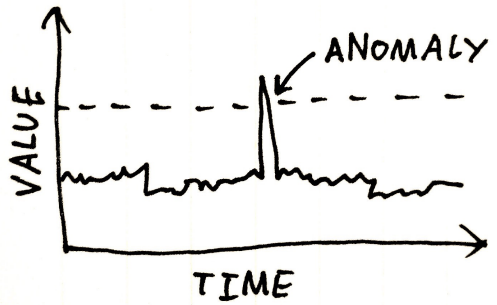
Performance anomalies

- Slow service response
- High temperature
- …

An anomaly is an observation or a sequence of observations which deviates remarkably from the general distribution of data [1].

Service performance is monitored with metrics

- Request latency
- Request success rate
- Traffic volume
- …



[1] Braei et al. Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art. arXiv '22.

Why Yet Another Detection Algorithm?

Indeed, many existing unsupervised approaches

- Forecasting-based: LSTM
- Reconstruction-based: Donut, LSTM-VAE
- Probabilistic: LODA, DAGMM, Extreme Value Theory
- Tree-based: Isolation Forest
- Others: SR-CNN, …

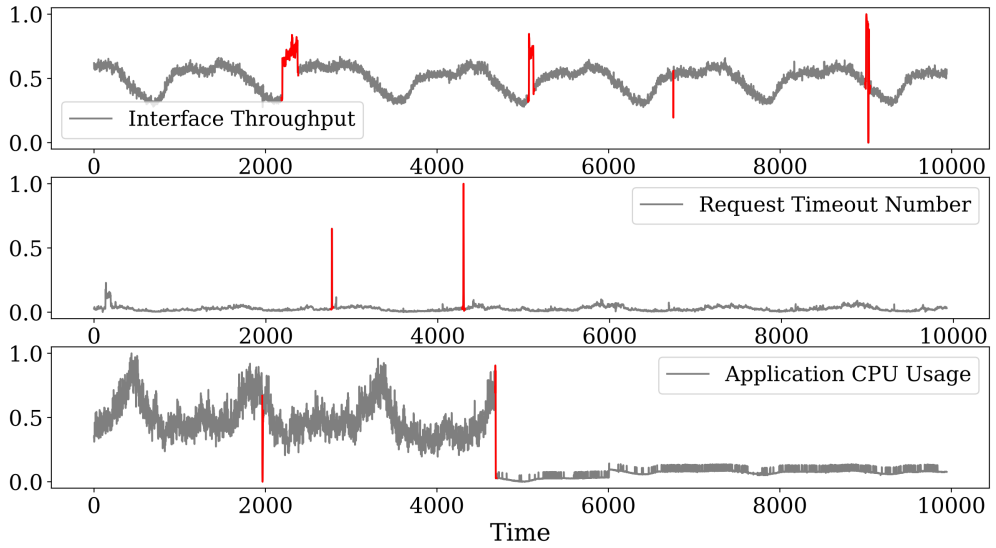
In production, we need

- Interpretability: gain engineers' trust, accelerate failure understanding
- Online adaptability: accommodate unseen patterns
- Human knowledge reusage: valuable company asset

Motivating Observations

Key observations

- Metric time series tends to develop individual and stable patterns
  - ✓ A metric pattern: repeated similar subsequences
  - ✓ Similar observations have been made [1-3]
- Similar anomalies incur similar anomalous patterns on the metric time series [4]



Anomalous patterns captured in Huawei Cloud

- ✓ Find metric patterns
- ✓ Distinguish the anomalous patterns from the normal ones
- ✓ Adapt to unseen patterns

[1] Hu et al. TS-InvarNet: Anomaly Detection and Localization based on Tempo-spatial KPI Invariants in Distributed Services. ICWS '22.
[2] Wu et al. Identifying Root-Cause Metrics for Incident Diagnosis in Online Service Systems. ISSRE '21.
[3] Ma et al. Diagnosing root causes of intermittent slow queries in cloud databases. VLDB '20.
[4] Lim et al. Identifying Recurrent and Unknown Performance Issues. ICDM '14.

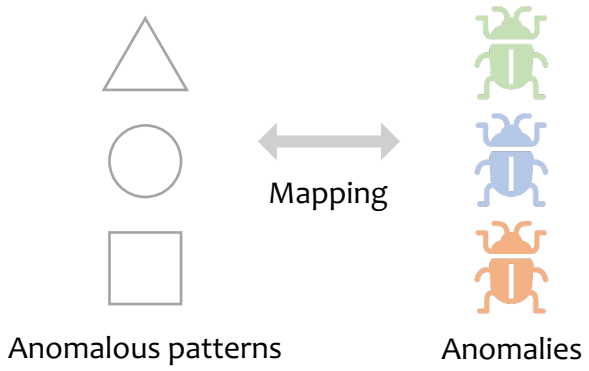
Motivating Observations

Anomaly detection strategy – Pattern Sketching

- When a service runs normally, it produces normal patterns
- If a new pattern deviates substantially from the normal ones, it could be abnormal

Interpretability

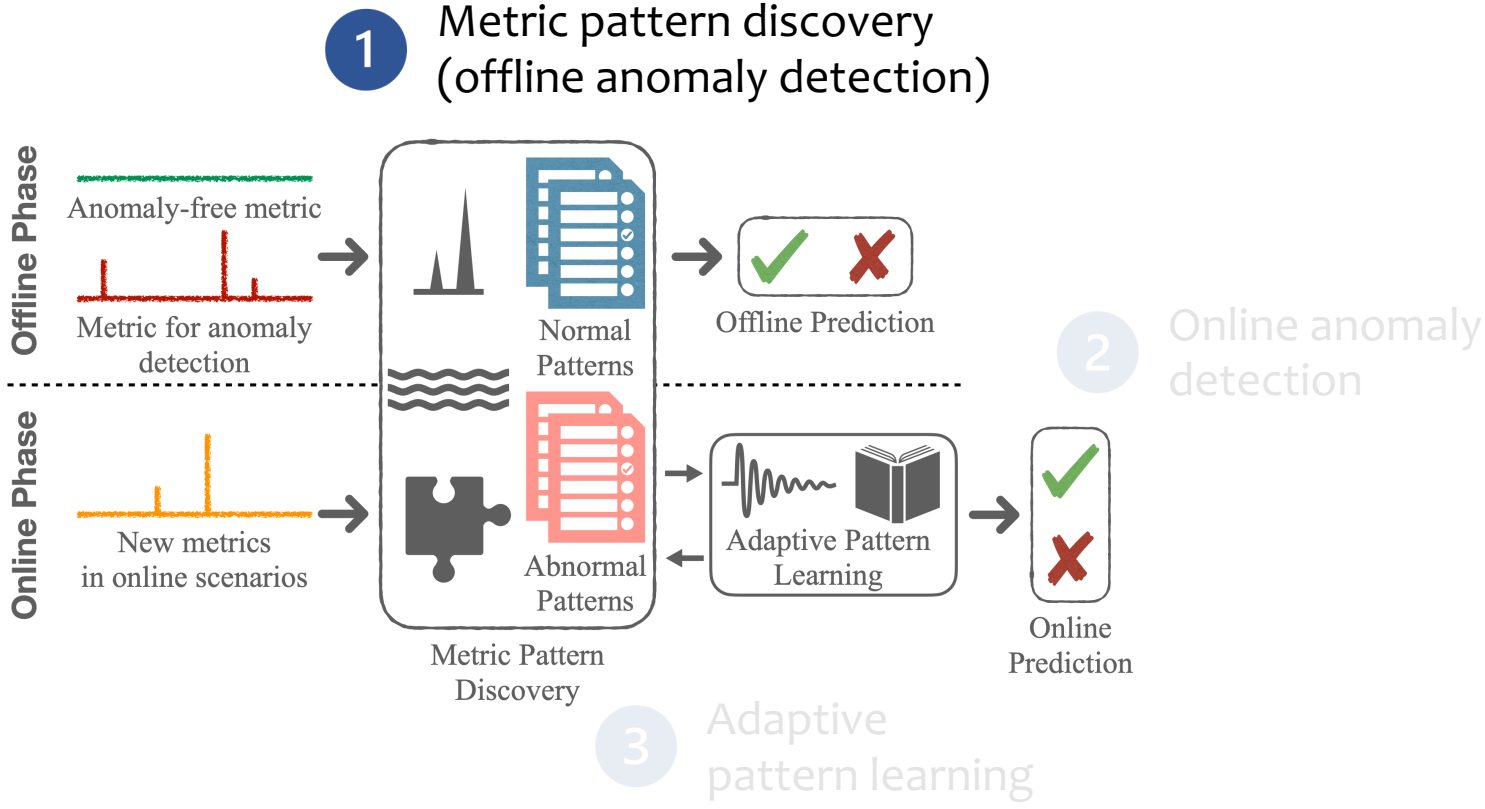
- If a known abnormal patterns is detected, we know what performance anomalies have happened



Content

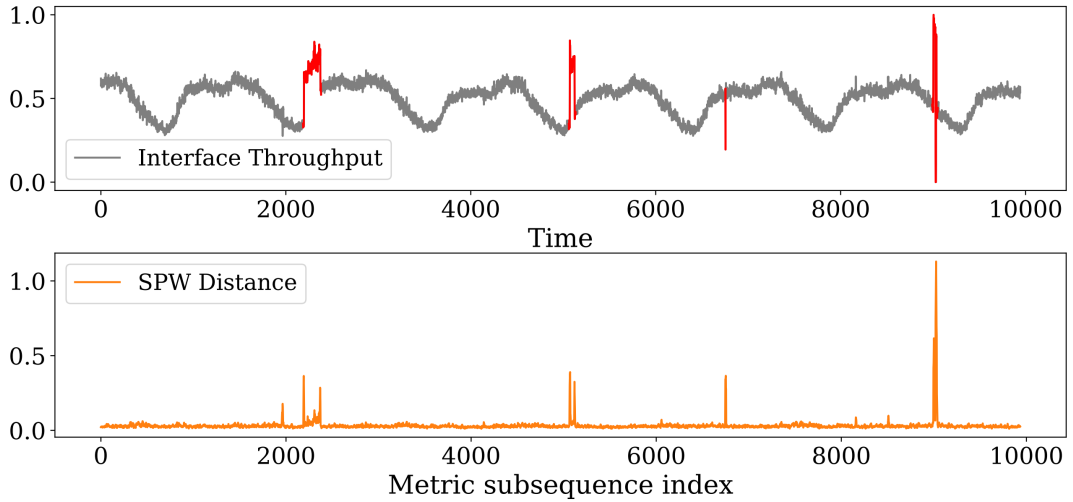
- Topic 2: Interpretable and adaptive performance anomaly detection
  - ✓ Motivation
  - ✓ Anomaly detection based on pattern sketching
  - ✓ Evaluation
  - ✓ Summary

ADSketch Overview



The Smallest Pair-Wise (SPW) Distance

- A subsequence: a continuous part of a metric time series
- The SPW distance of a subsequence: its smallest distance to other subsequences
- If a subsequence has a large SPW distance , it is likely an anomaly



The SPW distance of a metric time series

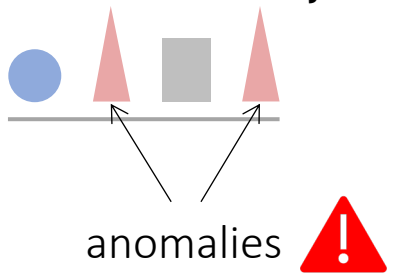
- Brute-force searching is not scalable
- STAMP [1] is faster by orders of magnitude
  - ✓ Fast Fourier Transform (FFT)

[1] Yeh et al. Matrix profile I: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. ICDM '16.

Metric Pattern Discovery

- Algorithm inputs
  - ✓ 1. Anomaly-free time series

  - 2. Time series for anomaly detection

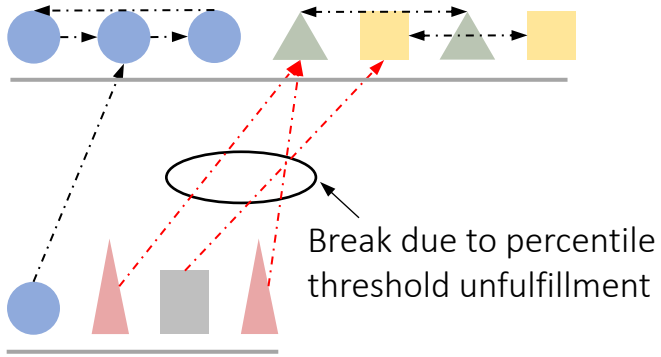


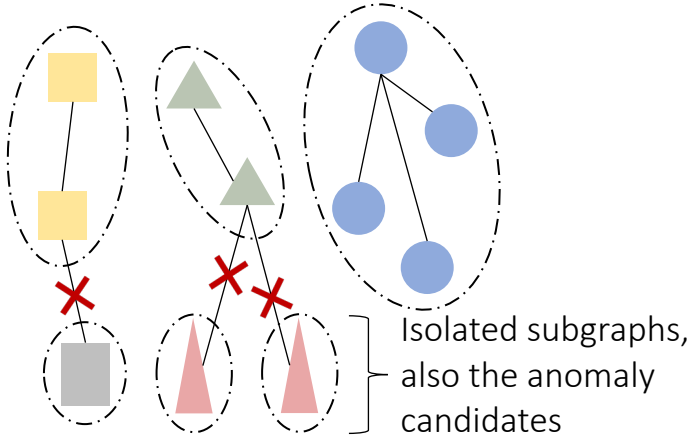
- Algorithm outputs
  - ✓ Anomalies
  - ✓ Normal and abnormal patterns

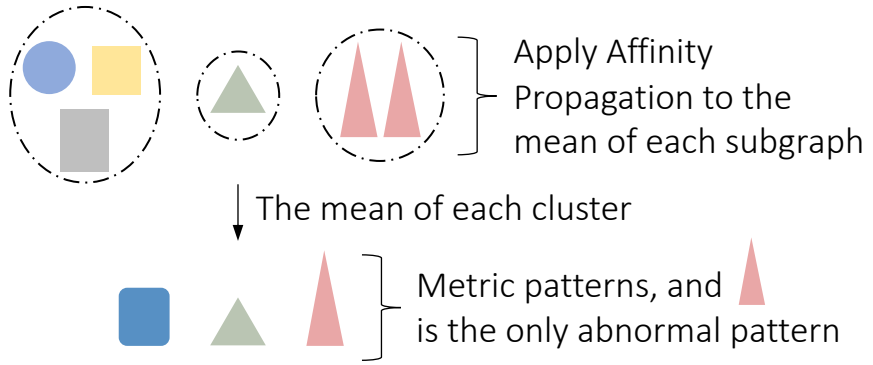
Algorithm 1: Performance Anomaly Pattern Discovery

Input: $\mathcal{T}_n$, $\mathcal{T}_a$, m, and p
Output: Two disjoint sets of $\mathcal{P}_n$ and $\mathcal{P}_a$

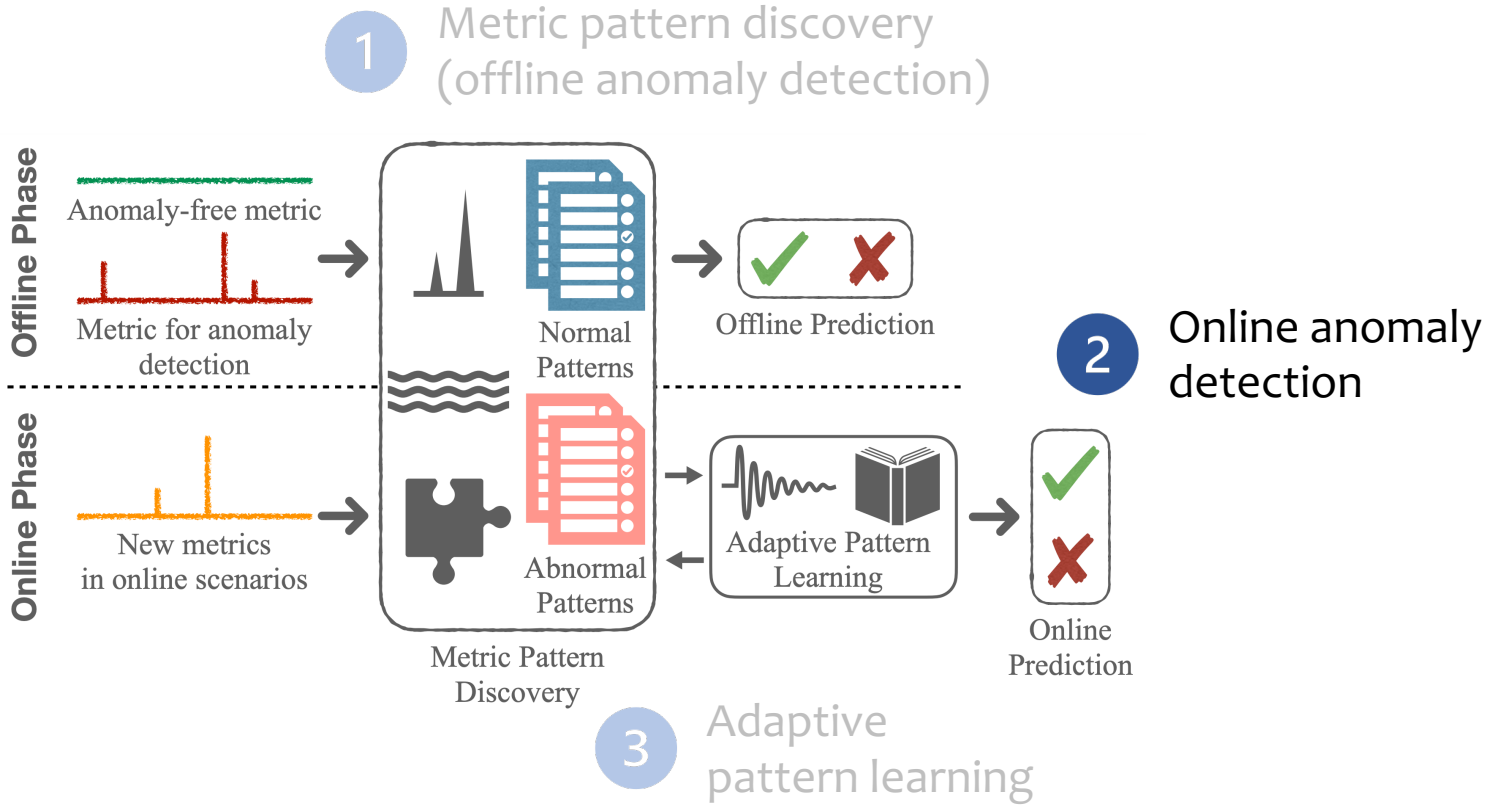
$\mathcal{I}_{nn}, \mathcal{S}_{nn} \leftarrow \text{STAMP}(\mathcal{T}_n, \mathcal{T}_n, m)$
$\mathcal{I}_{na}, \mathcal{S}_{na} \leftarrow \text{STAMP}(\mathcal{T}_n, \mathcal{T}_a, m)$
$G \leftarrow \text{ConnectedSubgraphs}(\mathcal{I}_{nn} + \mathcal{I}_{na}, \mathcal{S}_{na}, p)$
$N_i \leftarrow \text{IsolatedNodes}(G)$
$\mu_G \leftarrow \text{GraphWiseMean}(G)$
$C \leftarrow \text{AffinityPropagation}(\mu_G)$
$\mu_C \leftarrow \text{ClusterWiseMean}(C)$
$\mathcal{P}_n \leftarrow \text{EmptyArray}, \mathcal{P}_a \leftarrow \text{EmptyArray}$
for *each idx in* $1 : \text{Size}(C)$ **do**
    // $C[idx]$: all subsequences in the cluster
    if $C[idx] \subset N_i$ **then**
        $\mathcal{P}_a \leftarrow$ Append $\mathcal{P}_a$ with idx
    else
        $\mathcal{P}_n \leftarrow$ Append $\mathcal{P}_n$ with idx
    end
end







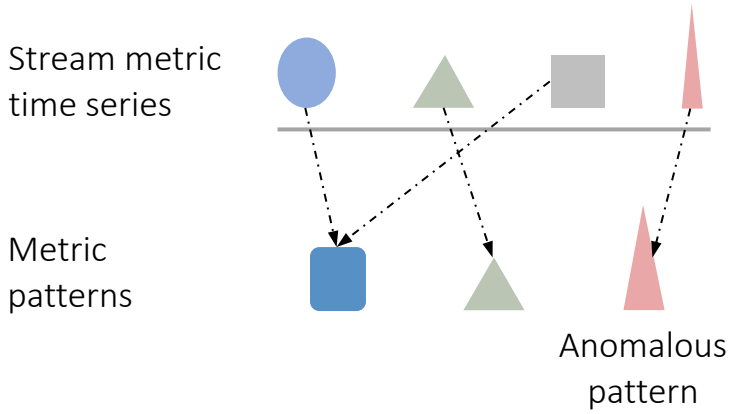
ADSketch Overview



Online Anomaly Detection

- Algorithm inputs
  - ✓ Streaming time series for anomaly detection

- Algorithm outputs
  - ✓ Anomalies in the time series



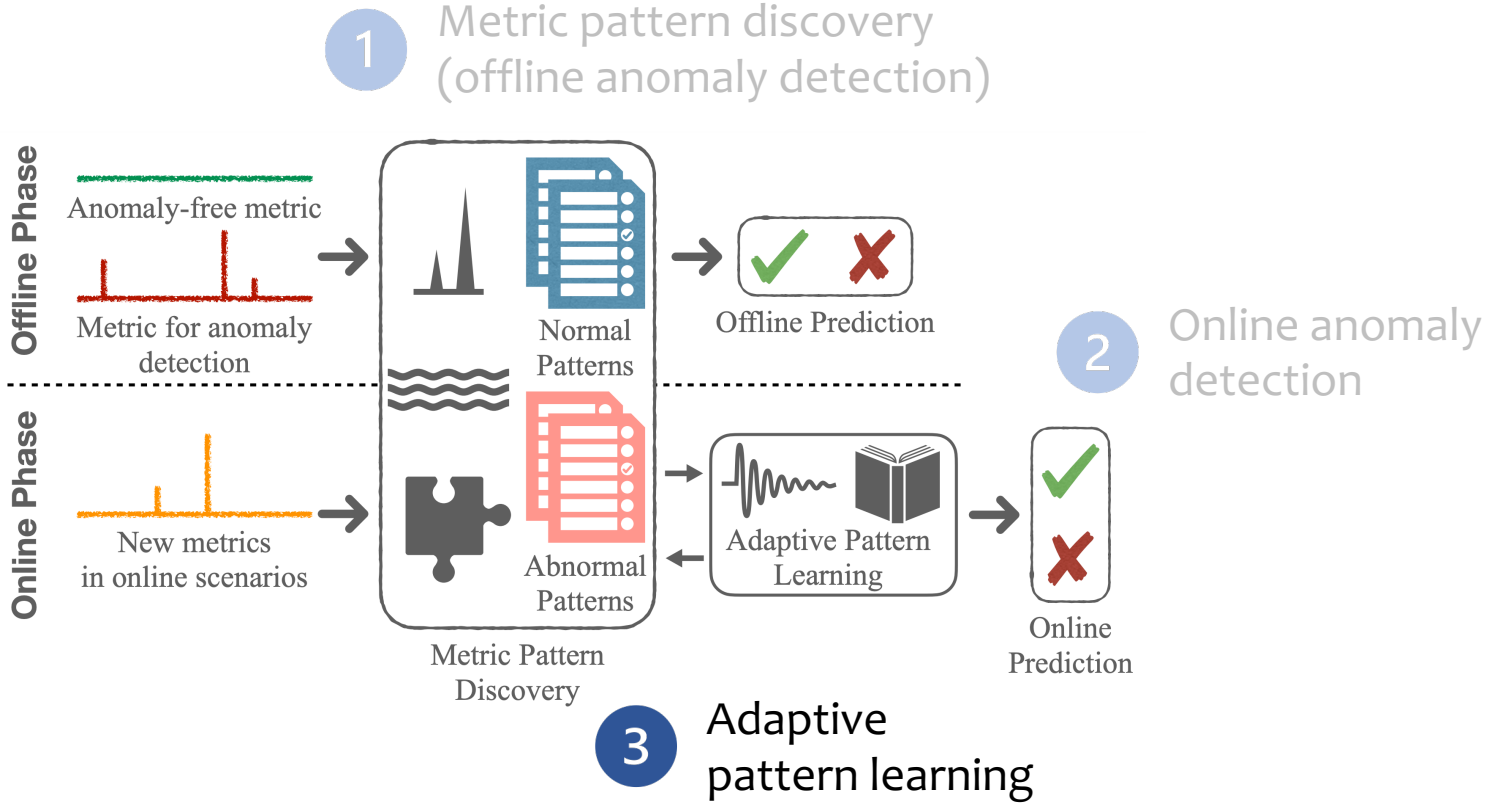
Algorithm 2: Performance Anomaly Detection

Input: t, $\mathcal{P}_a$, and μ_C
Output: Anomaly detection result for t

```
1 D_t ← PairWiseDistance(t, μ_C)
2 idx ← MinIndex(D_t)
3 if idx ∈ P_a then
4 |   return True
5 else
6 |   return False
7 end
```

Prediction results

ADSketch Overview



Adaptive Pattern Learning

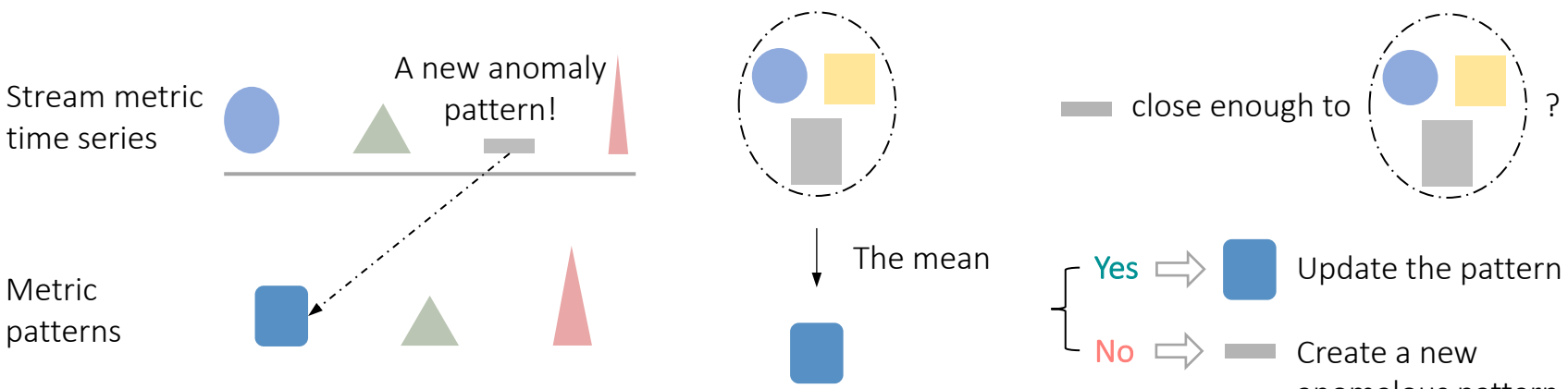
- Algorithm inputs
  - ✓ Streaming time series for anomaly detection

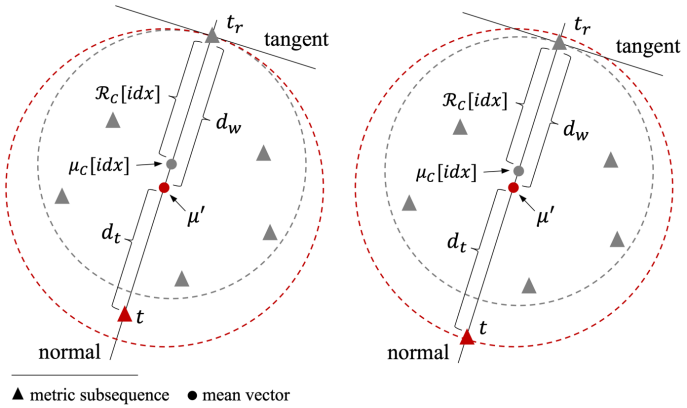
- Algorithm outputs
  - ✓ Anomalies
  - ✓ Updated metric patterns

Algorithm 3: Adaptive Pattern Learning

Input: t, $\mathcal{P}_n$, $\mathcal{P}_a$, μ_C, $\mathcal{S}_C$, and $\mathcal{R}_C$
Output: Updated variables: $\mathcal{P}_n$, $\mathcal{P}_a$, μ_C, $\mathcal{S}_C$, and $\mathcal{R}_C$

$\mathcal{D}_t \leftarrow \text{PairWiseDistance}(t, \mu_C)$
$idx \leftarrow \text{MinIndex}(\mathcal{D}_t)$
$\mu' \leftarrow (\mu_G[idx] \times \mathcal{S}_C[idx] + t)/(\mathcal{S}_C[idx] + 1)$
$d_w \leftarrow \text{Distance}(\mu_C[idx], \mu') + \mathcal{R}_C[idx]$
$d_t \leftarrow \text{Distance}(t, \mu')$
$d' \leftarrow \text{Max}(d_t, d_w)$
$d_n, d_a \leftarrow \text{Max}(\mathcal{R}_C[\mathcal{P}_n]), \text{Max}(\mathcal{R}_C[\mathcal{P}_a])$
if $idx \in \mathcal{P}_a$ **then** $d \leftarrow d_a$ **else** $d \leftarrow d_n$ **end**
if $\mathcal{D}_t[idx] < d$ **then**
  // add t to the most similar cluster
  $\mu_C[idx], \mathcal{S}_C[idx], \mathcal{R}_C[idx] \leftarrow \mu', \mathcal{S}_C[idx] + 1, d'$
  if $\mathcal{S}_C[idx] > \text{Max}(\mathcal{S}_C[\mathcal{P}_a])$ and *idx is a new cluster* **then**
    $\mathcal{P}_n \leftarrow$ Append $\mathcal{P}_n$ with idx
    $\mathcal{P}_a \leftarrow$ Remove idx from $\mathcal{P}_a$
  else
    $d \leftarrow \text{Max}(d, d')$ // d will be assigned to d_n or d_a accordingly
  end
else
  // create a new anomalous cluster for t
  $\mathcal{P}_a \leftarrow$ Append $\mathcal{P}_a$ with $\text{Length}(\mu_C) + 1$
  $\mu_G \leftarrow$ Append μ_G with t
  $\mathcal{R}_C \leftarrow$ Append $\mathcal{R}_C$ with 0
  $\mathcal{S}_C \leftarrow$ Append $\mathcal{S}_C$ with 1
end





The worst case for pattern updates

Complexity Analysis

- Time complexity
  - ✓ The closest pair searching: $\mathcal{O}(n^2)$
  - ✓ Affine propagation algorithm: $\mathcal{O}(|C|^2)$, $|C|$ is the number of clusters, which is small
  - ✓ Online anomaly detection and pattern updating: $\mathcal{O}(n)$
  - ✓ Overall: $\mathcal{O}(n^2)$
  - ✓ Easily parallelizable
  - ✓ Ultra-fast approximation is attainable

- Space complexity
  - ✓ The indexes of metric patterns: $\mathcal{O}(|C|)$
  - ✓ The storage of metric patterns: $\mathcal{O}(m \times |C|)$, m is the length of subsequences
  - ✓ Our design makes it trivial

Content

- Topic 2: Interpretable and adaptive performance anomaly detection
  - ✓ Motivation
  - ✓ Anomaly detection based on pattern sketching
  - ✓ Evaluation
  - ✓ Summary

Evaluation Questions

- RQ1: How effective is ADSketch's **offline** anomaly detection?
- RQ2: How effective is ADSketch's **online** anomaly detection?
- RQ3: How effective is ADSketch's **adaptive** pattern learning?

Experiment Settings

- Datasets

Dataset	#Curves	#Points	Anomaly Ratio
Yahoo	67	94,866	1.8%
AIOps18	58	5,922,913	2.26%
Industry	436	4,394,880	1.07%

- Evaluation Metrics

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN}, \quad F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Experimental Results

- Offline anomaly detection
  - ✓ 2.1%-54% improvement in Yahoo
  - ✓ 26%-86% improvement in AIOps18
  - ✓ 17%-70% improvement in Industry

Method	Yahoo			AIOps18			Industry		
	precision	**recall**	**F1 score**	**precision**	**recall**	**F1 score**	**precision**	**recall**	**F1 score**
LSTM	0.598	**0.706**	0.530	0.499	0.531	0.518	0.704	0.656	0.632
LSTM-VAE	0.622	0.634	0.484	0.510	0.625	0.537	0.717	0.639	0.622
Donut	0.530	0.658	0.524	0.405	0.527	0.382	0.693	0.628	0.604
LODA	**0.754**	0.583	0.428	0.553	0.429	0.401	0.583	0.498	0.529
iForest	0.713	0.597	0.437	0.555	0.439	0.413	0.616	0.567	0.538
DAGMM	0.643	0.517	0.401	0.590	0.477	0.461	0.597	0.542	0.530
SR-CNN	0.433	0.618	0.307	0.424	0.387	0.363	0.519	0.471	0.434
ADSketch	0.511	0.673	**0.541**	**0.744**	**0.670**	**0.677**	**0.811**	**0.813**	**0.740**

Experimental Results

- Online anomaly detection
  - ✓ 24%-65% improvement in AIOps18
  - ✓ 0.8%-48% improvement in Industry

Method	AIOps18			Industry		
	prec.	**rec.**	**F1**	**prec.**	**rec.**	**F1**
LSTM	0.425	0.462	0.408	0.612	0.606	0.592
LSTM-VAE	0.336	0.521	0.389	0.624	0.598	0.601
Donut	0.431	0.326	0.376	0.662	0.581	0.590
LODA	0.407	0.397	0.355	0.653	0.526	0.503
iForest	0.397	0.334	0.322	0.576	0.507	0.487
DAGMM	0.392	0.367	0.378	0.557	0.538	0.502
SR-CNN	0.329	0.288	0.307	0.438	0.422	0.410
ADSketch	**0.543**	**0.575**	**0.507**	**0.705**	**0.603**	**0.606**

Experimental Results

- Adaptive anomaly detection
  - ✓ 35%-42% improvement in AIOps18
  - ✓ 52%-83% improvement in Industry

Method	AIOps18			Industry		
	prec.	**rec.**	**F1**	**prec.**	**rec.**	**F1**
LODA	0.424	0.405	0.387	0.623	0.512	0.548
EVT	0.455	0.528	0.406	0.710	0.612	0.458
ADSketch	**0.594**	**0.557**	**0.548**	**0.882**	**0.856**	**0.832**

Industrial Deployment

ADSketch has been deployed in Huawei Cloud

- Serve tens of thousands of service instances and devices
- The accuracy of anomaly detection has been substantially improved
- Being integrated into the anomaly detection service for internal users

https://www.huaweicloud.com/lab/cnl/paper_anomaly_detection.html

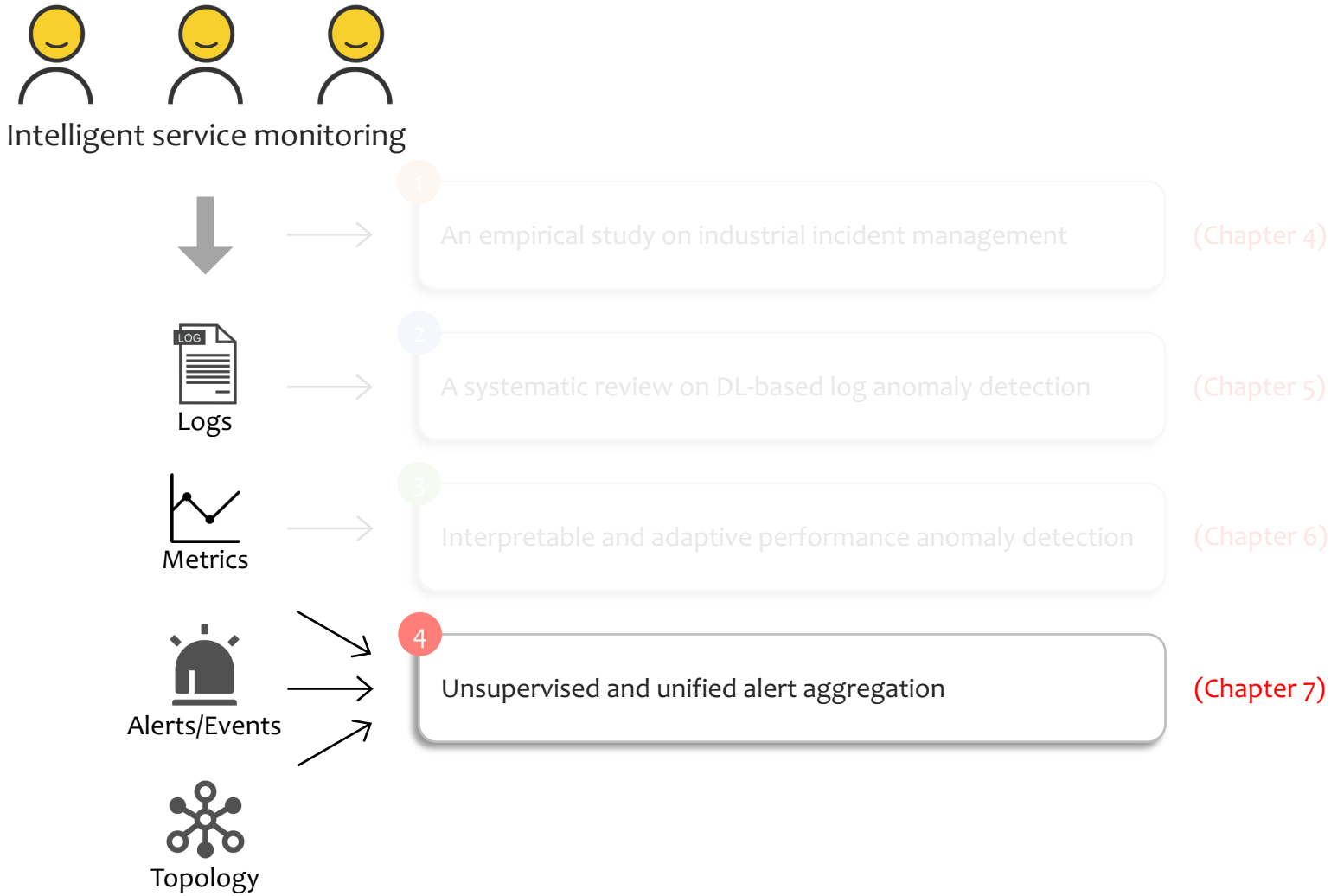
Content

- Topic 2: Interpretable and adaptive performance anomaly detection
  - ✓ Motivation
  - ✓ Anomaly detection based on pattern sketching
  - ✓ Evaluation
  - ✓ Summary

Summary of Topic 2

- ADSketch: A performance anomaly detector based on pattern sketching
  - ✓ An explicit metric pattern discovery algorithm
  - ✓ An adaptive pattern learning algorithm
  - ✓ A labeling scheme to improve interpretability and reuse human knowledge
- ADSketch has been deployed in production and performs well

Outline


Intelligent service monitoring
Logs
Metrics
Alerts/Events
Topology
An empirical study on industrial incident management
(Chapter 4)
A systematic review on DL-based log anomaly detection
(Chapter 5)
Interpretable and adaptive performance anomaly detection
(Chapter 6)
4
Unsupervised and unified alert aggregation
(Chapter 7)

Content

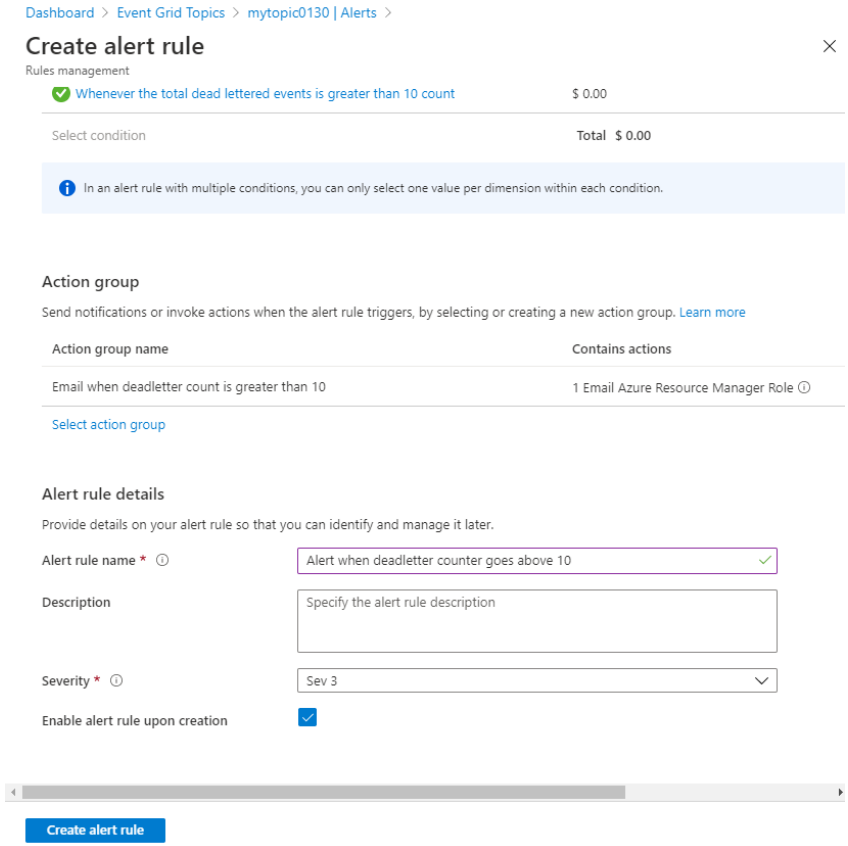
- Topic 3: Unsupervised and unified alert aggregation
  - ✓ Motivation
  - ✓ Graph representation learning for alert aggregation
  - ✓ Evaluation
  - ✓ Summary

Alerting in Online Services

- Alerting gives timely awareness to problems in cloud applications
- Monitors render an alert upon alerting policy violation
  - E.g., Specify the values of HTTP response latency that require user responses

Alert title: The HTTP response latency is higher than 2s for at least 5m.

Alert format	Alert ID, Alert type, Alert title, Alert time, Severity, Component, etc.



Setting alert rules in Microsoft Azure

Image from: https://learn.microsoft.com/en-us/azure/event-grid/set-alerts

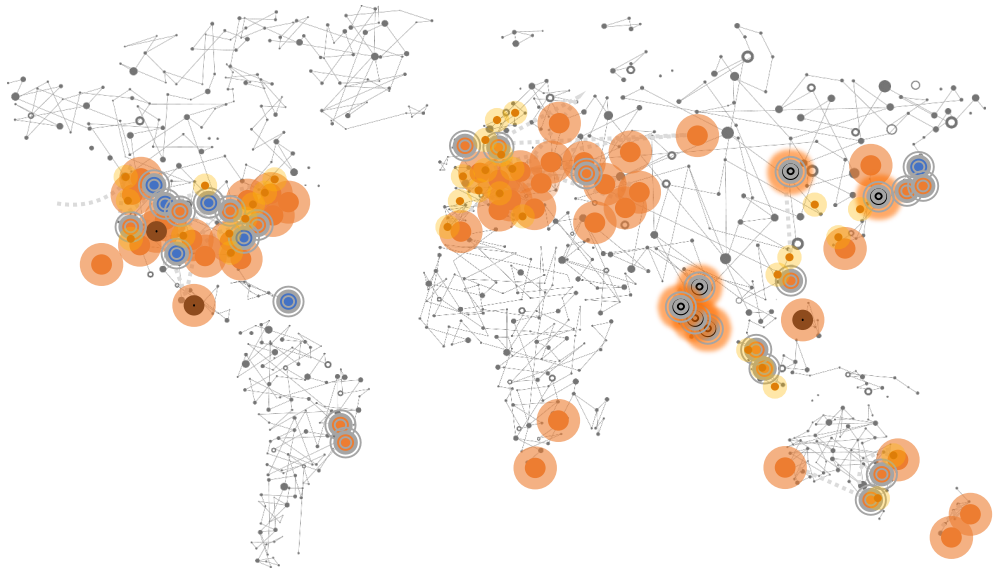
Flooding Alerts

Incidents often come with many alerts

- Complex service dependencies, i.e., cascading effect
- Conservative alerting policies

Pain points of site reliability engineers

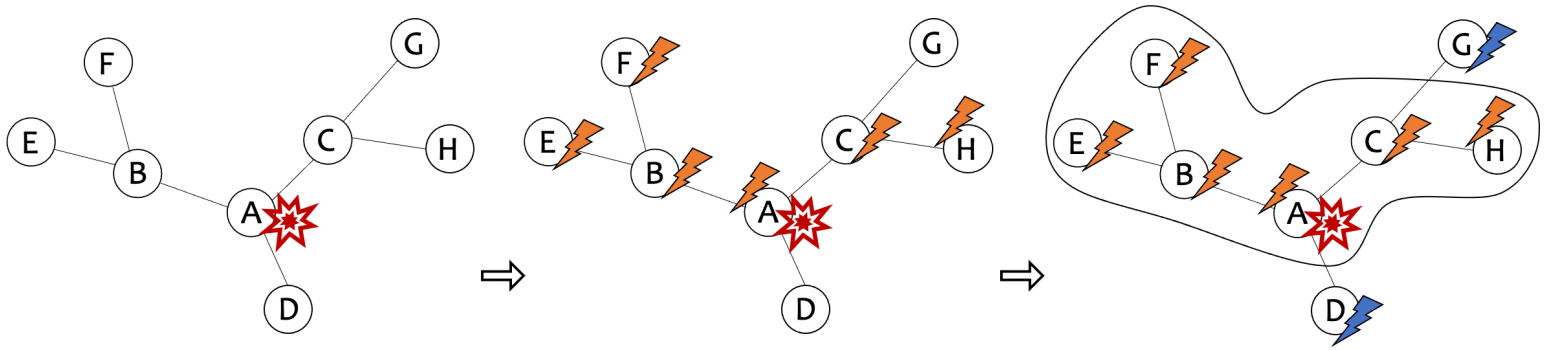
- Duplicate engineering efforts
- Delayed root cause analysis



Alert Aggregation

Group alerts associated with the same failure

- ✓ Estimate failure impact scope
- ✓ Save duplicate engineering effort



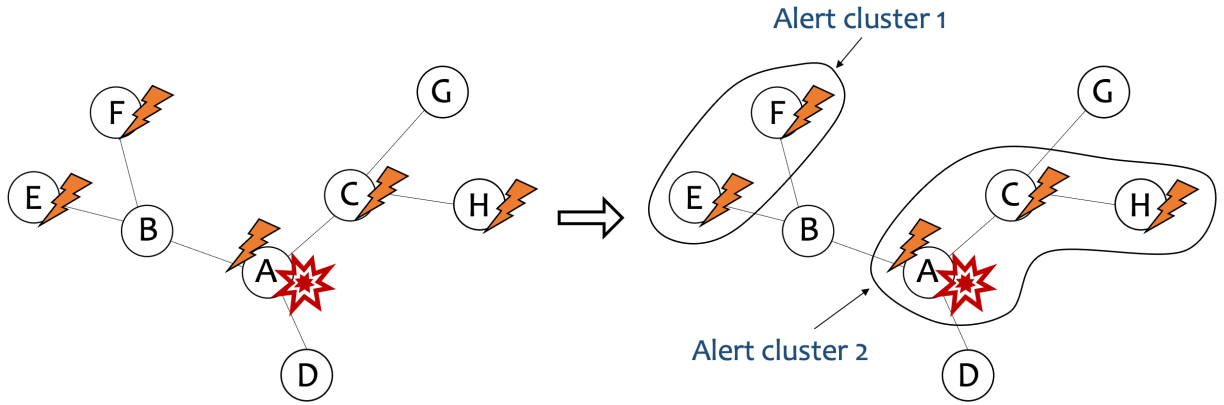
A failure happened to service A

Failure propagation

Failure-impact graph (the circled area)

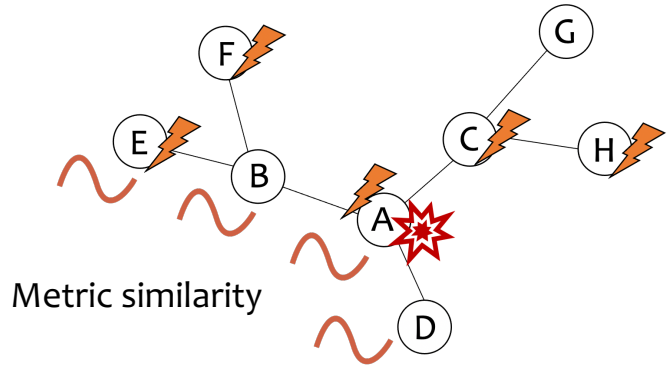
Challenges

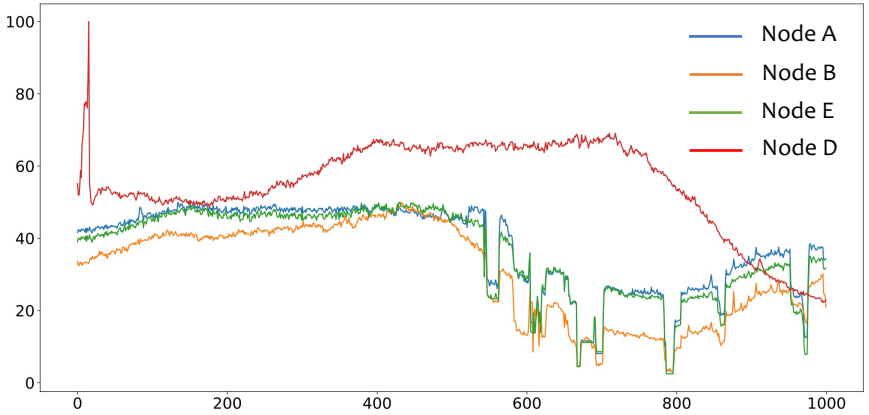
- Background noise
- Little textual similarity
  - ✓ “Traffic burst seen in Nginx node” and “Traffic burst seen in LVS node”
  - ✓ “Virtual machine is in abnormal state” and “OSPF protocol state change”
- Lack of labeled data
- Incomplete failure-impact graph based on alerts
  - ✓ Alerting policies not triggered
  - ✓ Fault tolerance bears anomalies

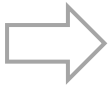


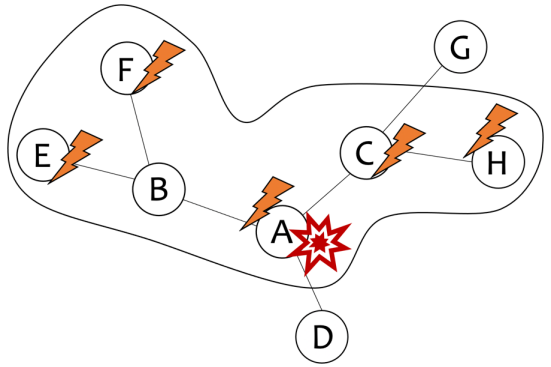
Incorporating Metric Information

- Metrics characterize failure impact in a more fine-grained way





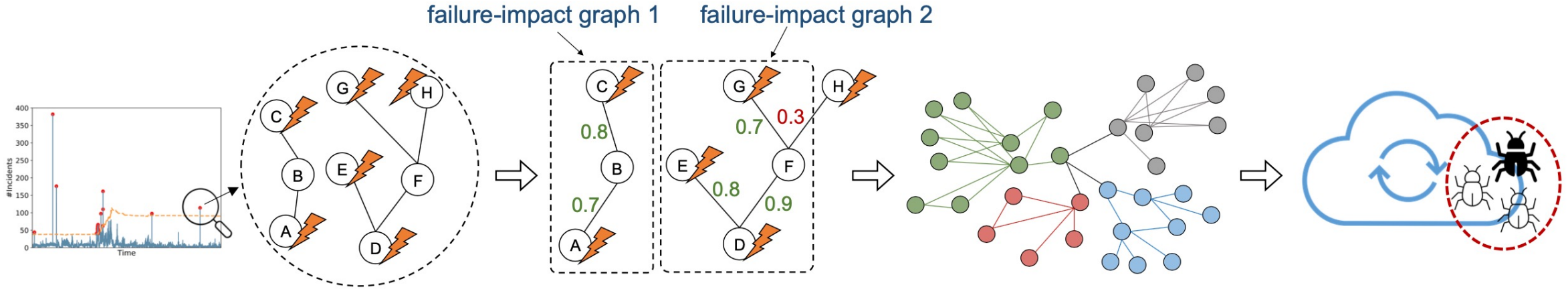




Content

- Topic 3: Unsupervised and unified alert aggregation
  - ✓ Motivation
  - ✓ Graph representation learning for alert aggregation
  - ✓ Evaluation
  - ✓ Summary

Girdle Overview



① Service failure detection

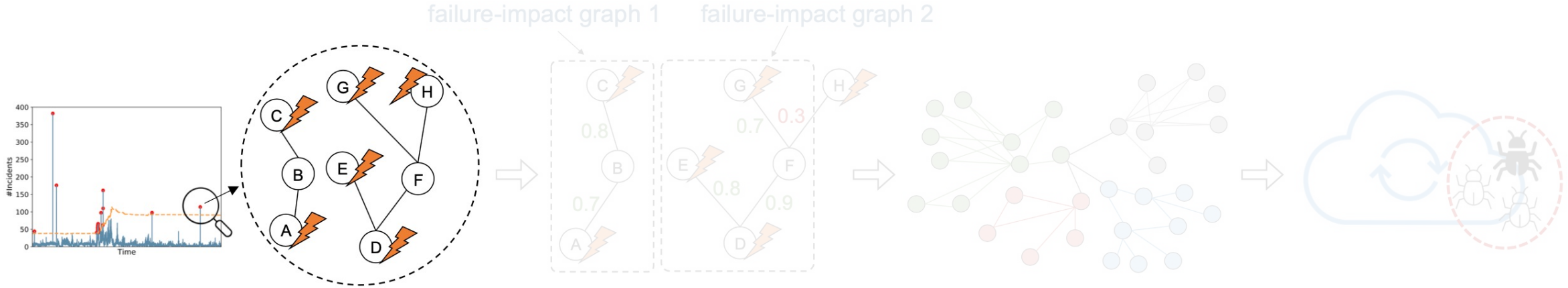
② Failure-impact graph completion

③ Graph representation learning

④ Online alert aggregation

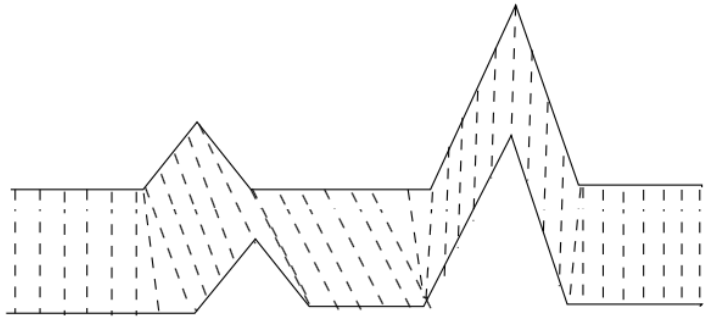
Service Failure Detection

- Detect historical failures for alert correlation learning
- Flooding alerts (check the no. of alerts/min)
- Extreme Value Theory (EVT)
  - ✓ No hand-set thresholds
  - ✓ No assumption on data distribution



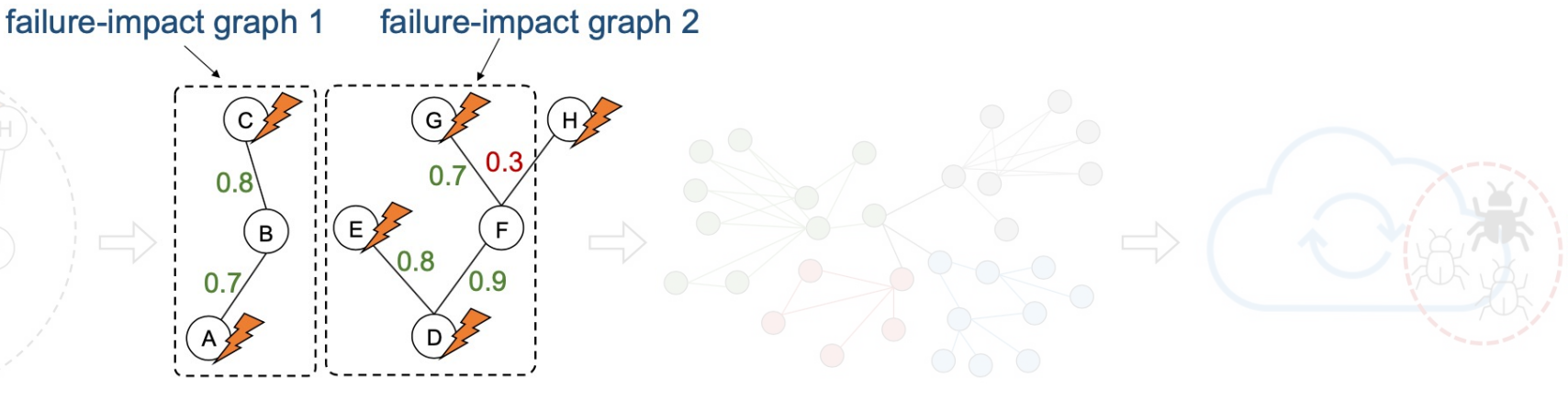
Failure-impact Graph Completion

- Identify alerts triggered by the common failure
- Community detection
  - ✓ Identify similar node sets in a graph
  - ✓ The key is the design of two nodes' similarity
    - ❖ Alert set similarity (Jaccard index)
    - ❖ Metric similarity (Dynamic time warping)
- Preliminary correlations between alerts



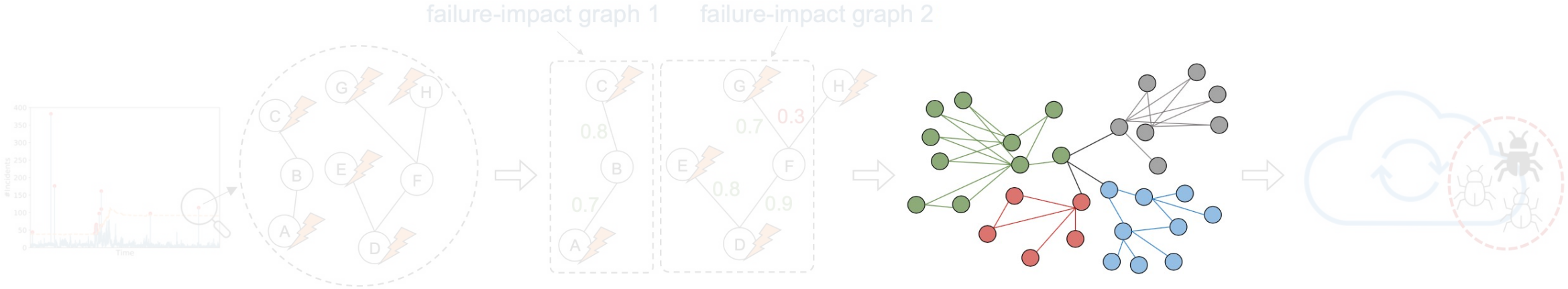
Dynamic time warping*

Deal with possible clock non-sync between nodes during metric collection



Graph Representation Learning

- Learn more significant correlations between alerts from historical failures
- Existing work combines different features by a simple weighted sum
- Graph representation learning
  - ✓ Learn a feature vector v for each unique type of alert
  - ✓ Unify the temporal and topological correlations of alerts



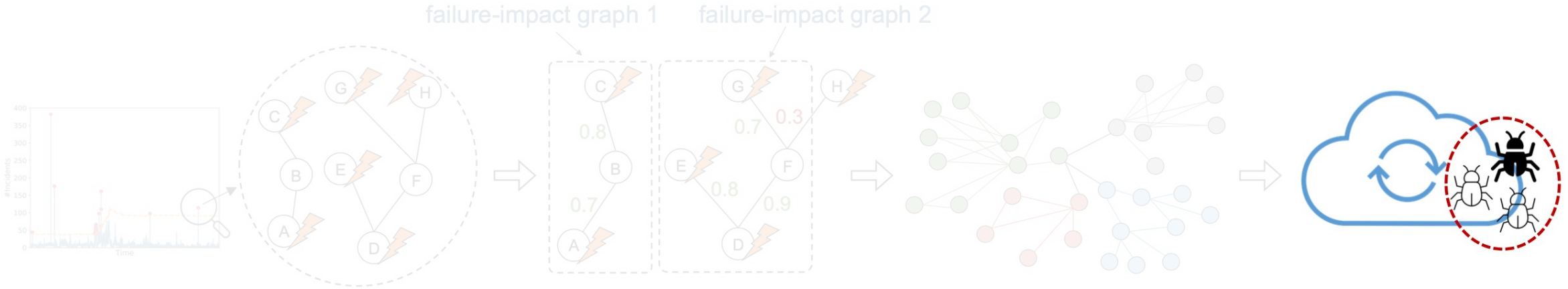
Online Alert Aggregation

- Quickly aggregate alerts when failures happen in production environment
- Two alerts i and j will be grouped if their similarity score is large

$$sim(i,j) = \mathcal{T}(i,j) \times \mathcal{H}(i,j)$$

Historical closeness
$\mathcal{H}(i,j) = \frac{v_i \cdot v_j}{\|\|v_i\|\| \times \|\|v_j\|\|}$

Topological rescaling
$\mathcal{T}(i,j) = \frac{1}{\max(1, dis(i,j) - \mathcal{D})}$



Content

- Topic 3: Unsupervised and unified alert aggregation
  - ✓ Motivation
  - ✓ Graph representation learning for alert aggregation
  - ✓ Evaluation
  - ✓ Summary

Dataset

- Alerts
  - ✓ Networking service of Huawei Cloud
  - ✓ Alerts are reported by various devices and virtual network function (VNF) instances

- Metrics
  - ✓ CPU usage
  - ✓ Round trip delay
  - ✓ Port in-bound/out-bound traffic rate
  - ✓ Package receiving/sending rate
  - ✓ Package receiving/sending error rate

Dataset	Training period	Testing period	#alerts	#failures
Dataset1	2020 May - July	2020 Aug.	~18k/~8k	105/46
Dataset2	2020 May - Aug.	2020 Sept.	~26k/~10k	151/52
Dataset3	2020 May - Sept.	2020 Oct.	~36k/~8k	203/38

Evaluation Metrics

- Service failure detection (binary classification)
  - ✓ Precision, Recall, and F1 score

$$Precision = \frac{TP}{TP + FP}, \qquad Recall = \frac{TP}{TP + FN}, \qquad F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

- Alert aggregation (clustering)
  - ✓ Normalized Mutual Information (NMI) in [0, 1] (the larger the better)

$$NMI(Y, C) = \frac{2 \times I(Y;\ C)}{H(Y) + H(C)}$$

$Y = class\ labels$
$C = cluster\ labels$
$H(\cdot) = Entropy$
$I(Y; C) = Mutual\ info\ b/w\ Y\ and\ C$

Service Failure Detection

- Girdle outperforms simple thresholding by 8.9%- 24.7%

Dataset	Metric	Thresholding	Girdle
Dataset1	Precision	0.711	**0.917**
	Recall	0.913	**0.957**
	F1 Score	0.799	**0.937**
Dataset2	Precision	0.831	**0.944**
	Recall	0.942	**0.981**
	F1 Score	0.883	**0.962**
Dataset3	Precision	0.648	**0.925**
	Recall	0.921	**0.974**
	F1 Score	0.761	**0.949**

Alert Aggregation

- Girdle achieves 10.4%-72.7% improvement
  - ✓ FP-Growth [1] is vulnerable to noise and unable to address rare yet important alerts
  - ✓ UHAS [2] does not learn from history
  - ✓ LiDAR [3] uses textual similarity which is not reliable

Method	Dataset1	Dataset2	Dataset3
FP-Growth	0.481	0.523	0.546
UHAS	0.697	0.71	0.707
LiDAR	0.742	0.758	0.826
GIRDLE	**0.831**	**0.866**	**0.912**

[1] Han et al. Mining frequent patterns without candidate generation. ACM SIGMOD Record '00.
[2] Zhao et al. Understanding and handling alert storm for online service systems. ICSE-SEIP '20.
[3] Chen et al. Identifying linked incidents in large-scale online service systems. ESEC/FSE '20.

Content

- Topic 3: Unsupervised and unified alert aggregation
  - ✓ Motivation
  - ✓ Graph representation learning for alert aggregation
  - ✓ Evaluation
  - ✓ Summary

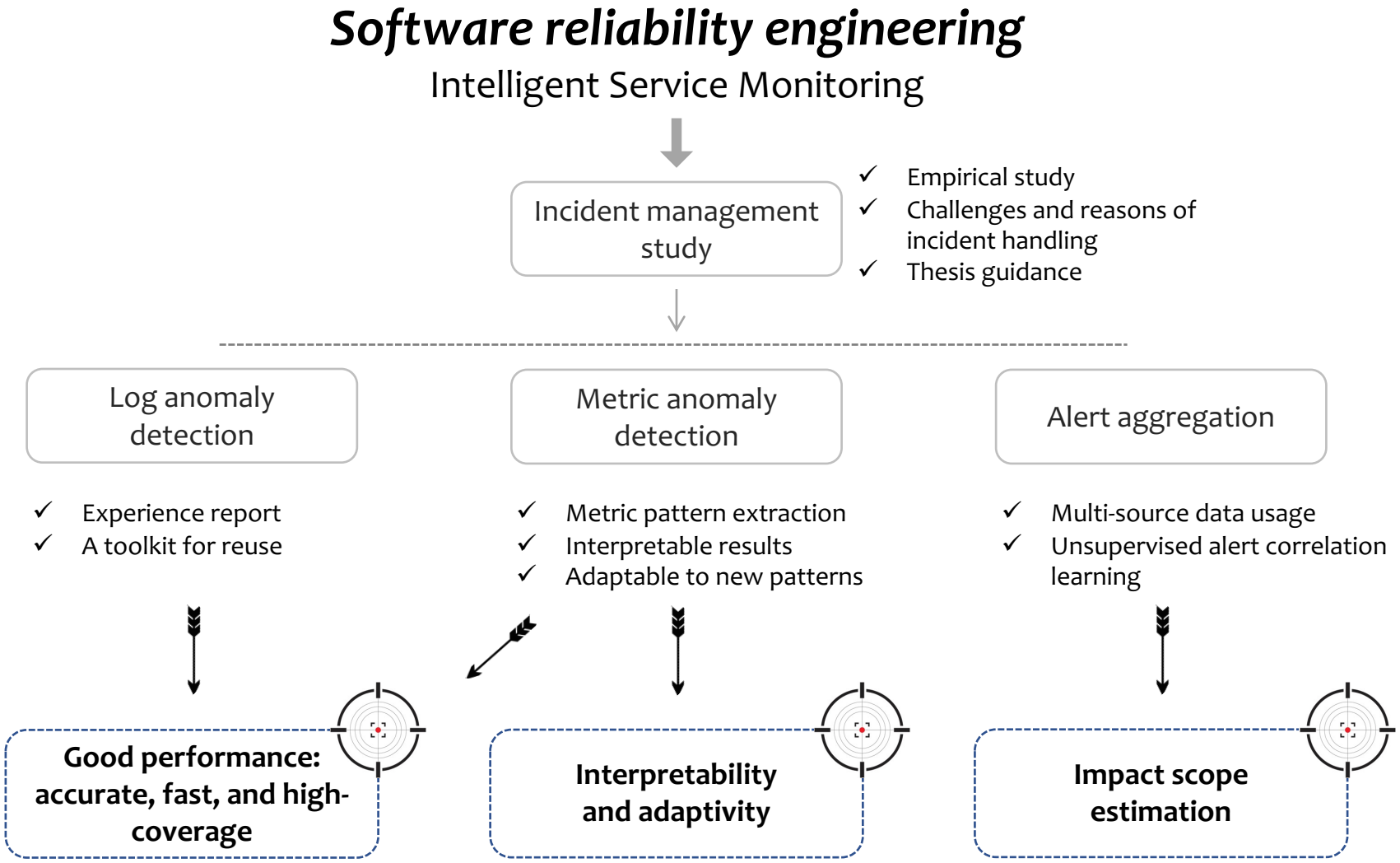
Summary of Topic 3

- Graph representation learning for alert aggregation
  - ✓ Incomplete cascading topology of failures
  - ✓ Learn alert correlation with multi-source information
- Girdle has been deployed in production and we received positive feedback

Outline

- Topic 1: An empirical study on industrial incident management
- Topic 2: Interpretable and adaptive performance anomaly detection
- Topic 3: Unsupervised and unified alert aggregation
- Conclusion and Future work

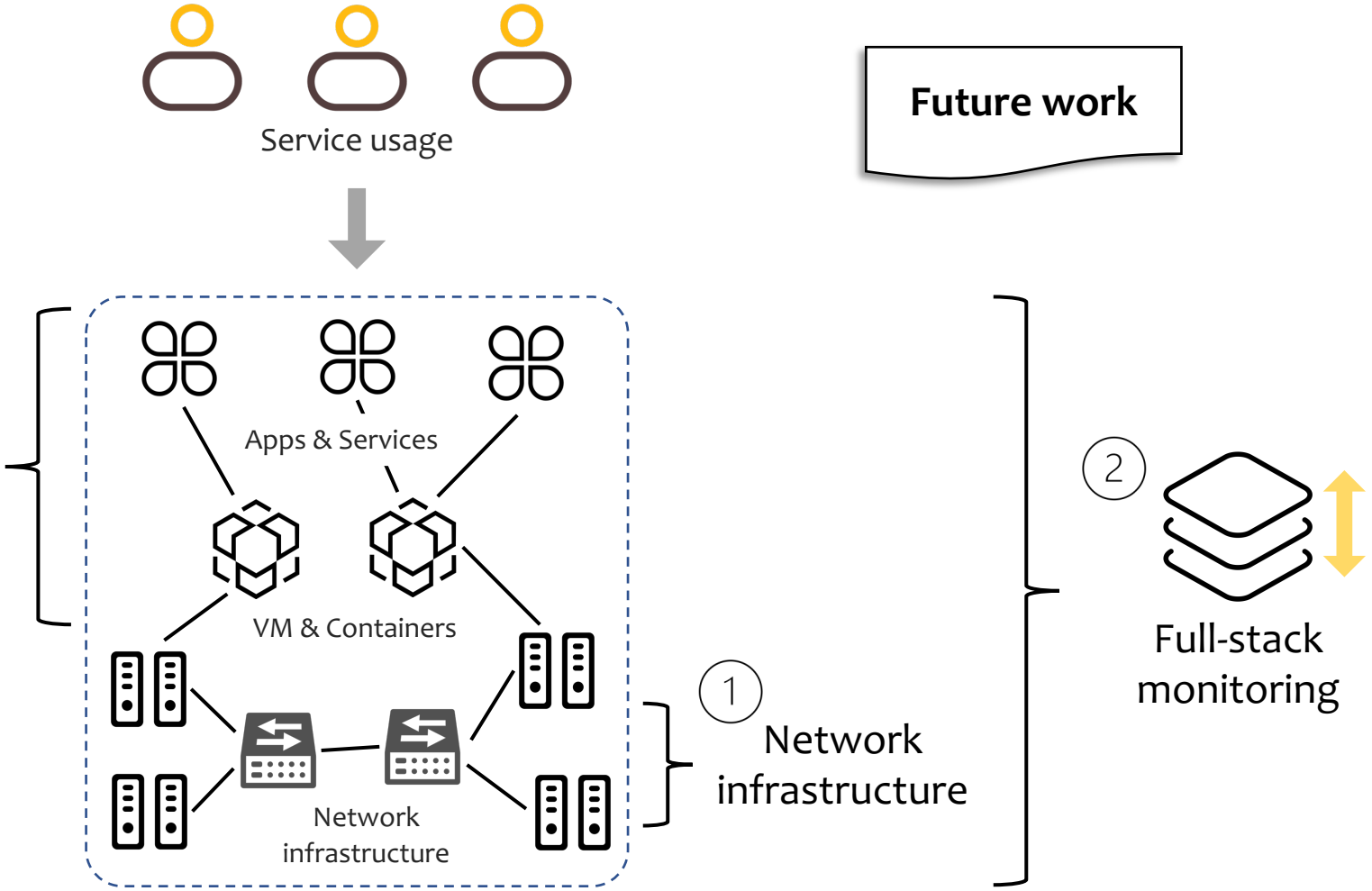
Conclusion


Software reliability engineering
Intelligent Service Monitoring
Incident management study
✓ Empirical study
✓ Challenges and reasons of incident handling
✓ Thesis guidance
Log anomaly detection
Metric anomaly detection
Alert aggregation
✓ Experience report
✓ A toolkit for reuse
✓ Metric pattern extraction
✓ Interpretable results
✓ Adaptable to new patterns
✓ Multi-source data usage
✓ Unsupervised alert correlation learning
Good performance: accurate, fast, and high-coverage
Interpretability and adaptivity
Impact scope estimation

Future Work

Current work

Software side of the cloud, i.e., SaaS and PaaS layers



Future work

Future Work (1)

Performance Monitoring and Diagnosis for Cloud Overlay Networks

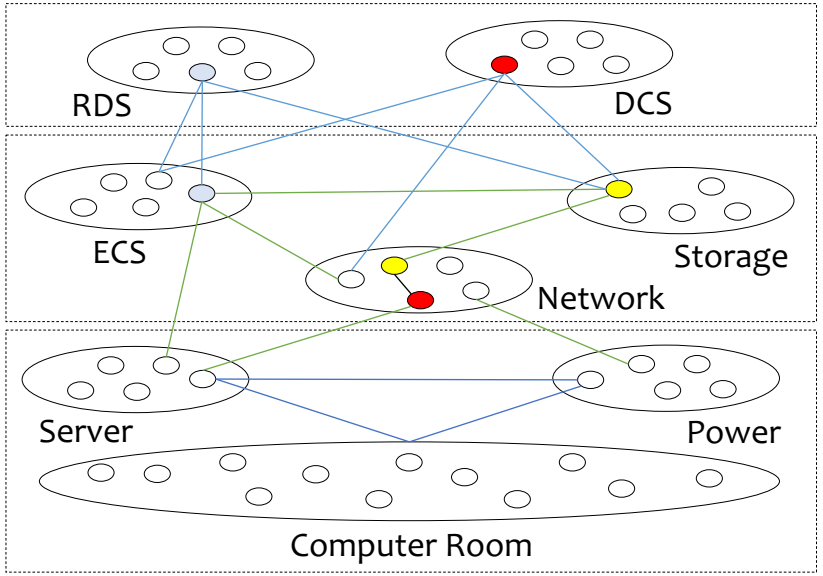
- Overlay networks are created by abstracting physical infrastructure
- Performance monitoring via probing
- Probing task design with the following two objectives
  - ✓ Minimum probing overhead
  - ✓ Fast diagnosis capability

Future Work (2)

Cross-layer Failure Propagation Modeling in Cloud Systems

- Existing work assumes isolated failures
  - ✓ Faults only exist in the service or layer under discussion, while others function normally
  - ✓ Not realistic in production systems

- Full-stack cloud monitoring
  - ✓ Trace problems at all cloud layers



Cross-layer failure propagation

Publications (1)

1. He, Pinjia, **Zhuangbin Chen**, Shilin He, and Michael R. Lyu. ``Characterizing the natural language descriptions in software logging statements.'' In 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 178-189. IEEE, 2018.
2. Bai, Haoli, **Zhuangbin Chen**, Michael R. Lyu, Irwin King, and Zenglin Xu. ``Neural relational topic models for scientific article analysis.'' In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 27-36. 2018.
3. Xu, Hui, **Zhuangbin Chen**, Weibin Wu, Zhi Jin, Sy-yen Kuo, and Michael Lyu. ``NV-DNN: towards fault-tolerant DNN systems with N-version programming'' In 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), pp. 44-47. IEEE, 2019.
4. **Zhuangbin Chen**, Yu Kang, Feng Gao, Li Yang, Jeffrey Sun, Zhangwei Xu, Pu Zhao et al. ``Aiops innovations of incident management for cloud services'' In AAAI Cloud Intelligence Workshop (2020).
5. **Zhuangbin Chen**, Yu Kang, Liqun Li, Xu Zhang, Hongyu Zhang, Hui Xu, Yangfan Zhou et al. ``Towards intelligent incident management: why we need it and how we make it'' In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 1487-1497. 2020.
6. He, Shilin, Pinjia He, **Zhuangbin Chen**, Tianyi Yang, Yuxin Su, and Michael R. Lyu. ``A survey on automated log analysis for reliability engineering'' ACM Computing Surveys (CSUR) 54, no. 6 (2021): 1-37.

Publications (2)

7. **Zhuangbin Chen**, Jinyang Liu, Wenwei Gu, Yuxin Su, and Michael R. Lyu. ``Experience report: deep learning-based system log analysis for anomaly detection'' arXiv preprint arXiv:2107.05908 (2021).
8. Xu, Hui, **Zhuangbin Chen**, Mingshen Sun, Yangfan Zhou, and Michael R. Lyu. ``Memory-Safety Challenge Considered Solved? An In-Depth Study with All Rust CVEs'' ACM Transactions on Software Engineering and Methodology (TOSEM) 31, no. 1 (2021): 1-25.
9. **Zhuangbin Chen**, Jinyang Liu, Yuxin Su, Hongyu Zhang, Xuemin Wen, Xiao Ling, Yongqiang Yang, and Michael R. Lyu. ``Graph-based Incident Aggregation for Large-Scale Online Service Systems'' In 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 430-442. IEEE, 2021.
10. **Zhuangbin Chen**, Jinyang Liu, Yuxin Su, Hongyu Zhang, Xiao Ling, Yongqiang Yang, and Michael R. Lyu. ``Adaptive performance anomaly detection for online service systems via pattern sketching'' In Proceedings of the 44th International Conference on Software Engineering (ICSE), pp. 61–72. 2022.
11. Li, Yichen, Xu Zhang, Shilin He, **Zhuangbin Chen**, Yu Kang, Jinyang Liu, Liqun Li et al. ``An Intelligent Framework for Timely, Accurate, and Comprehensive Cloud Incident Detection'' ACM SIGOPS Operating Systems Review, pp. 1-7. 2022.
12. Li, Baitong, Tianyi Yang, **Zhuangbin Chen**, Yuxin Su, Yongqiang Yang, and Michael R. Lyu. ``Heterogeneous Anomaly Detection for Software Systems via Attentive Multi-modal Learning'' arXiv preprint arXiv:2207.02918 (2022).

香港中文大學
The Chinese University of Hong Kong

ARISE
Automated Reliable Intelligent Software Engineering

Thank you!

Q & A