SparseRCA: Unsupervised Root Cause Analysis in Sparse Microservice Testing Traces

Zhenhe Yao[1], Haowei Ye, Changhua Pei[2], Guang Cheng, Guangpei Wang, Zhiwei Liu,

Hongwei Chen, Hang Cui, Zeyan Li, Jianhui Li, Gaogang Xie, Dan Pei

1. Presenter. Email: yaozh20@mails.tsinghua.edu.cn

2. Corresponding Author

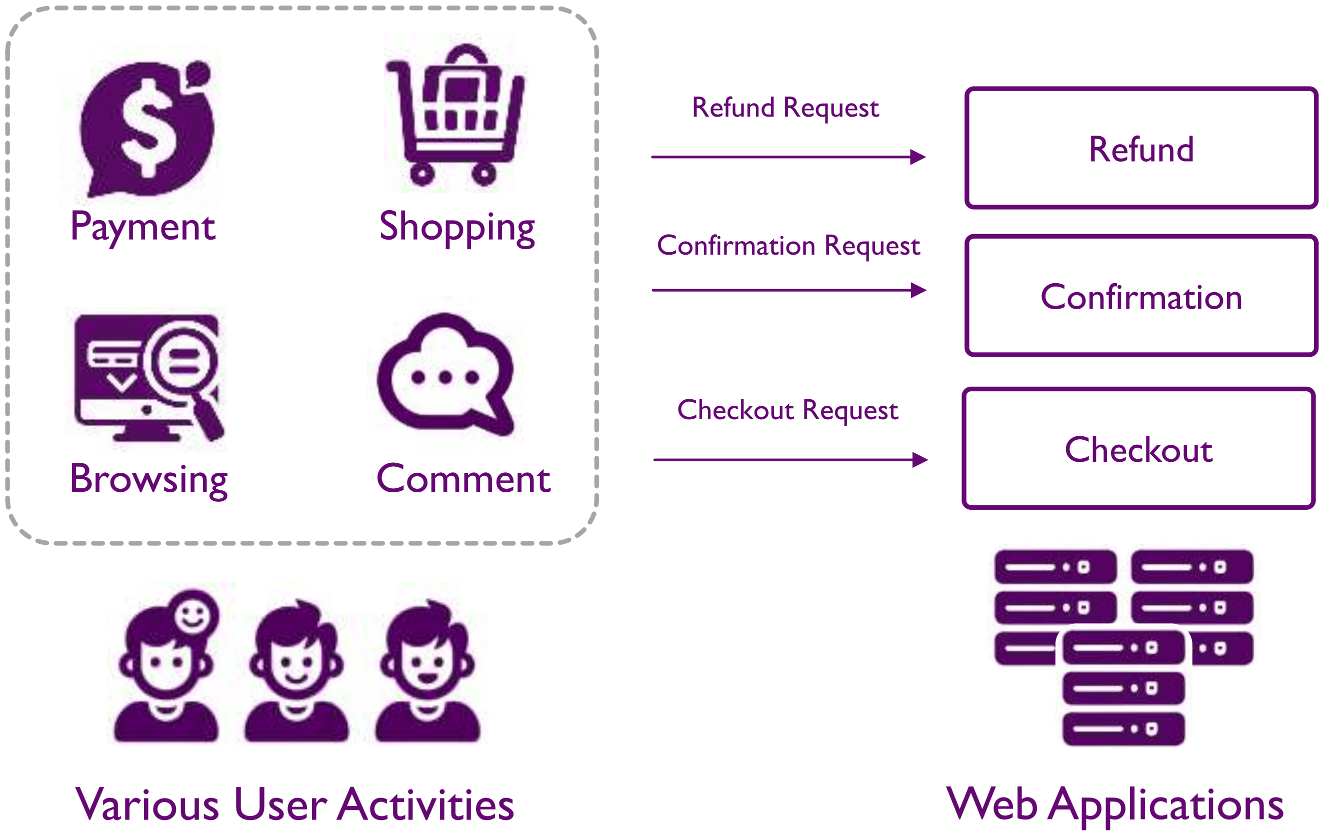
Outline

- Background
- Design
- Evaluation
- Summary

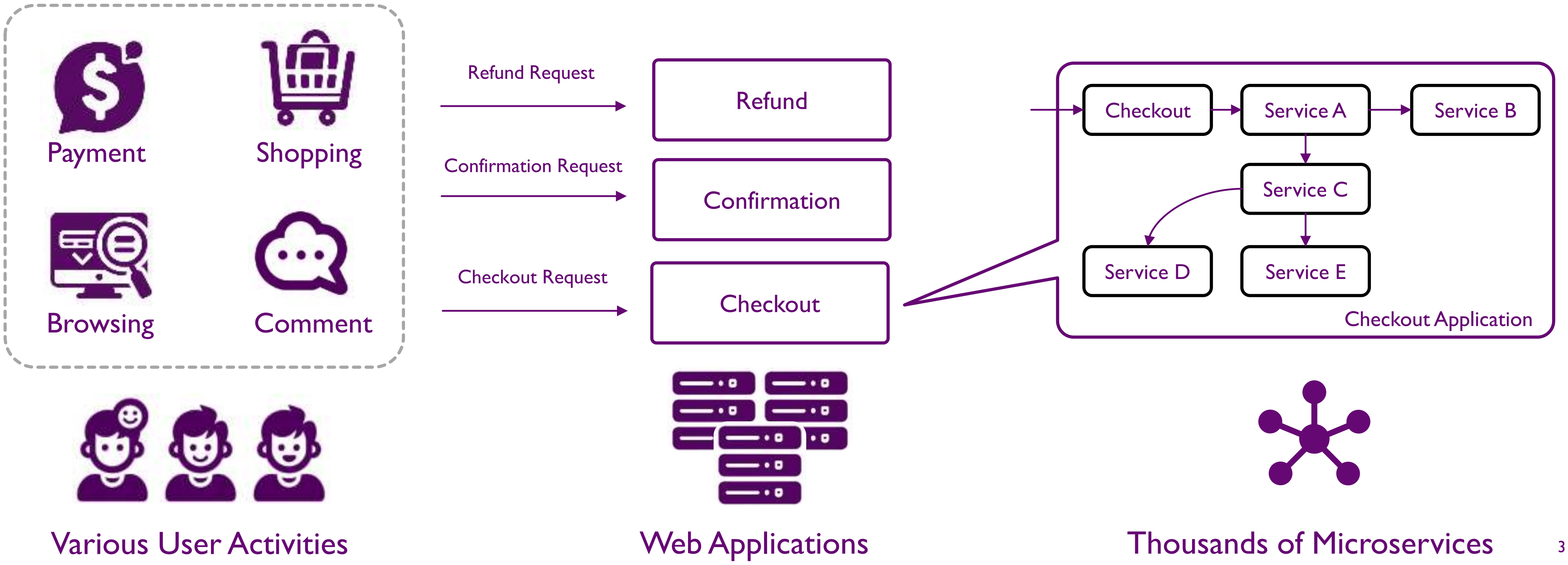
Microservice Architecture

Various User Activities

Microservice Architecture



Microservice Architecture

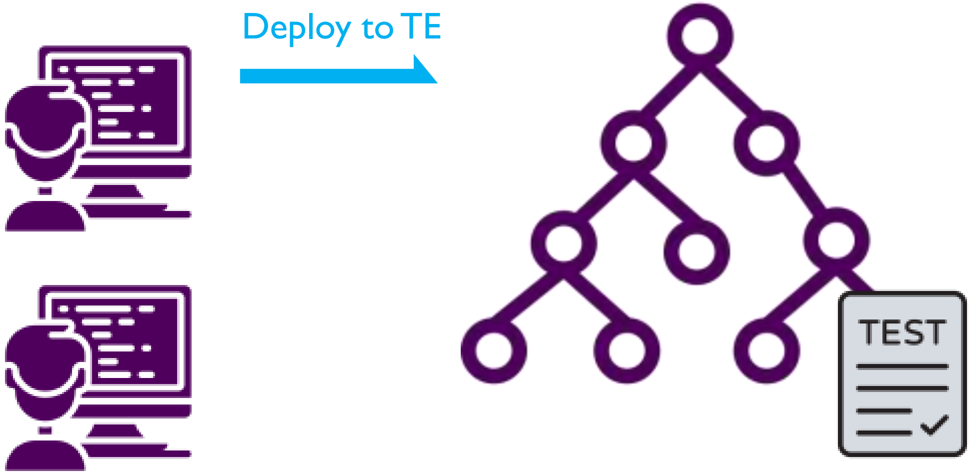


Ensuring Service-Level-Objectives (SLOs) in Testing Environments

Ensuring Service-Level-Objectives (SLOs) in Testing Environments

Developers

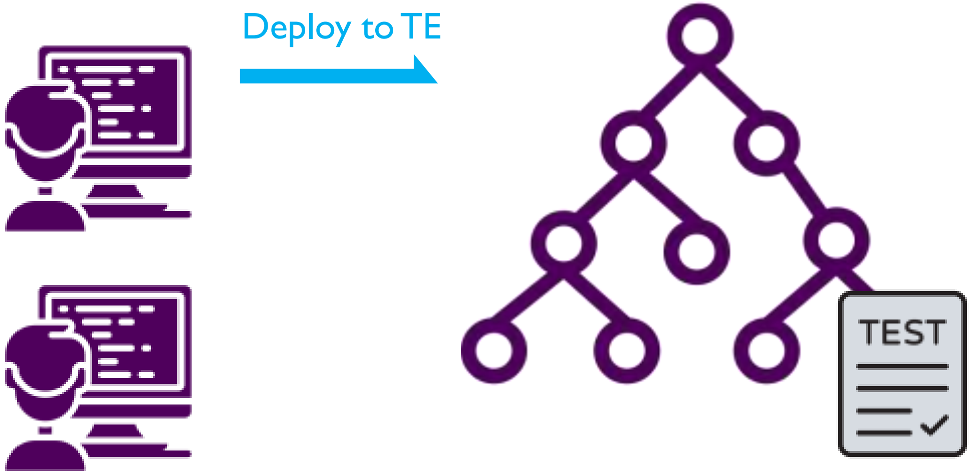
Ensuring Service-Level-Objectives (SLOs) in Testing Environments



Developers

Microservices in Testing Environment (TE)

Ensuring Service-Level-Objectives (SLOs) in Testing Environments



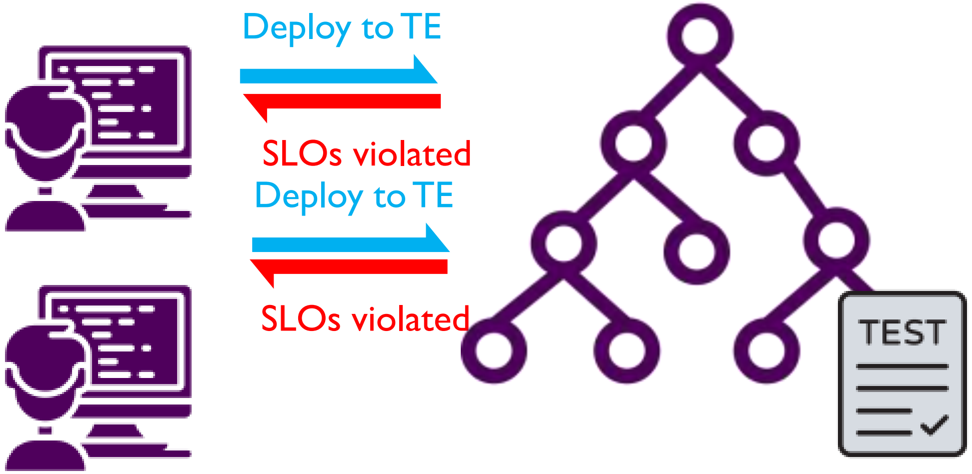
Developers

Microservices in Testing Environment (TE)

Service-Level-Objectives (SLOs):

- Normal latencies
- Correct response codes
- Non-abortions
- …

Ensuring Service-Level-Objectives (SLOs) in Testing Environments



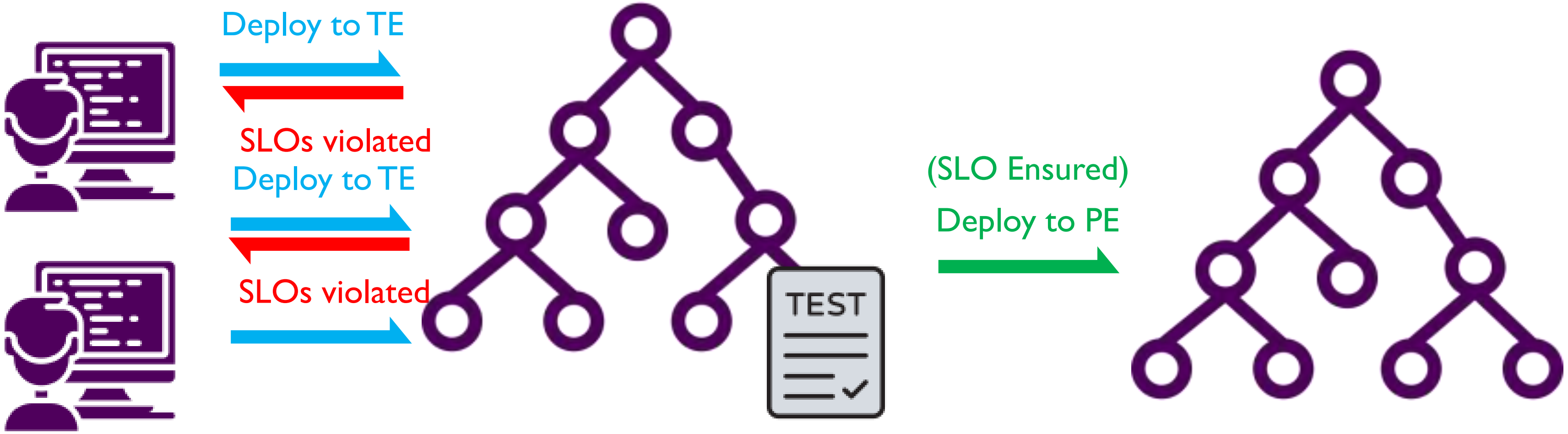
Developers

Microservices in Testing Environment (TE)

Service-Level-Objectives (SLOs):

- Normal latencies
- Correct response codes
- Non-abortions
- …

Ensuring Service-Level-Objectives (SLOs) in Testing Environments



Developers

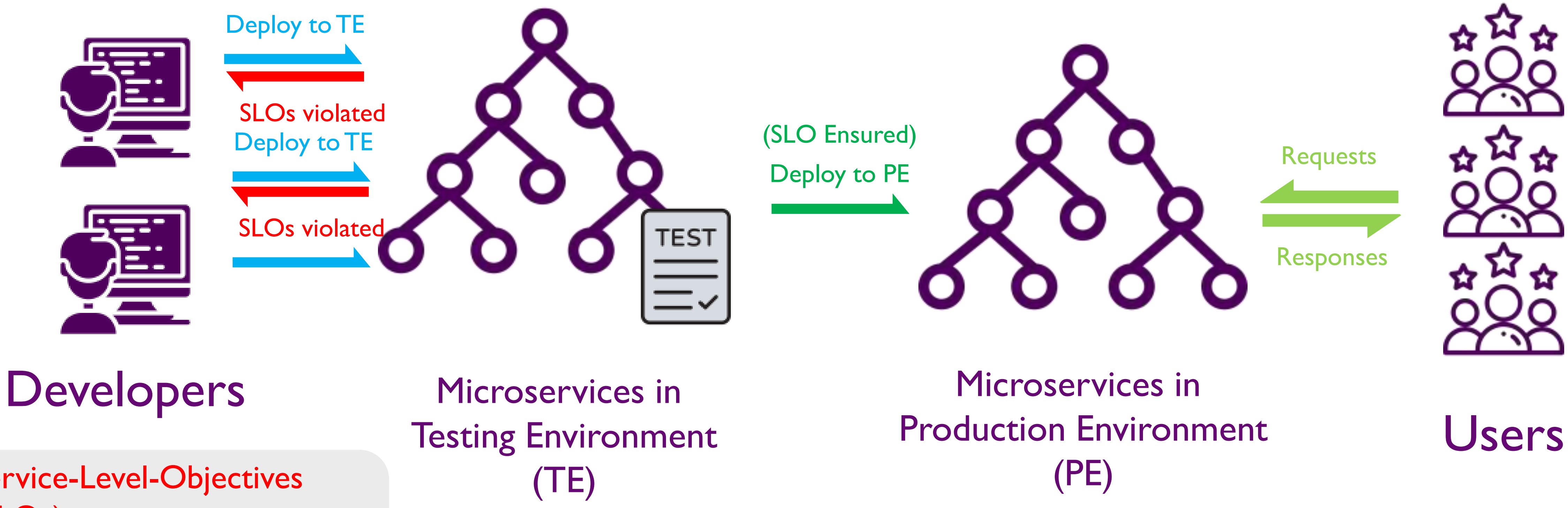
Microservices in Testing Environment (TE)

Microservices in Production Environment (PE)

Service-Level-Objectives (SLOs):

- Normal latencies
- Correct response codes
- Non-abortions
- …

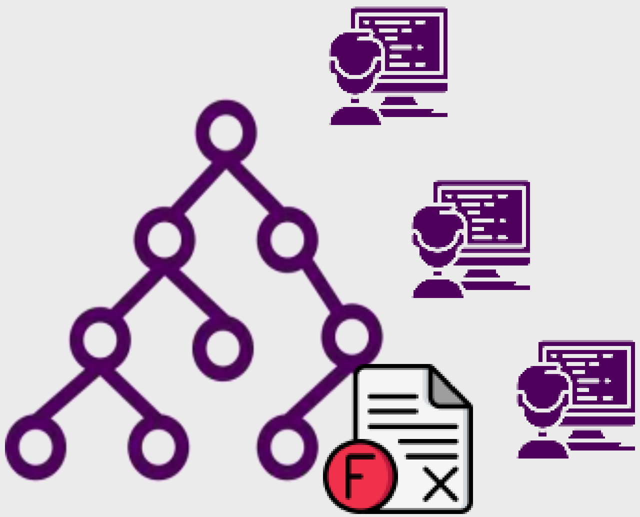
Ensuring Service-Level-Objectives (SLOs) in Testing Environments



Service-Level-Objectives (SLOs):

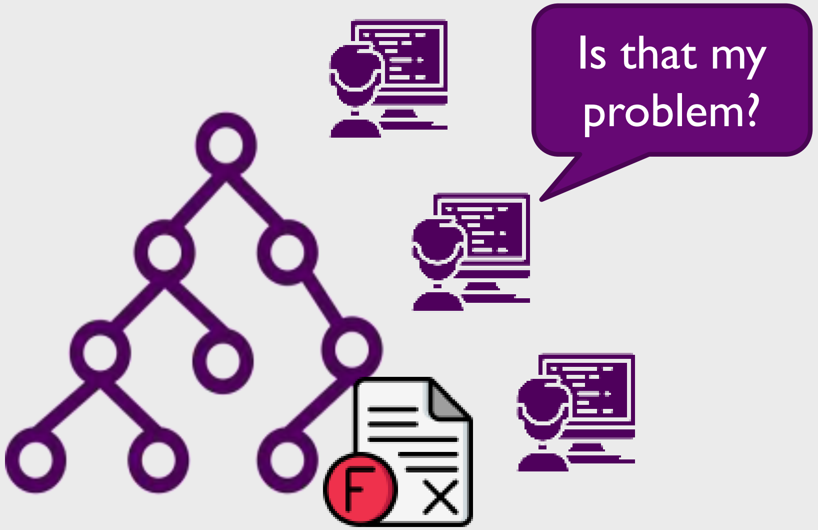
- Normal latencies
- Correct response codes
- Non-abortions
- …

RCA in Testing Environments



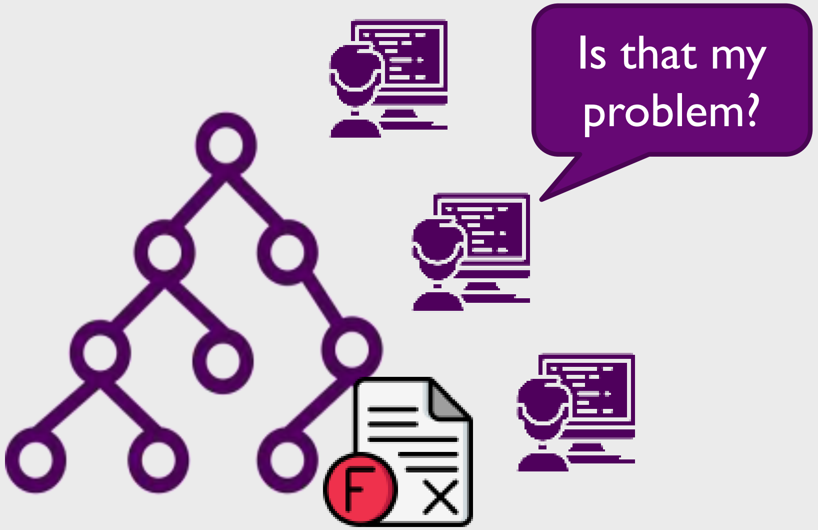
SLO-violated Microservice System in Testing Environment (TE)

RCA in Testing Environments

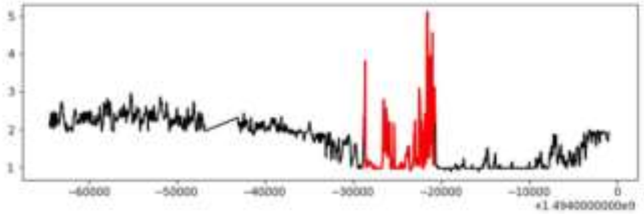


SLO-violated Microservice System in Testing Environment (TE)

RCA in Testing Environments



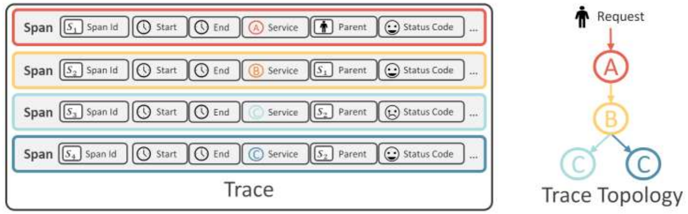
SLO-violated Microservice System in Testing Environment (TE)



Metric-based

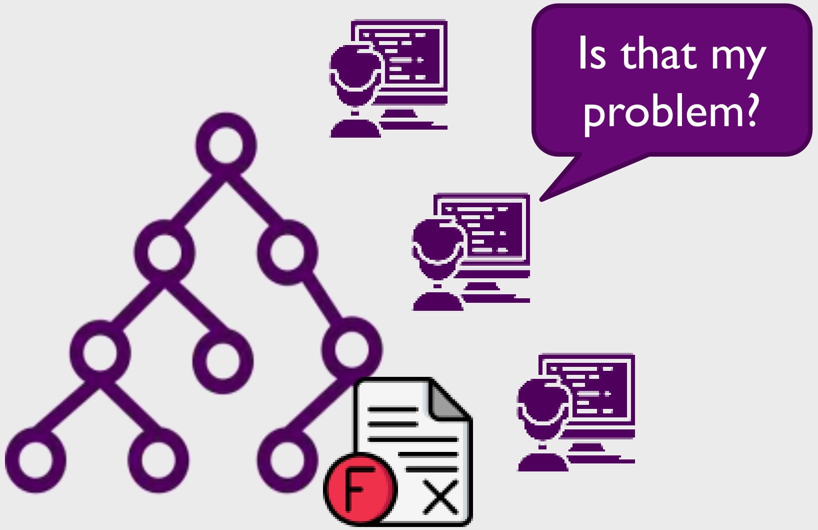
```
[00:00:01] [Info] checking if there are any updates...
[00:00:11] [Error] Connection Timeout.
[00:00:12] [Info] Time cost: 10.00s
[00:00:15] [Info] Time cost: 0.02s
```

Log-based

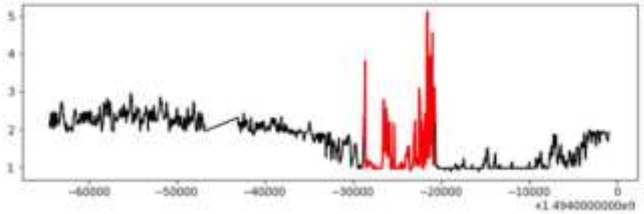


Trace-based

RCA in Testing Environments



SLO-violated Microservice System in Testing Environment (TE)

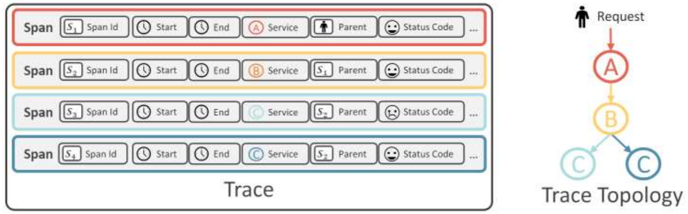


Metric-based

[00:00:01] [Info] checking if there are any updates...
[00:00:11] [Error] Connection Timeout.
[00:00:12] [Info] Time cost: 10.00s
[00:00:15] [Info] Time cost: 0.02s

Log-based

Only Information about a single service or a single calling relationship

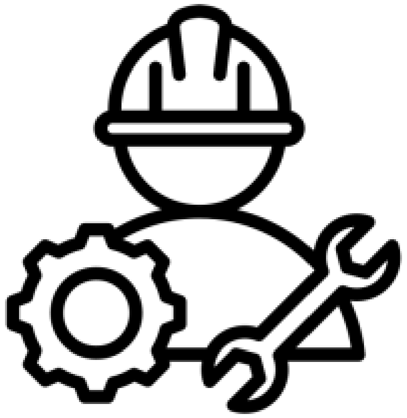


Trace-based

Record the complete calls and service interactions

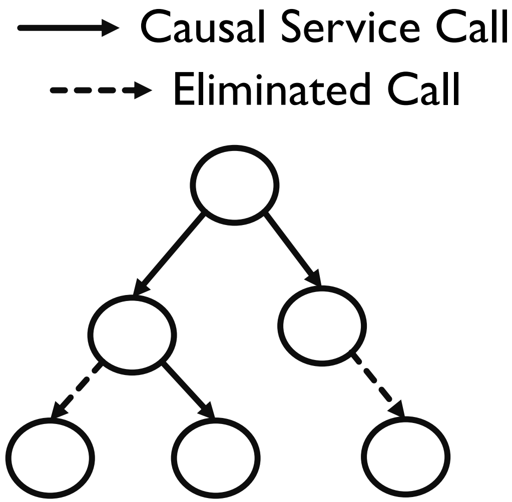
Previous Trace-based or Trace-related RCA Methods

Previous Trace-based or Trace-related RCA Methods



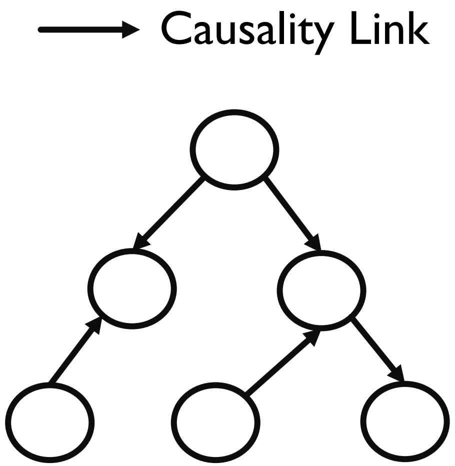
Rule-based

Groot(ASE'21)



Topology-based

MicroHECL(ICSE-SEIP'21)



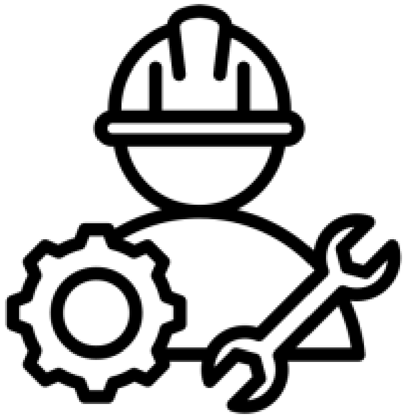
Causality-based

TraceRCA(IWQOS'21)

Spectrum-based

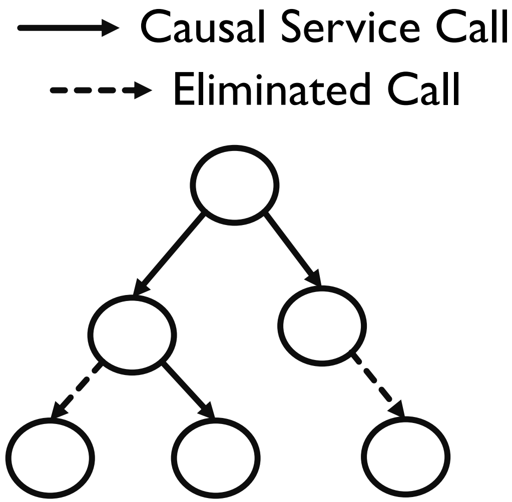
MicroRank(WWW'21)

Previous Trace-based or Trace-related RCA Methods



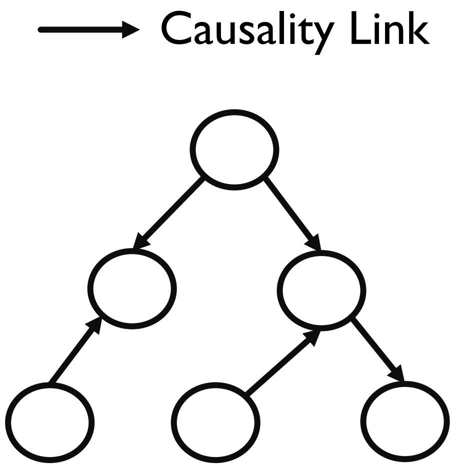
Rule-based

Groot(ASE'21)



Topology-based

MicroHECL(ICSE-SEIP'21)



Causality-based

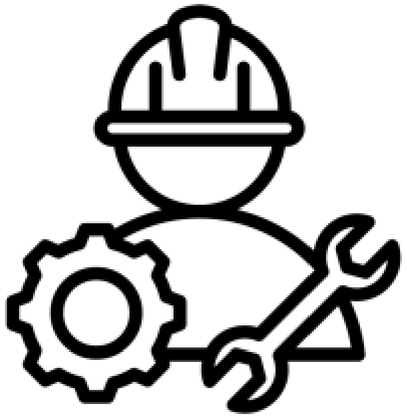
TraceRCA(IWQOS'21)

Spectrum-based

MicroRank(WWW'21)

- **Some are Dependent on continuous and dense trace data flow**
- **All need reliable and stable environment where rules/traces don't evolve too fast**

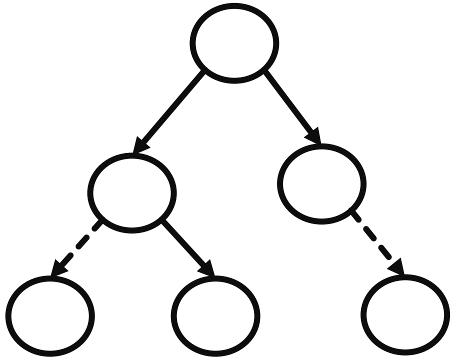
Previous Trace-based or Trace-related RCA Methods



Rule-based

Groot(ASE’21)

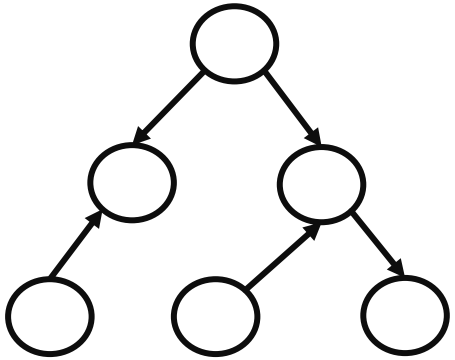
Causal Service Call
Eliminated Call



Topology-based

MicroHECL(ICSE-SEIP’21)

Causality Link



Causality-based

TraceRCA(IWQOS’21)

A Test Case

Spectrum-based

MicroRank(WWW’21)

- **Some are Dependent on continuous and dense trace data flow**
- **All need reliable and stable environment where rules/traces don’t evolve too fast**

Satisfied In Production Environment

Unsatisfied In Testing Environment

Problem 1: Trace Sparsity in Testing Environment

Production Environment

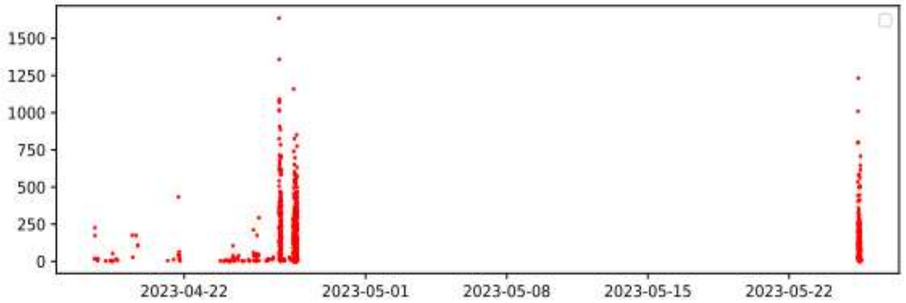
Traces from real user requests

- Dense traces (100k/min on average)
- Continuous data flow evenly distributed

Testing Environment

Traces from manually constructed test cases

- Sparse traces (mostly less than 10/min at peak)
- Centralized in testing period



Example of Trace Distribution in TE

#Trace/min	0	1	[2,10)	[10,30)	[30,50)	50+
Minute Pct. (%)	89.97	3.95	3.85	1.50	0.53	0.19
Abnormal Pct. (%)	/	11.11	8.01	6.16	7.12	2.04

Trace Number Distribution in TE

Problem 1: Trace Sparsity in Testing Environment

Production Environment

Traces from real user requests

- Dense traces (100k/min on average)
- Continuous data flow evenly distributed

Testing Environment

Traces from manually constructed test cases

- Sparse traces (mostly less than 10/min at peak)
- Centralized in testing period

Problem 1: Trace Sparsity in Testing Environment

Production Environment

Traces from real user requests

- Dense traces (100k/min on average)
- Continuous data flow evenly distributed

Testing Environment

Traces from manually constructed test cases

- Sparse traces (mostly less than 10/min at peak)
- Centralized in testing period

Unstable aggregated metrics

- Average latencies
- Response success rates

…

Problem 1: Trace Sparsity in Testing Environment

Production Environment

Traces from real user requests

- Dense traces (100k/min on average)
- Continuous data flow evenly distributed

Testing Environment

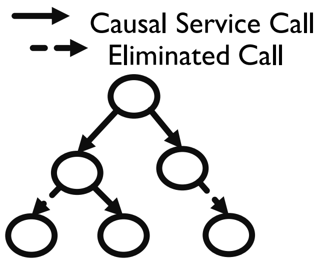
Traces from manually constructed test cases

- Sparse traces (mostly less than 10/min at peak)
- Centralized in testing period

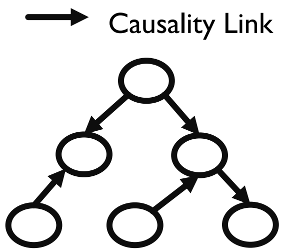
Unstable aggregated metrics

- Average latencies
- Response success rates

…



Topology-based



Causality-based

Spectrum-based

Problem 1: Trace Sparsity in Testing Environment

Production Environment

Traces from real user requests

- Dense traces (100k/min on average)
- Continuous data flow evenly distributed

Testing Environment

Traces from manually constructed test cases

- Sparse traces (mostly less than 10/min at peak)
- Centralized in testing period

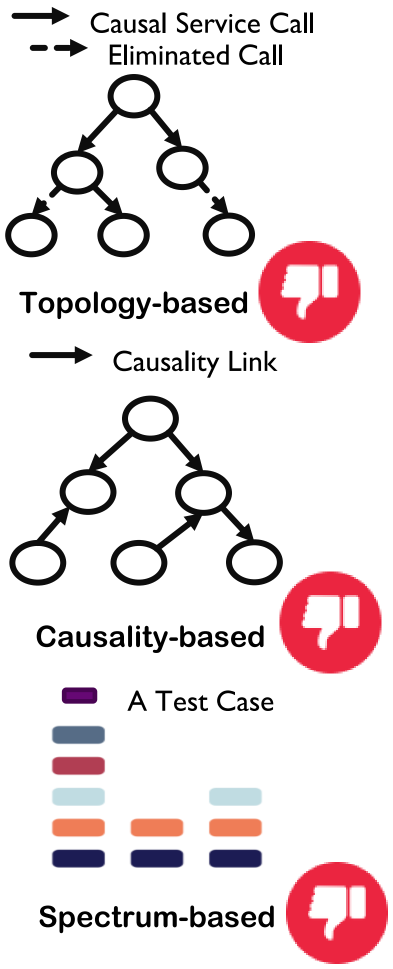
Unstable aggregated metrics

- Average latencies
- Response success rates

…

Unreliable mining results

- Correlation
- Path Elimination
- Causality Discovery
- Spectrum Analysis
- …



Problem 2: System Volatility in Testing Environment

Production Environment

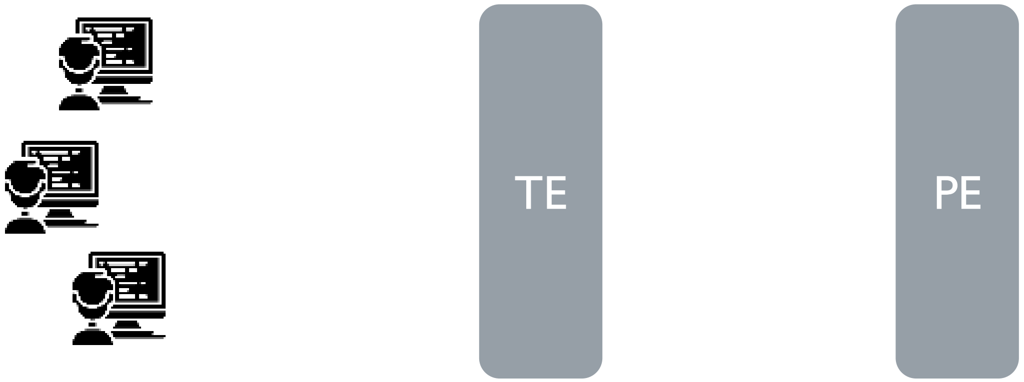
The environment is stable and less updated

- Experience from history lasts longer
- Microservices and calls can be found in the past

Testing Environment

The environment is frequently updated

- Knowledge expire quickly in frequent updates
- New microservices are deployed frequently
- New microservice calls emerge frequently



Problem 2: System Volatility in Testing Environment

Production Environment

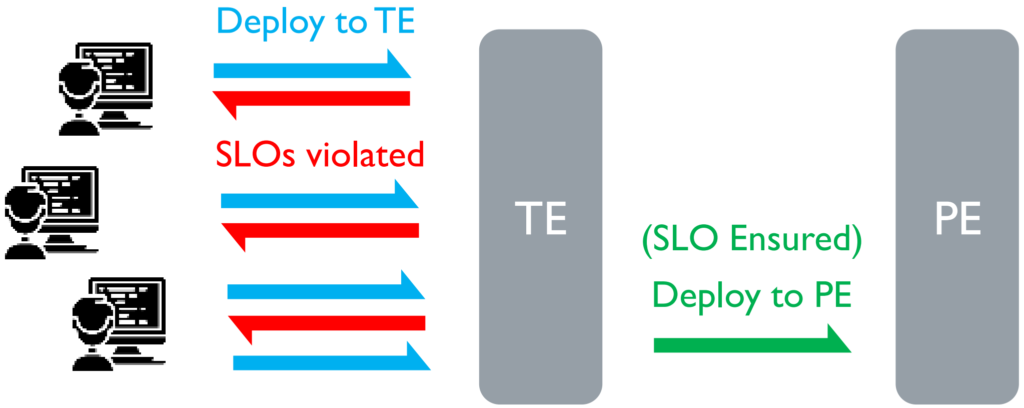
The environment is stable and less updated

- Experience from history lasts longer
- Microservices and calls can be found in the past

Testing Environment

The environment is frequently updated

- Knowledge expire quickly in frequent updates
- New microservices are deployed frequently
- New microservice calls emerge frequently



TE experiences more frequent updates than PE

Problem 2: System Volatility in Testing Environment

Production Environment

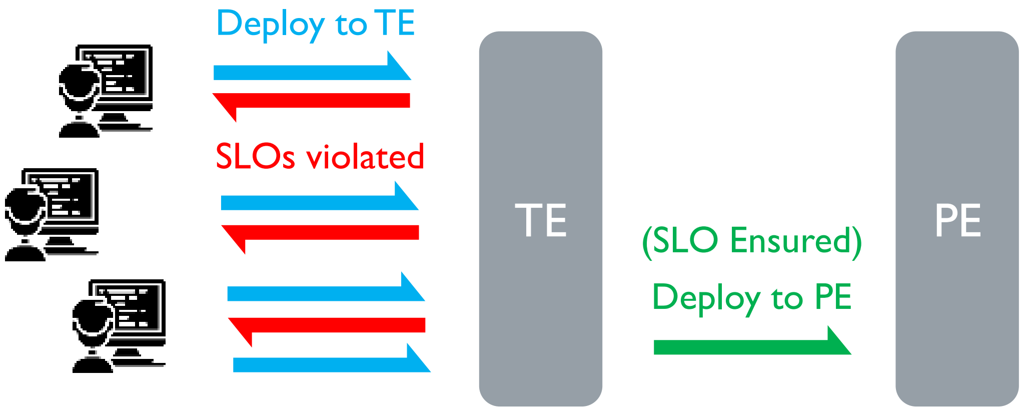
The environment is stable and less updated

- Experience from history lasts longer
- Microservices and calls can be found in the past

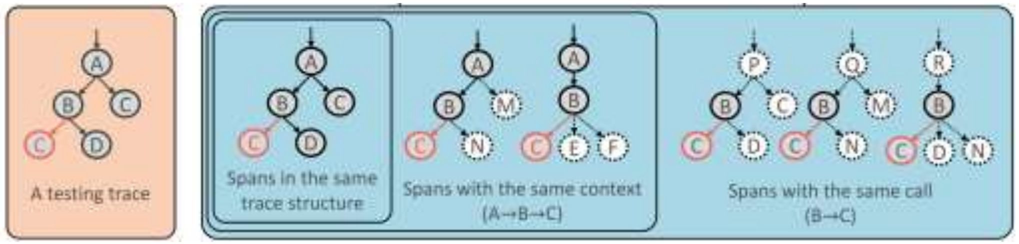
Testing Environment

The environment is frequently updated

- Knowledge expire quickly in frequent updates
- New microservices are deployed frequently
- New microservice calls emerge frequently



TE experiences more frequent updates than PE



New microservices and calls frequently emerge in TE

Problem 2: System Volatility in Testing Environment

Production Environment

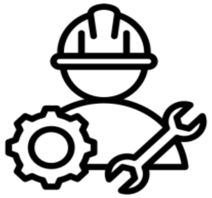
The environment is stable and less updated

- Experience from history lasts longer
- Microservices and calls can be found in the past

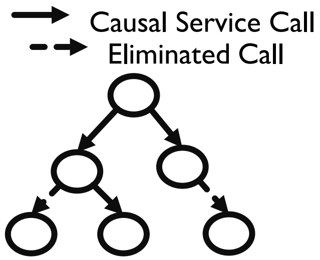
Testing Environment

The environment is frequently updated

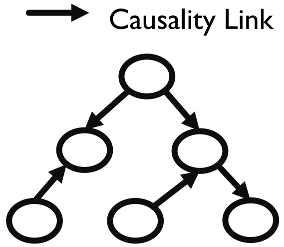
- Knowledge expire quickly in frequent updates
- New microservices are deployed frequently
- New microservice calls emerge frequently



Rule-based



Topology-based



Causality-based

Problem 2: System Volatility in Testing Environment

Production Environment

The environment is stable and less updated

- Experience from history lasts longer
- Microservices and calls can be found in the past

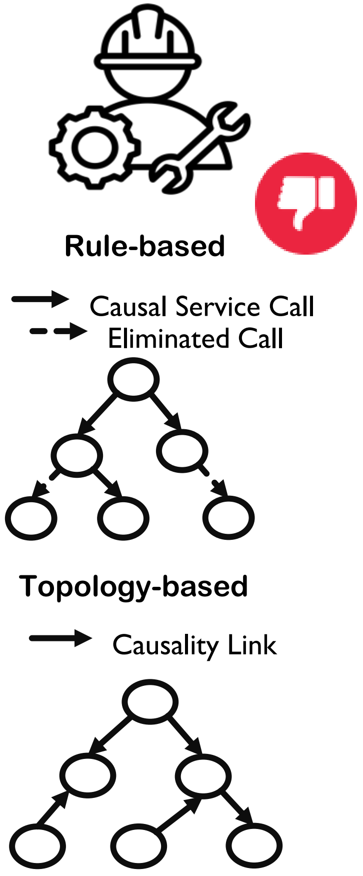
Testing Environment

The environment is frequently updated

- Knowledge expire quickly in frequent updates
- New microservices are deployed frequently
- New microservice calls emerge frequently

Manual Configuration for RCA in TE is Impractical

- Labor-intensive
- Knowledge expires soon



Causality-based

Problem 2: System Volatility in Testing Environment

Production Environment

The environment is stable and less updated

- Experience from history lasts longer
- Microservices and calls can be found in the past

Testing Environment

The environment is frequently updated

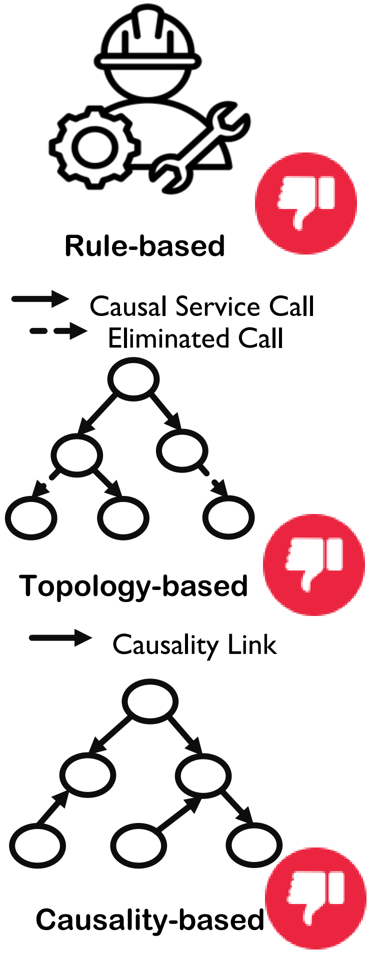
- Knowledge expire quickly in frequent updates
- New microservices are deployed frequently
- New microservice calls emerge frequently

Manual Configuration for RCA in TE is Impractical

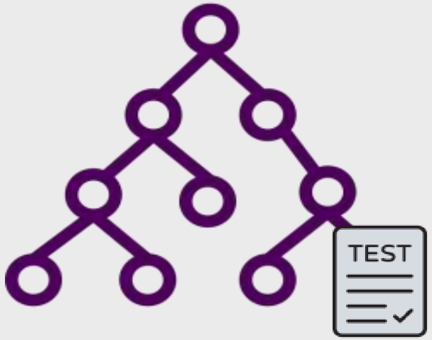
- Labor-intensive
- Knowledge expires soon

New microservices and calls unseen in history frequently emerge

- Topology changes
- Causality needs reestablishing

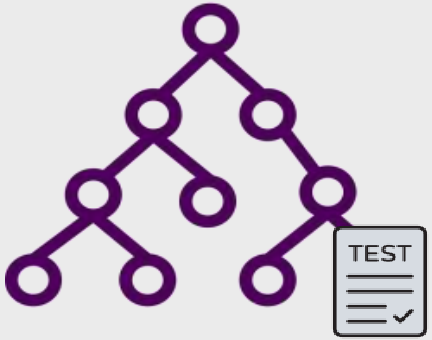


Challenges for Trace-based RCA in TE



Trace-based RCA in Testing Environment

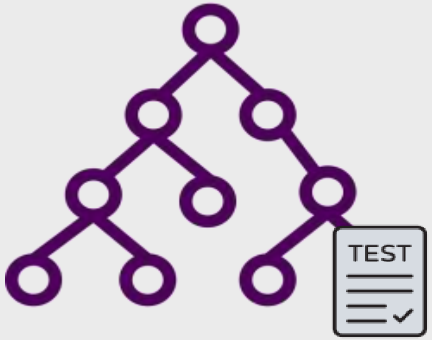
Challenges for Trace-based RCA in TE



Trace-based RCA in Testing Environment

- Trace Sparsity
- System Volatility

Challenges for Trace-based RCA in TE



Trace-based RCA in Testing Environment

Trace Sparsity

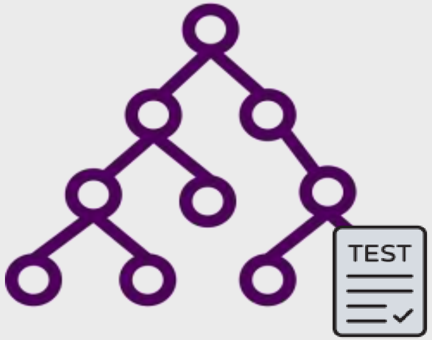
System Volatility

C1: The infeasibility of human intervention for RCA in TE

C2: The limitations introduced by trace sparsity

C3: Unseen microservices and calls in TE traces

Challenges for Trace-based RCA in TE



Trace-based RCA in Testing Environment

Trace Sparsity

System Volatility

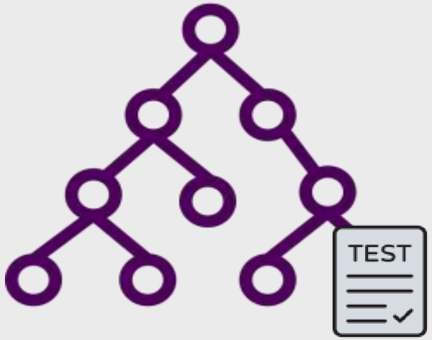
C1: The infeasibility of human intervention for RCA in TE

C2: The limitations introduced by trace sparsity

C3: Unseen microservices and calls in TE traces

Unsupervised Trace-based RCA trained with the same batch of traces

Challenges for Trace-based RCA in TE



Trace-based RCA in Testing Environment

- Trace Sparsity
- System Volatility

C1: The infeasibility of human intervention for RCA in TE

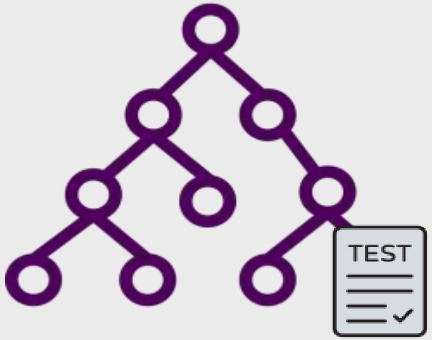
Unsupervised Trace-based RCA trained with the same batch of traces

C2: The limitations introduced by trace sparsity

Performing single-trace RCA inference

C3: Unseen microservices and calls in TE traces

Challenges for Trace-based RCA in TE



Trace-based RCA in Testing Environment

- Trace Sparsity
- System Volatility

C1: The infeasibility of human intervention for RCA in TE

Unsupervised Trace-based RCA trained with the same batch of traces

C2: The limitations introduced by trace sparsity

Performing single-trace RCA inference

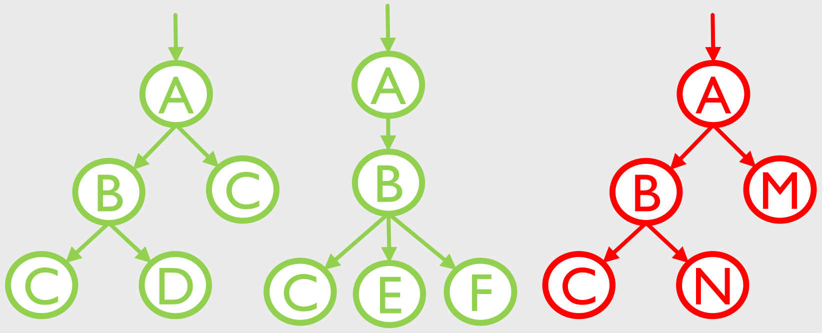
C3: Unseen microservices and calls in TE traces

Modeling Unseen Microservices and Calls

Problem Formulation

Training Input:

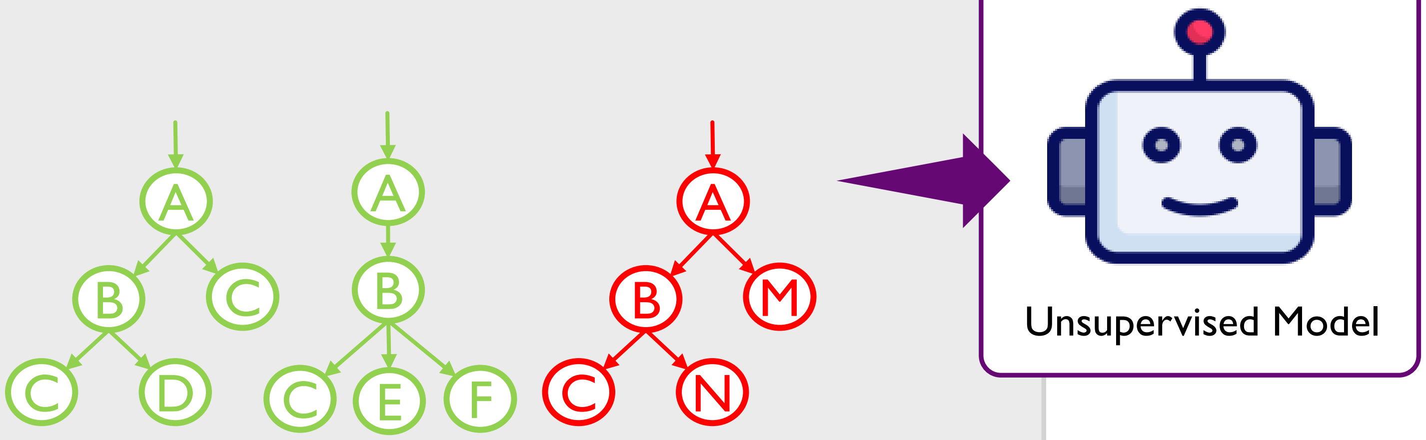
All traces satisfying and violating SLOs in the same test round



Problem Formulation

Training Input:

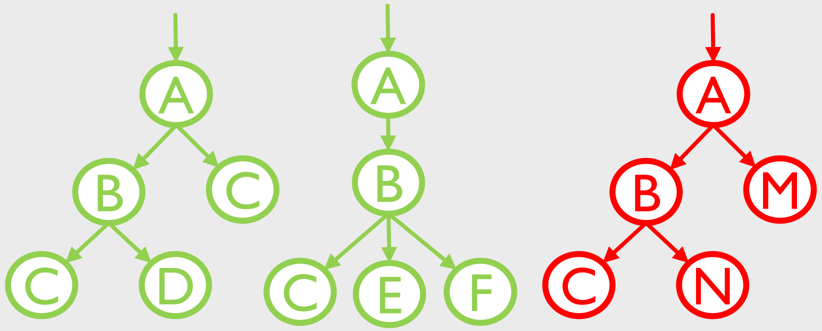
All traces satisfying and violating SLOs in the same test round

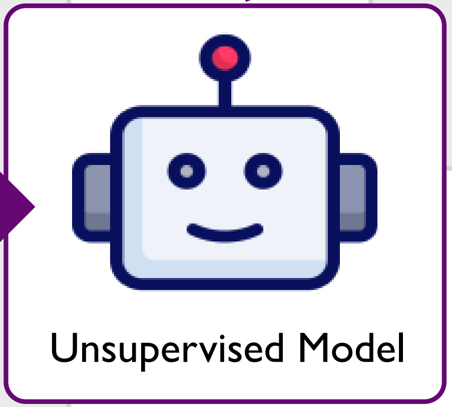


Problem Formulation

Training Input:

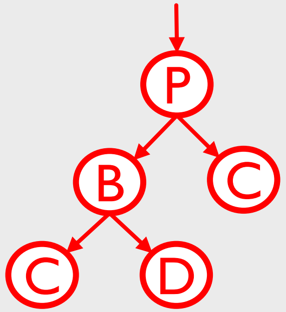
All traces satisfying and violating SLOs in the same test round





Inference Input

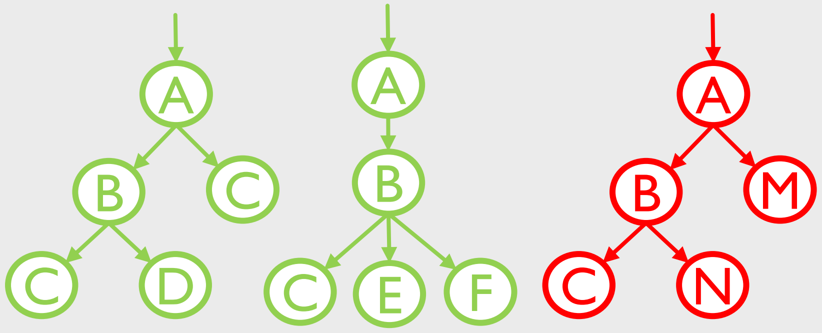
A single trace violating SLOs

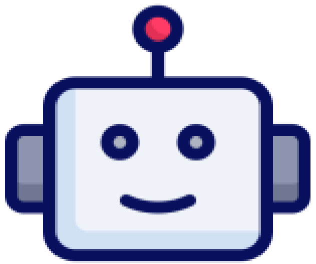


(with possible unseen microservices and calls)

Problem Formulation

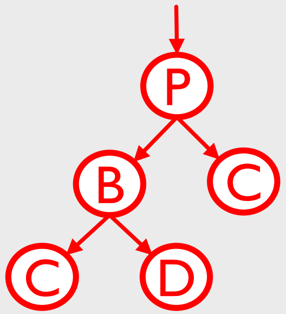
Training Input:
All traces satisfying and violating SLOs in the same test round



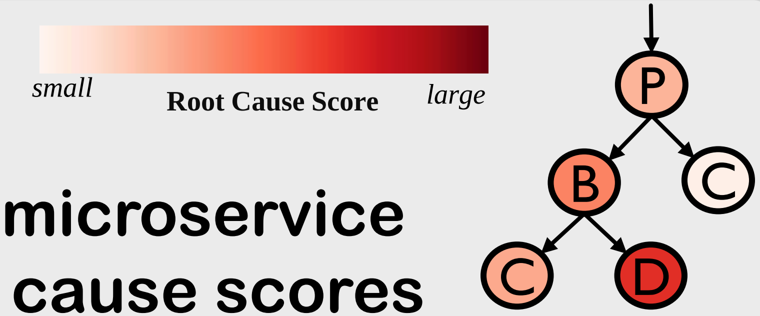


Unsupervised Model

Inference Input
A single trace violating SLOs



(with possible unseen microservices and calls)

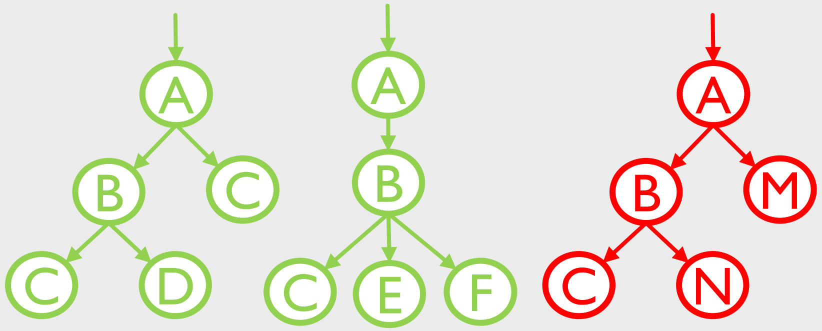


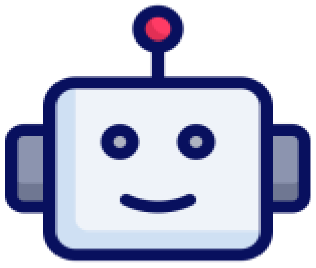
The microservice root cause scores
Inference Output

Problem Formulation

Training Input:

All traces satisfying and violating SLOs in the same test round

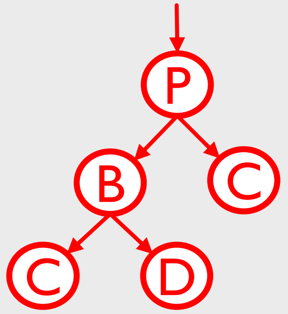




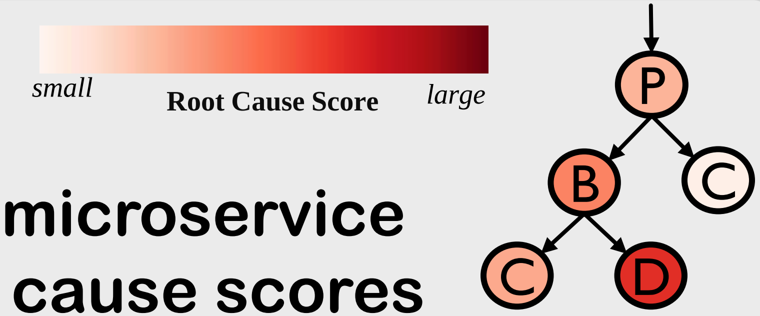
SparseRCA

Inference Input

A single trace violating SLOs



(with possible unseen microservices and calls)



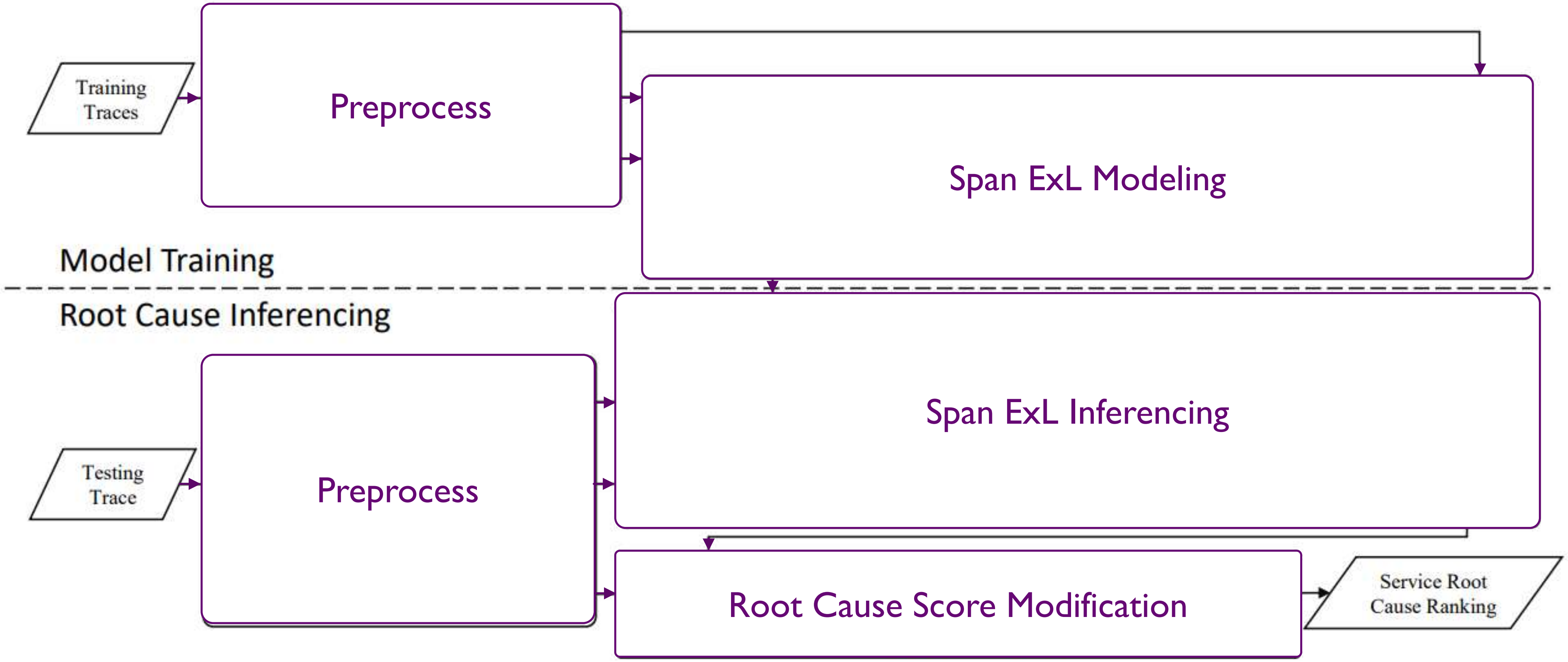
The microservice root cause scores

Inference Output

Outline

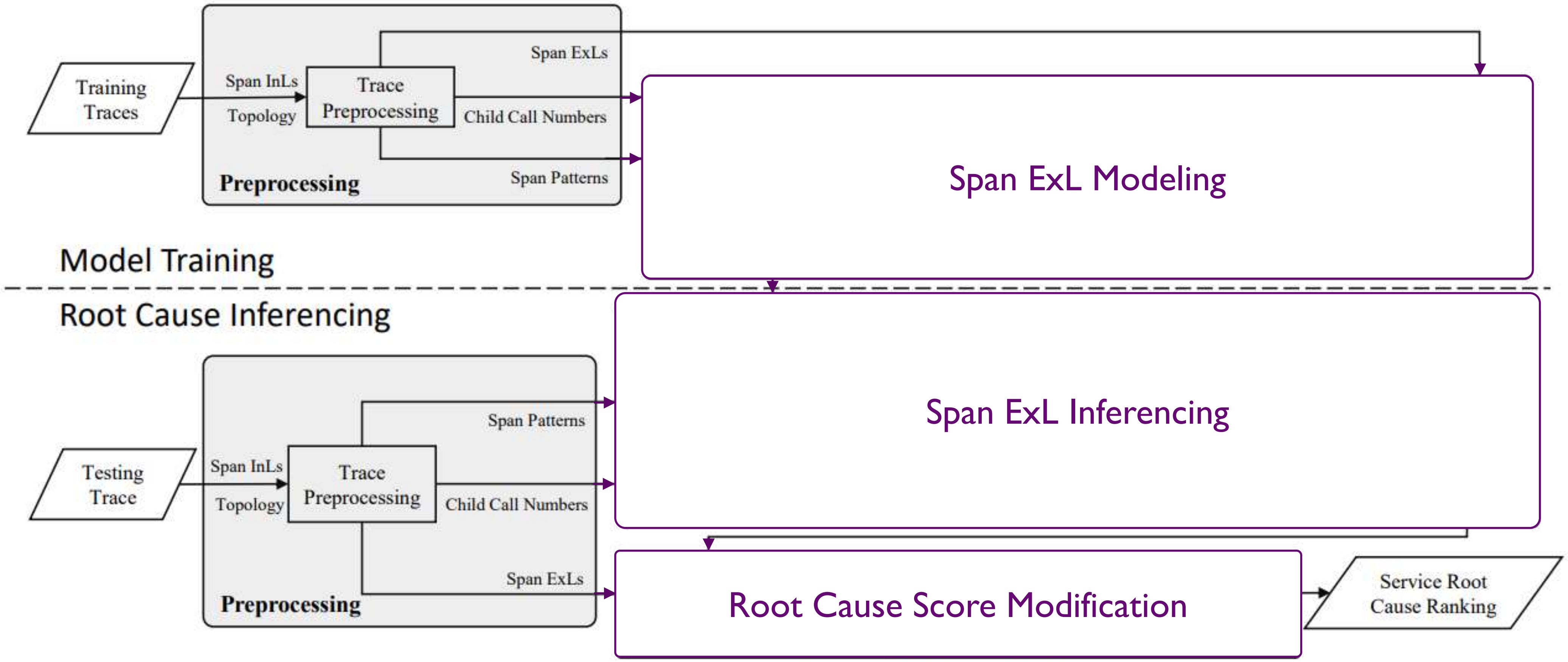
- Background
- Design
- Evaluation
- Summary

Overview of SparseRCA



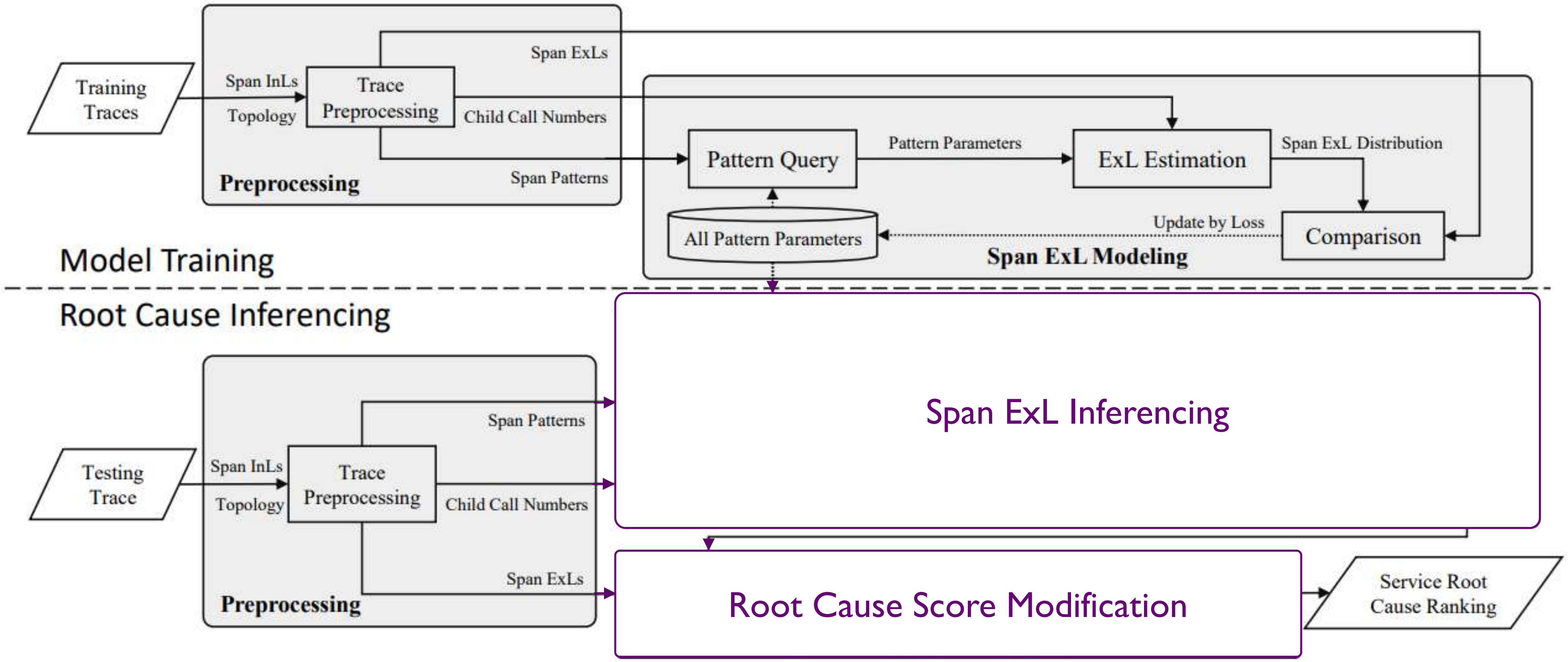
Workflow of SparseRCA

Overview of SparseRCA



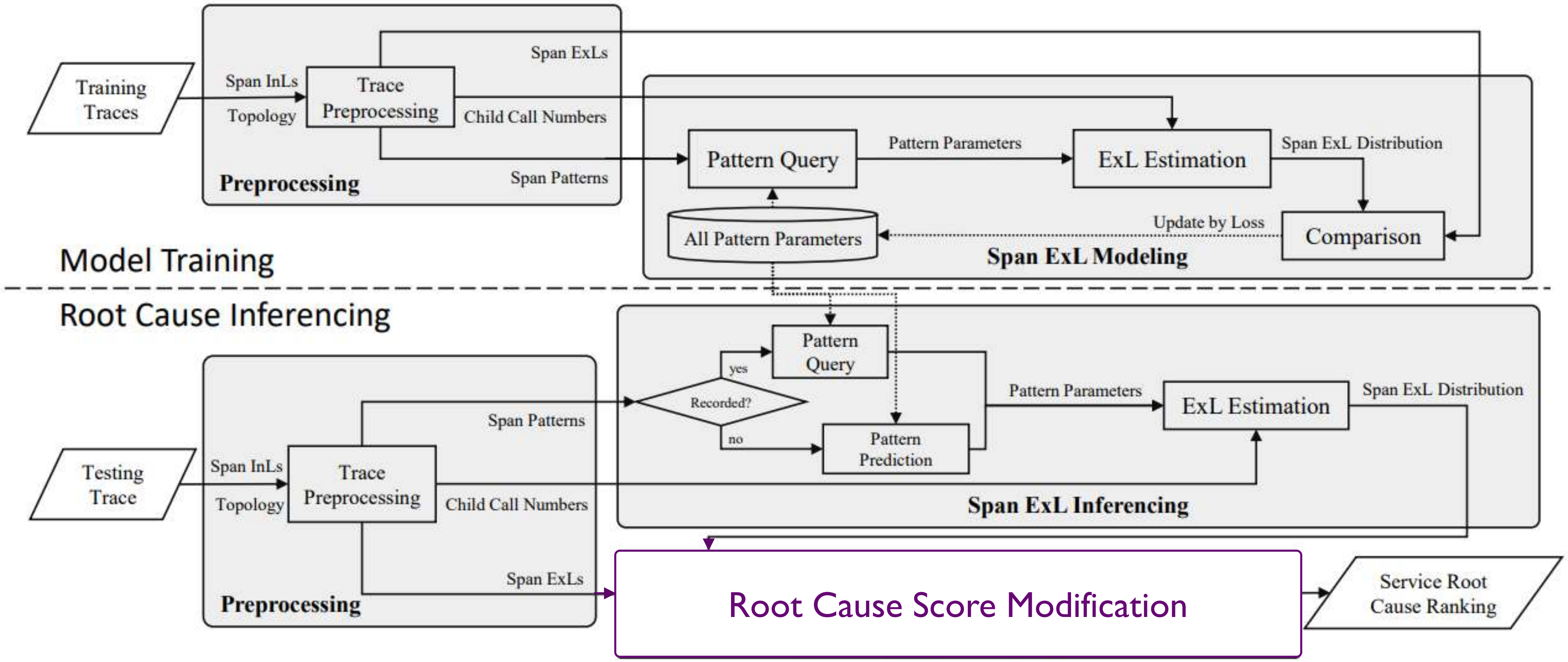
Workflow of SparseRCA

Overview of SparseRCA



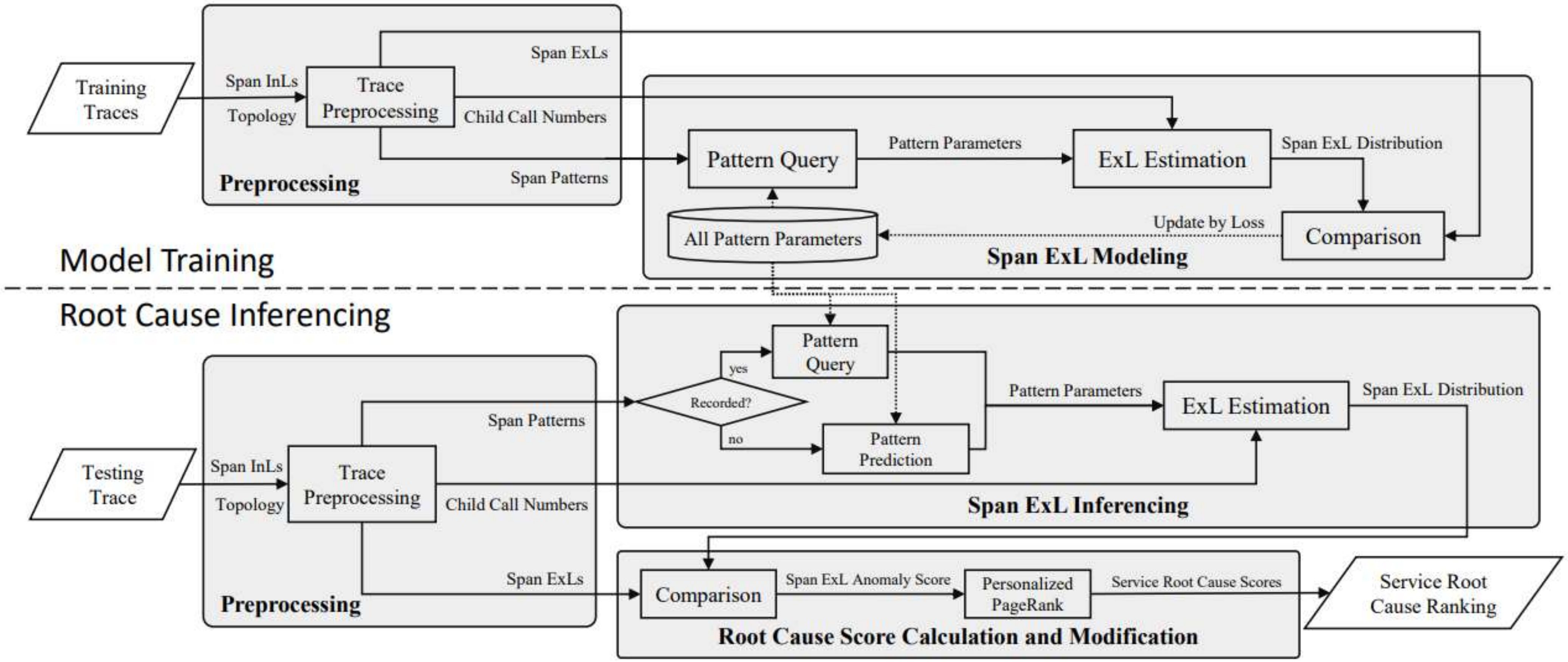
Workflow of SparseRCA

Overview of SparseRCA



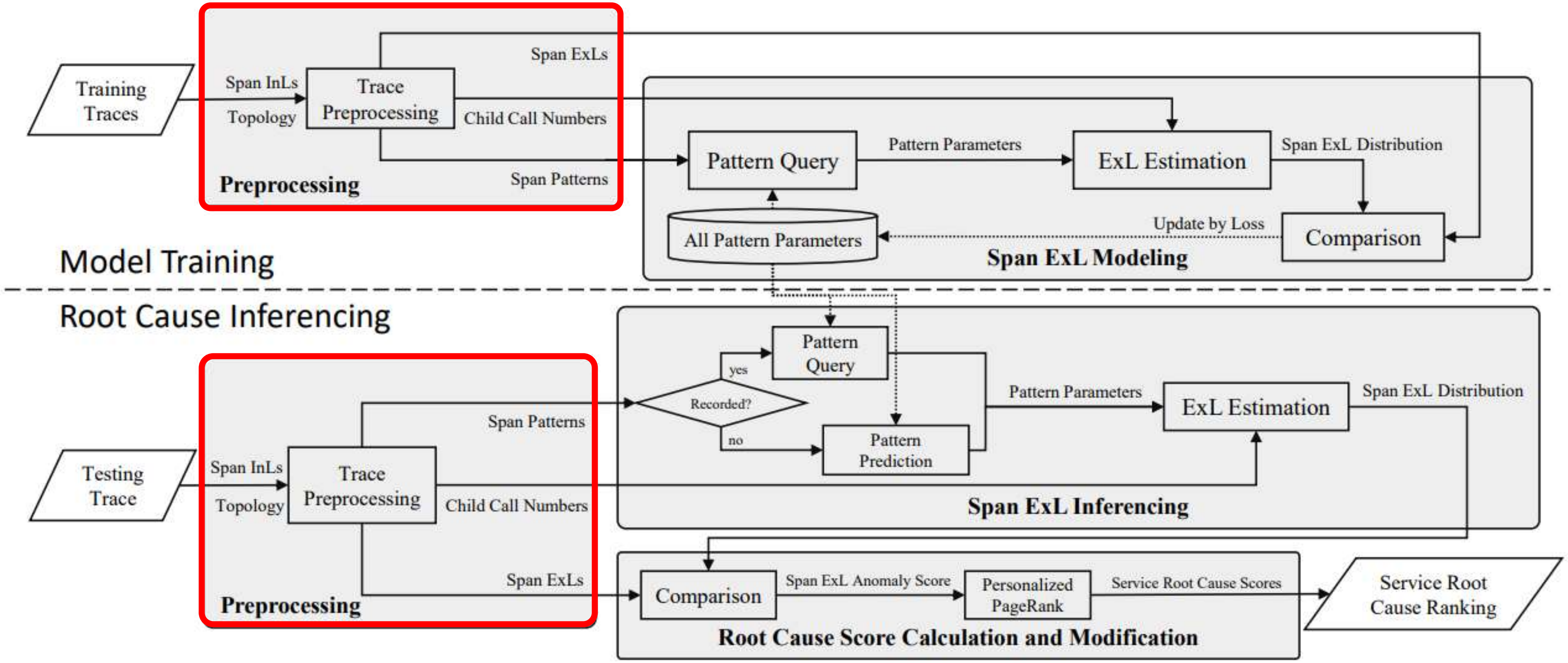
Workflow of SparseRCA

Overview of SparseRCA



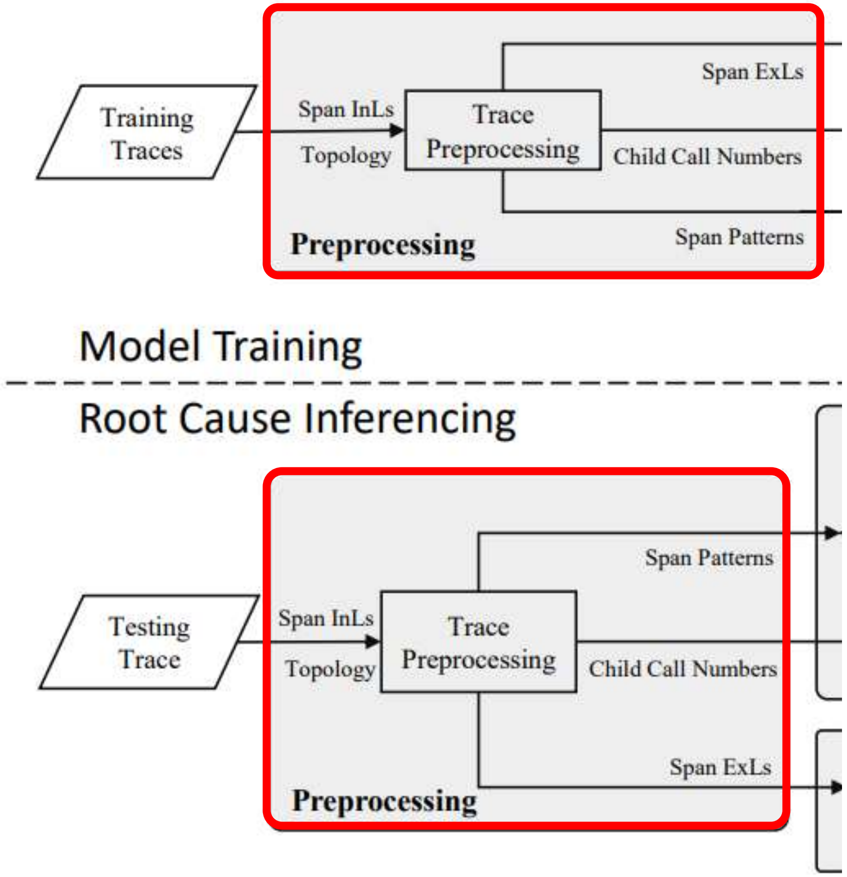
Workflow of SparseRCA

Overview of SparseRCA



Workflow of SparseRCA

Overview of SparseRCA



Workflow of SparseRCA

Concepts & Definitions

Insights from Empirical Experiments

Design: Concepts and Empirical Insights

Concepts & Definitions

Design: Concepts and Empirical Insights

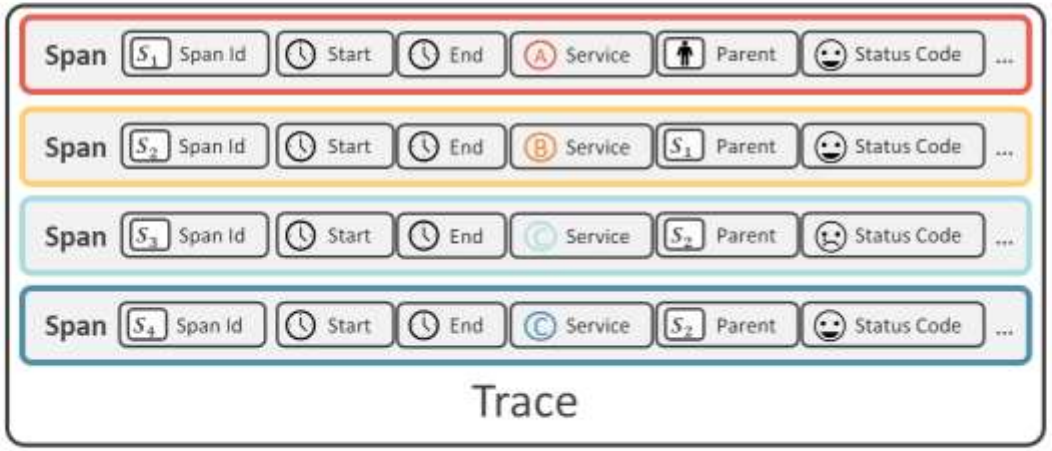
Concepts & Definitions

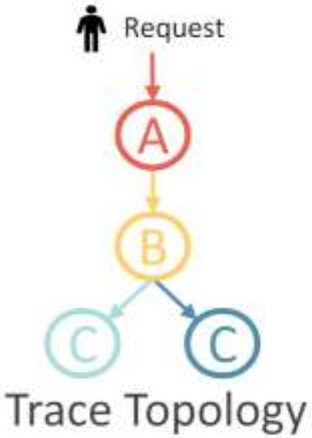
- Trace Span

Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Span

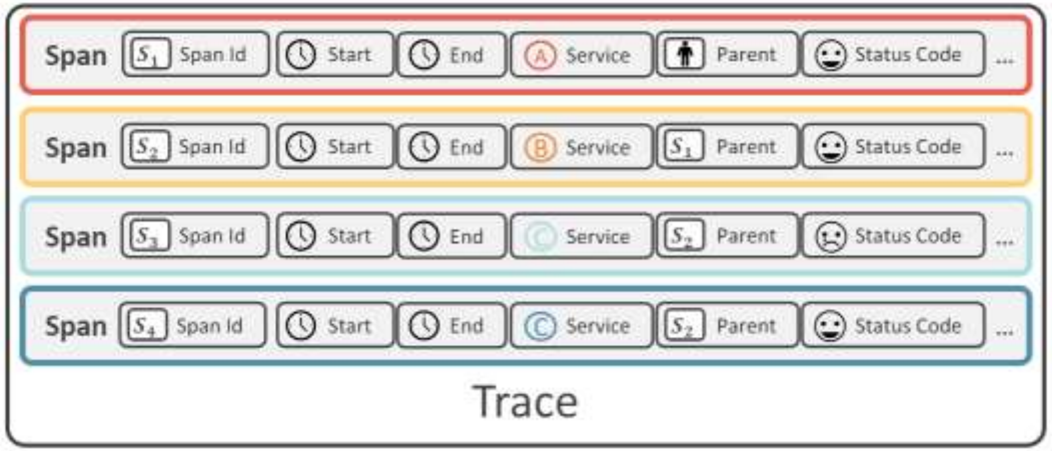


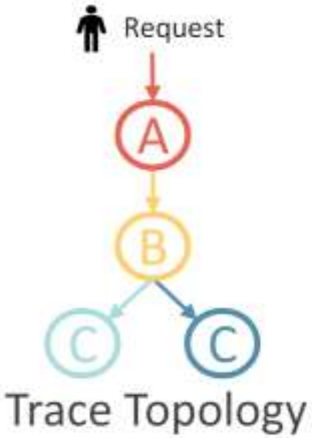


Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Span
- Inclusive Latency (InL) of a span
  - Overall time consumption for a span

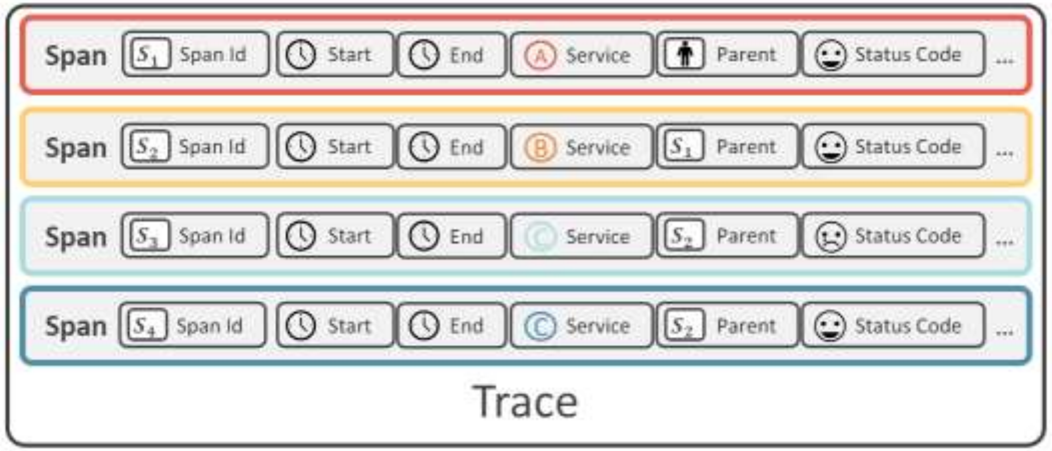




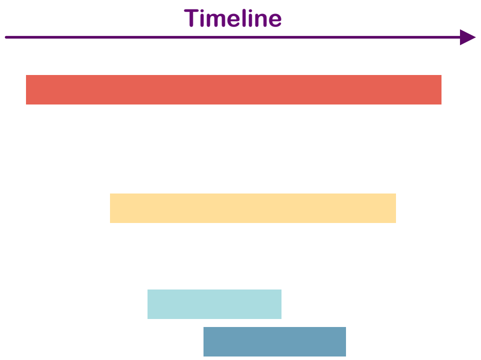
Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Span
- Inclusive Latency (InL) of a span
  - Overall time consumption for a span



Trace Topology



Span InLs

Design: Concepts and Empirical Insights

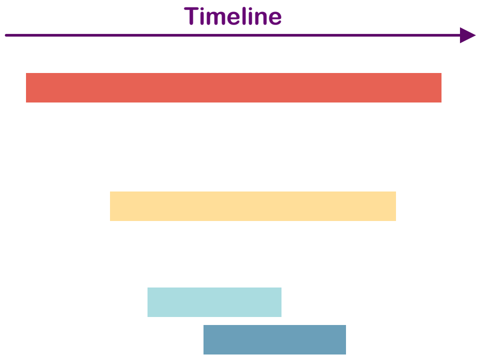
Concepts & Definitions

- Trace Span

- Inclusive Latency (InL) of a span
  - Overall time consumption for a span
- Exclusive Latency(ExL) of a span
  - InL excluding the InL of child spans

Trace

Trace Topology

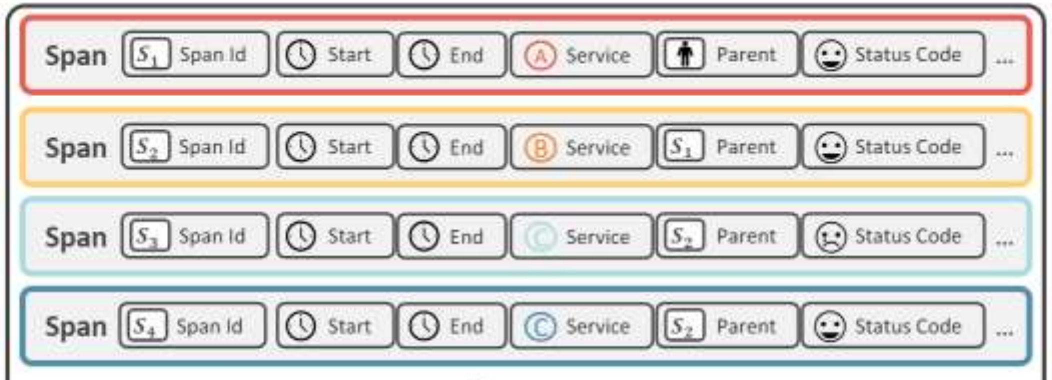


Span InLs

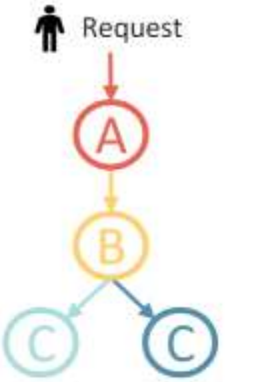
Design: Concepts and Empirical Insights

Concepts & Definitions

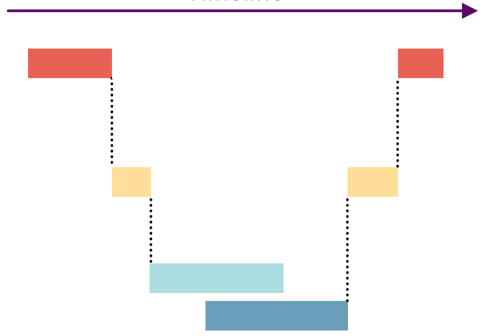
- Trace Span
- Inclusive Latency (InL) of a span
  - Overall time consumption for a span
- Exclusive Latency(ExL) of a span
  - InL excluding the InL of child spans



Trace



Trace Topology

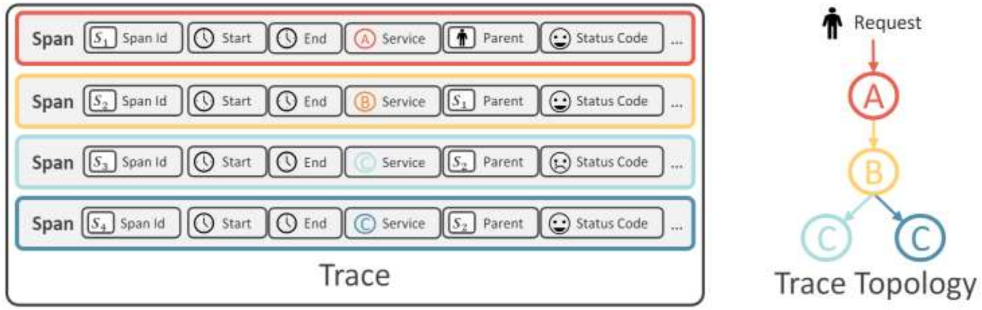


Span ExLs

Design: Concepts and Empirical Insights

Concepts & Definitions

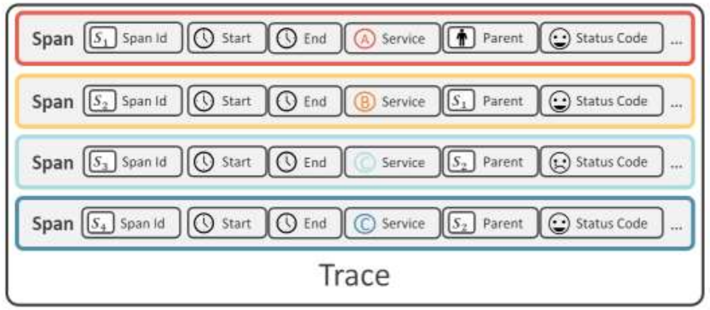
- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span

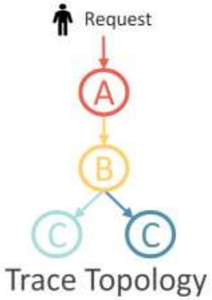


Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span



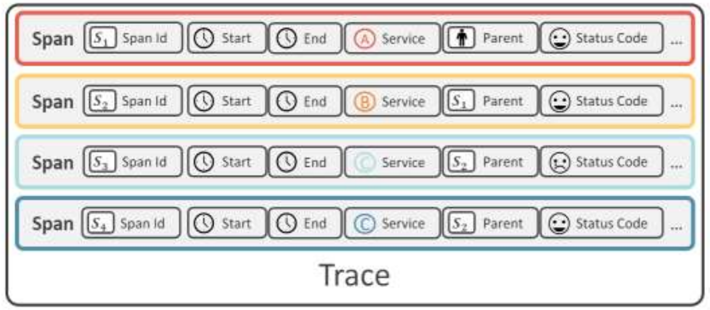


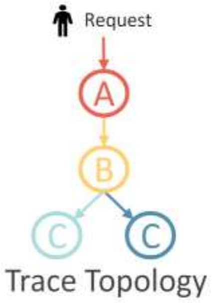
Among InL, ExL, and status codes, what metric is most suitable as RCA indicator?

Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span





Among InL, ExL, and status codes, what metric is most suitable as RCA indicator?

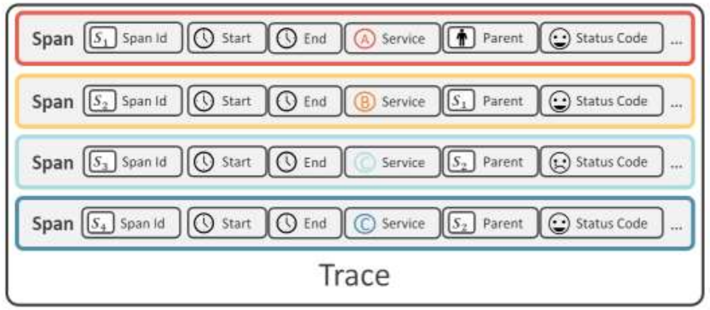
Rank Metric	Top-k Accuracy (%)		
	A@1	A@2	A@3
ExL Anomaly	30.12	50.15	57.85
InL Anomaly	27.52	45.34	54.68
Status Code Anomaly	8.52	21.64	35.41
Random Selection	8.77	17.54	26.31

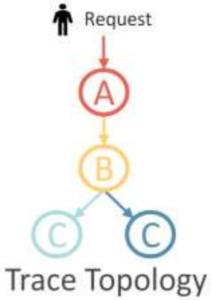
Empirical Comparison among the metrics

Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span





Among InL, ExL, and status codes, what metric is most suitable as RCA indicator?

The ExL serves as an effective indicator to coarsely identify the root causes

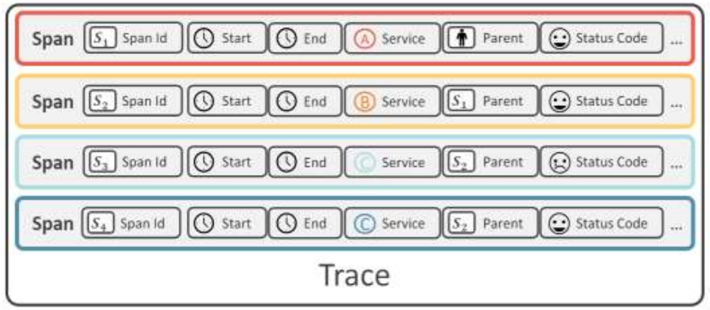
Rank Metric	Top-k Accuracy (%)		
	A@1	A@2	A@3
ExL Anomaly	30.12	50.15	57.85
InL Anomaly	27.52	45.34	54.68
Status Code Anomaly	8.52	21.64	35.41
Random Selection	8.77	17.54	26.31

Empirical Comparison among the metrics

Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span



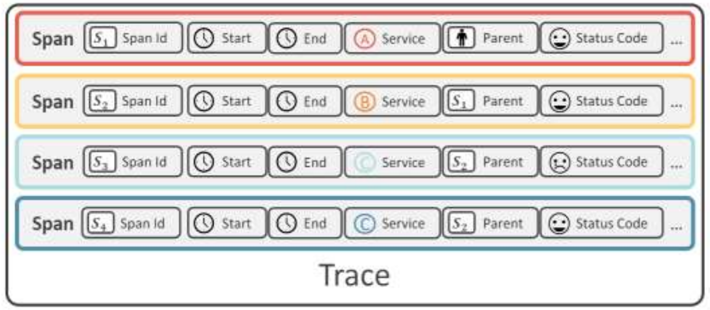
Among InL, ExL, and status codes, what metric is most suitable as RCA indicator?

The ExL serves as an effective indicator to coarsely identify the root causes

Design: Concepts and Empirical Insights

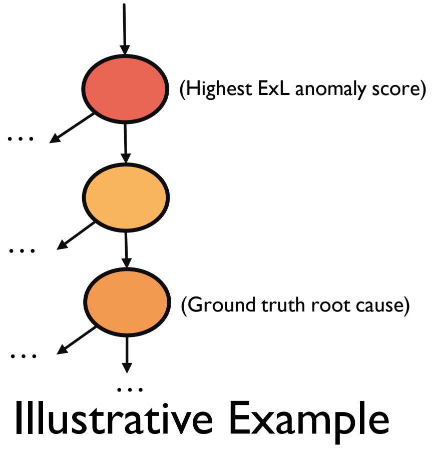
Concepts & Definitions

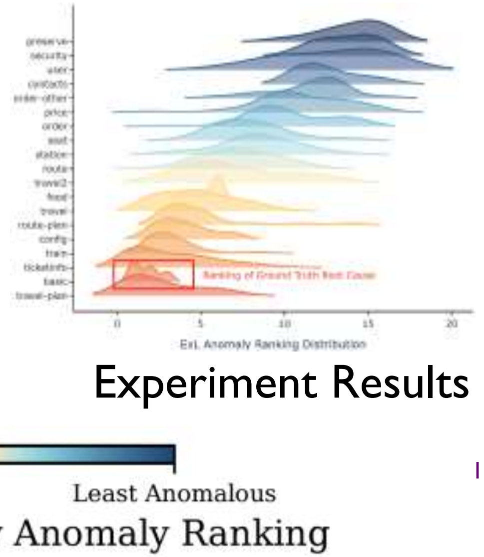
- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span



Among InL, ExL, and status codes, what metric is most suitable as RCA indicator?

The ExL serves as an effective indicator to coarsely identify the root causes

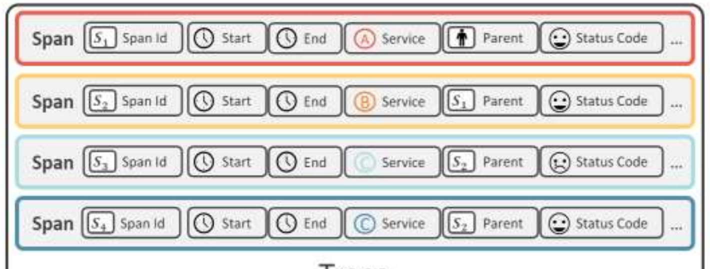




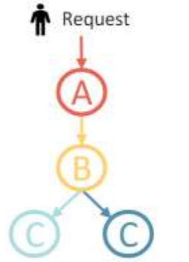
Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span



Trace

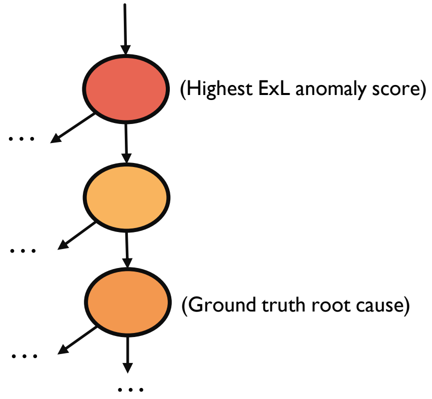


Trace Topology

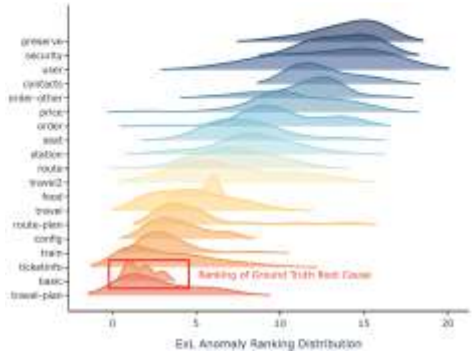
Among InL, ExL, and status codes, what metric is most suitable as RCA indicator?

The ExL serves as an effective indicator to coarsely identify the root causes

RCA cannot fully depend on ExL anomalies because anomalous high ExLs may also appear in ancestor nodes



Illustrative Example



Experiment Results

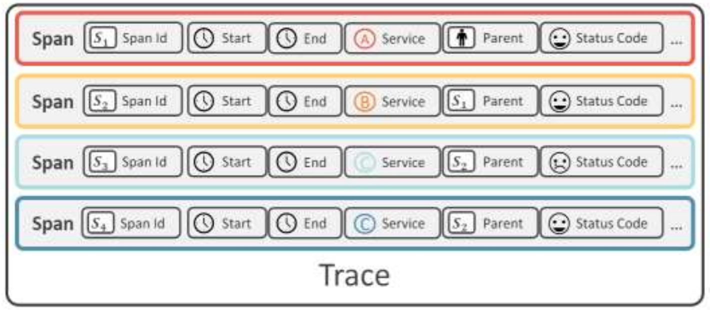
Most Anomalous   Least Anomalous

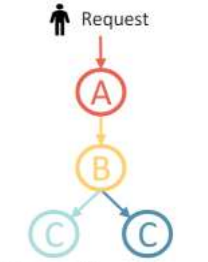
Exclusive Latency Anomaly Ranking

Design: Concepts and Empirical Insights

Concepts & Definitions

- Trace Spans
- Inclusive Latency (InL) of a span
- Exclusive Latency(ExL) of a span



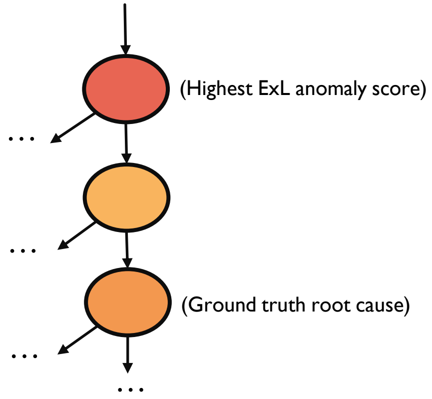


Trace Topology

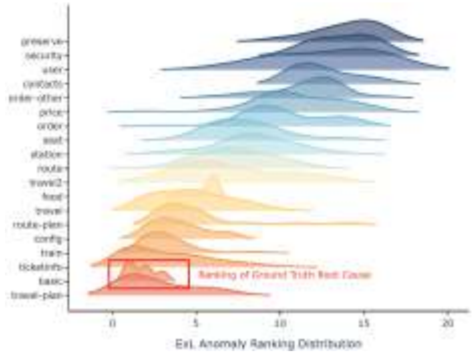
Among InL, ExL, and status codes, what metric is most suitable as RCA indicator?

The ExL serves as an effective indicator to coarsely identify the root causes

RCA cannot fully depend on ExL anomalies because anomalous high ExLs may also appear in ancestor nodes



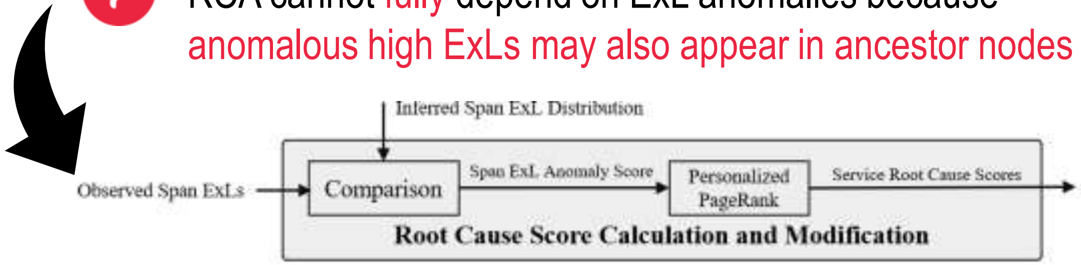
Illustrative Example



Experiment Results

Most Anomalous
Least Anomalous

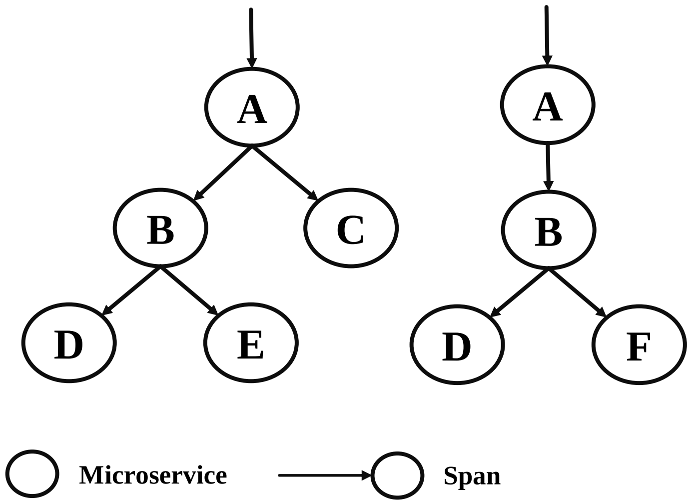
Exclusive Latency Anomaly Ranking



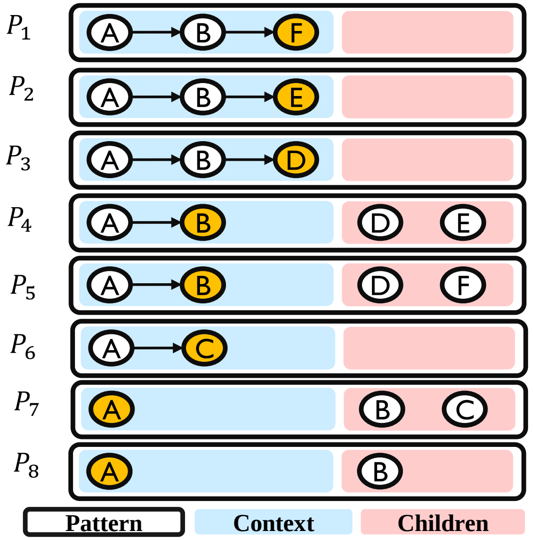
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

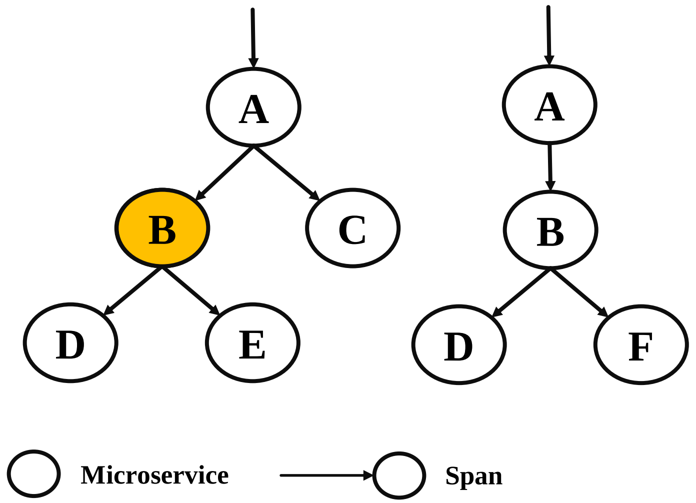


All Span Patterns in the Example Traces

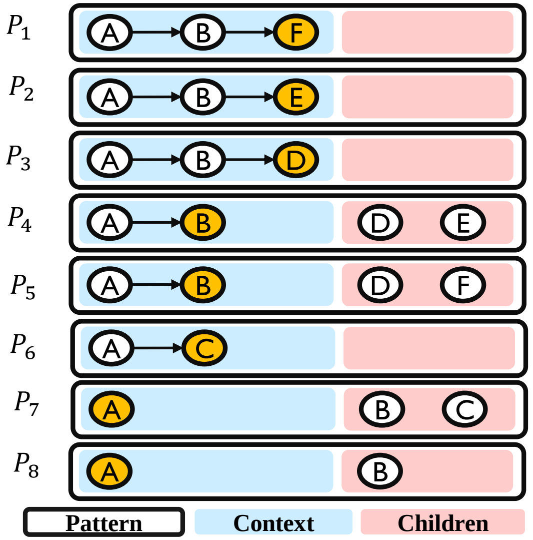
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

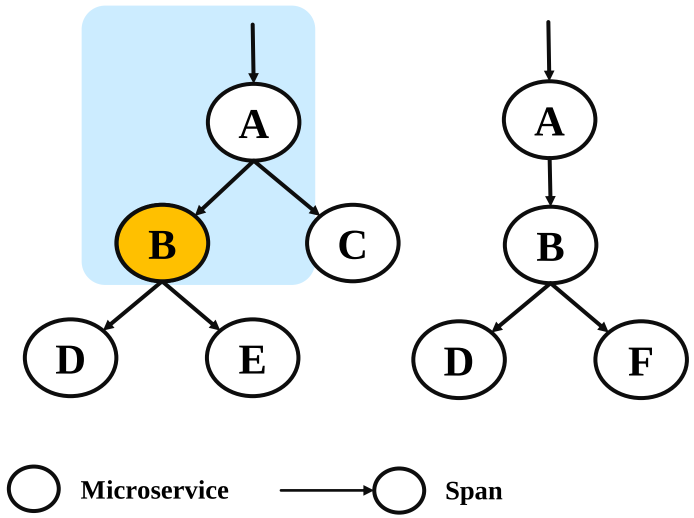


All Span Patterns in the Example Traces

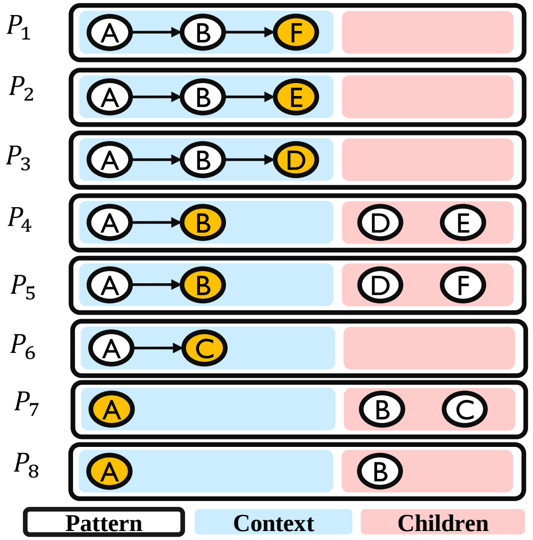
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

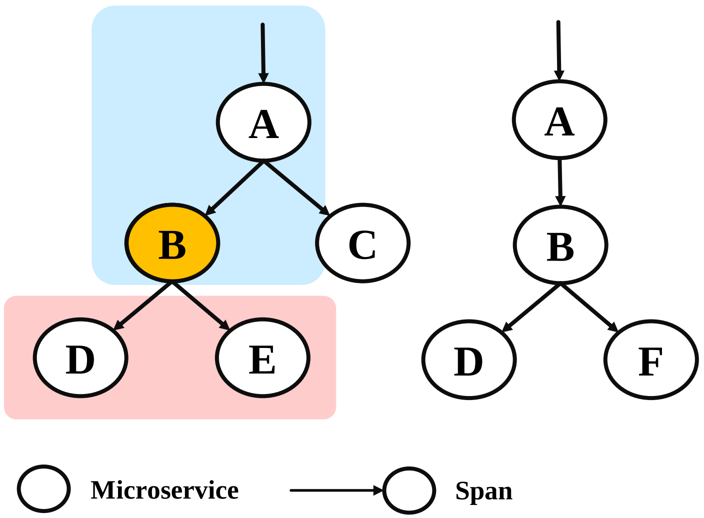


All Span Patterns in the Example Traces

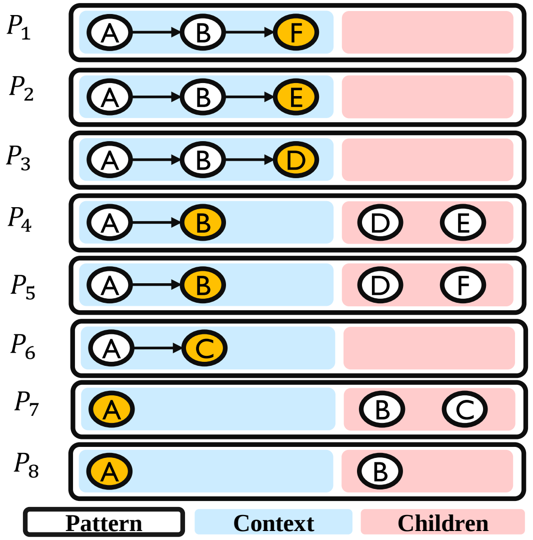
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

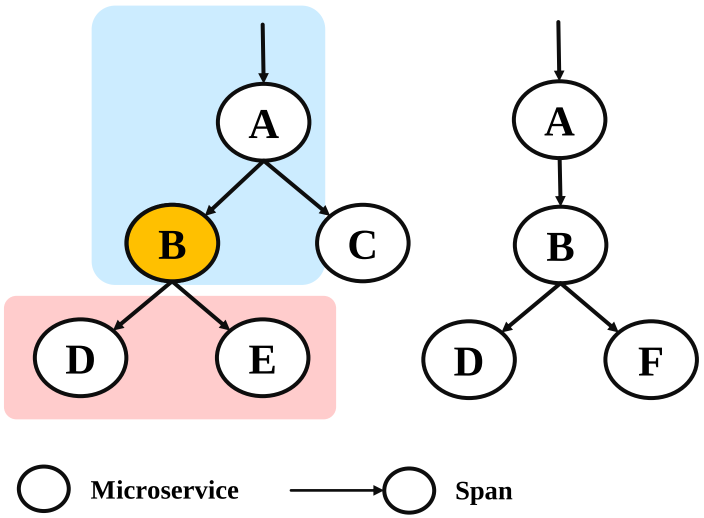


All Span Patterns in the Example Traces

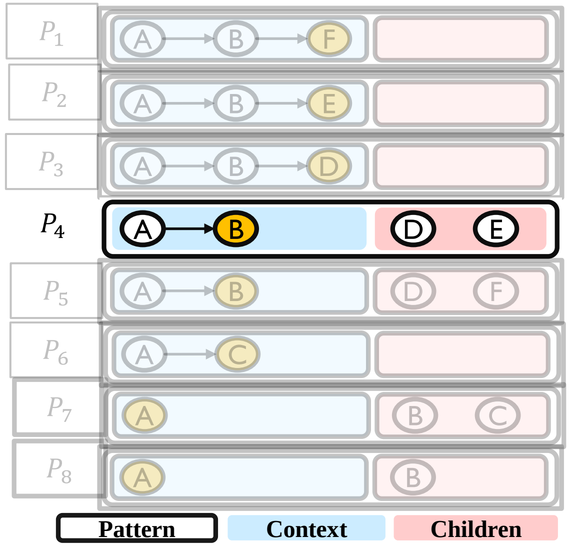
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

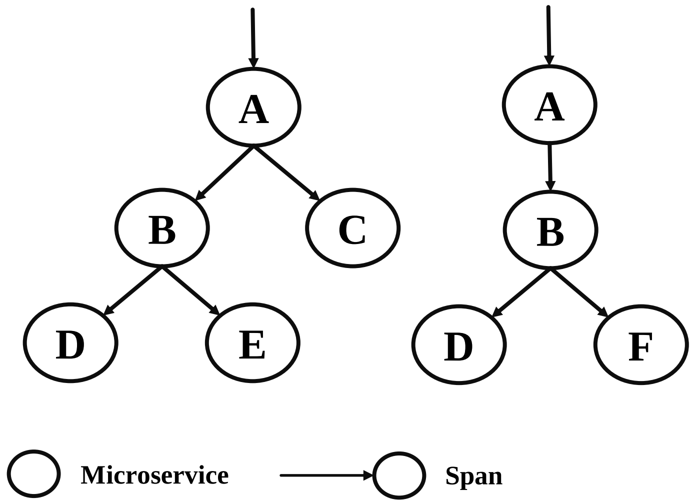


All Span Patterns in the Example Traces

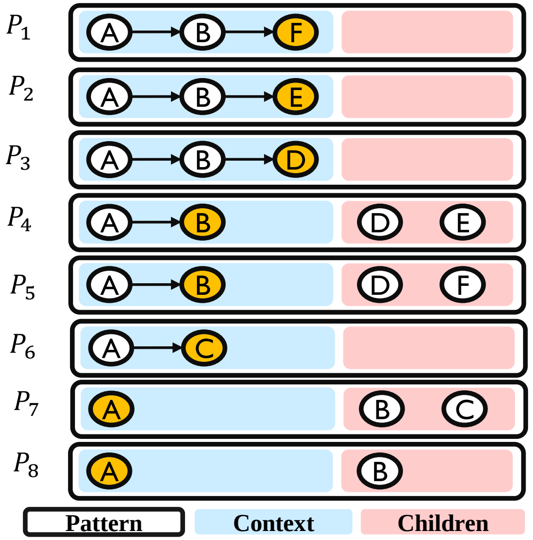
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

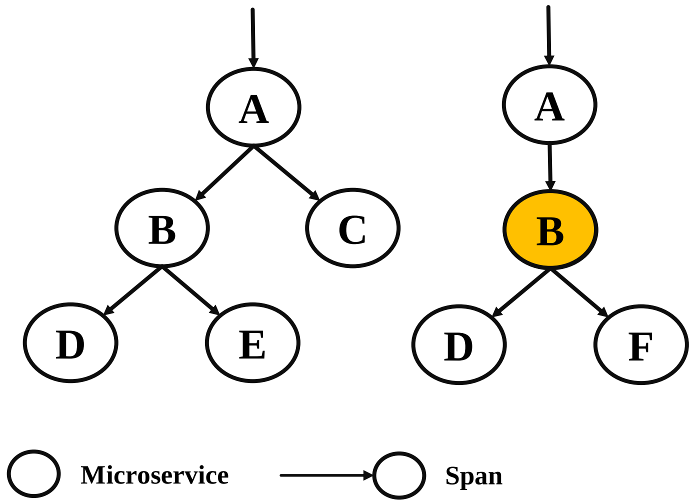


All Span Patterns in the Example Traces

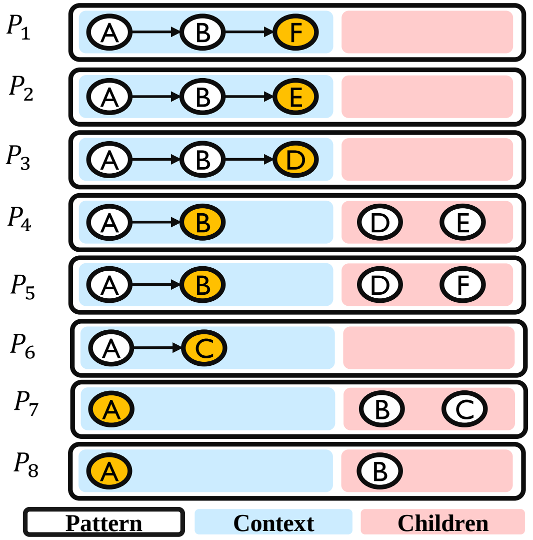
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

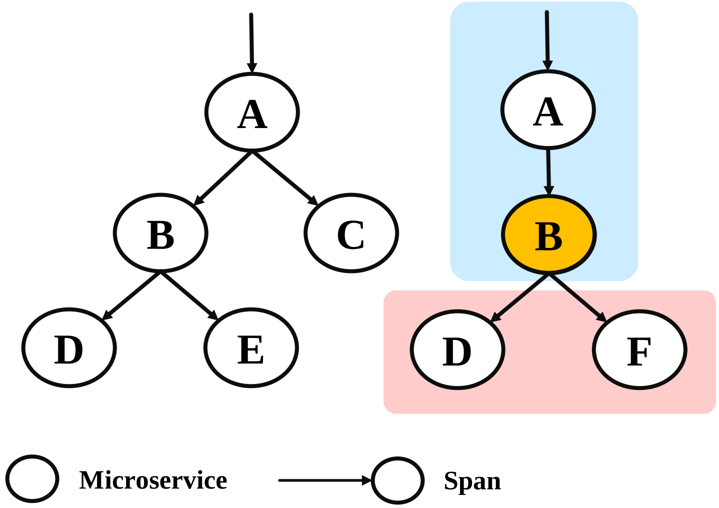


All Span Patterns in the Example Traces

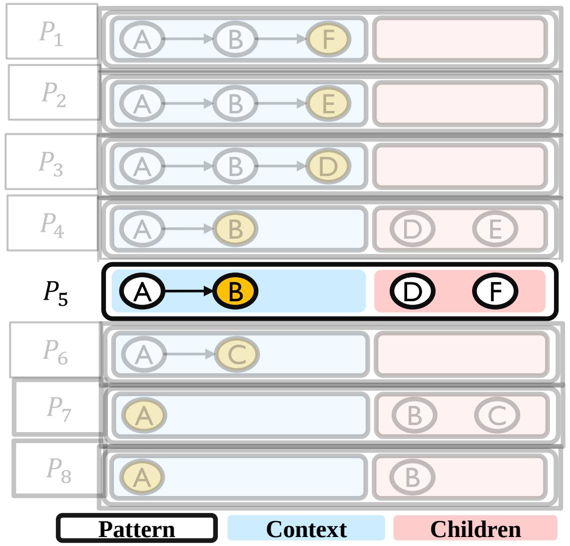
Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



Two Example Traces

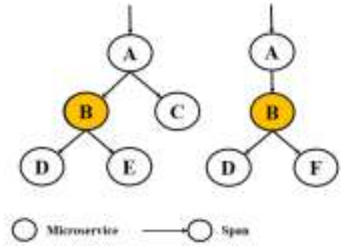


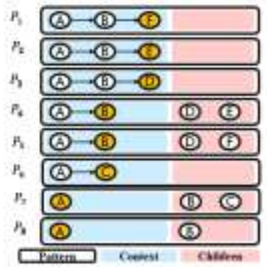
All Span Patterns in the Example Traces

Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set

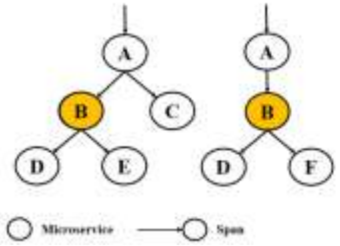


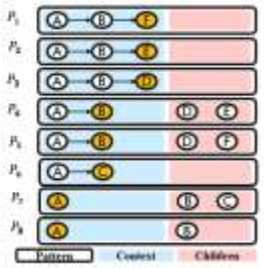


Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set



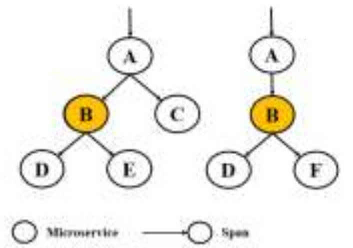


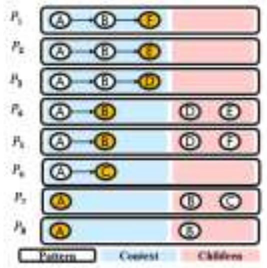
Why is the span pattern important?

Design: Concepts and Empirical Insights

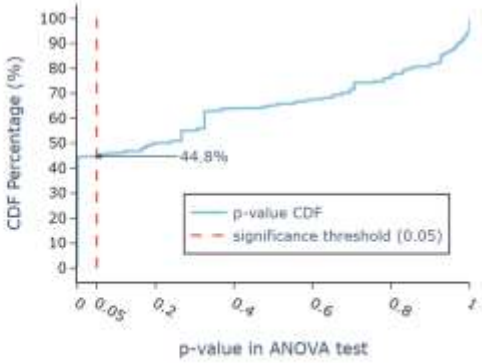
Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set





Why is the span pattern important?

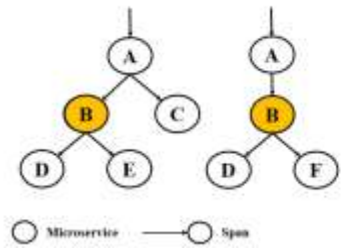


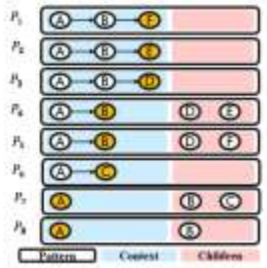
Relevance Hypothesis Test

Design: Concepts and Empirical Insights

Concepts & Definitions

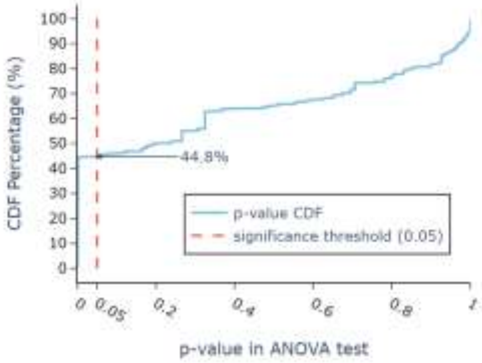
- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set





Why is the span pattern important?

The latency of nearly half of the spans is significantly related to their contexts

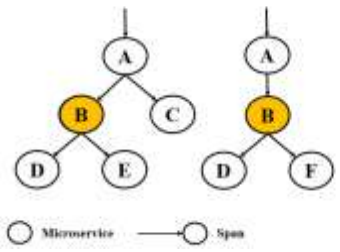


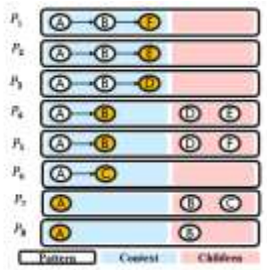
Relevance Hypothesis Test

Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set





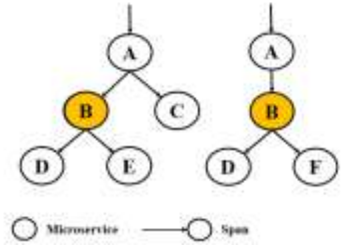
Why is the span pattern important?

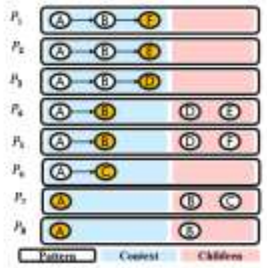
The latency of nearly half of the spans is significantly related to their contexts

Design: Concepts and Empirical Insights

Concepts & Definitions

- Pattern of a span
  - **Context** upstream microservice list
  - **Children Set** downstream microservice set





Why is the span pattern important?

The latency of nearly half of the spans is significantly related to their contexts

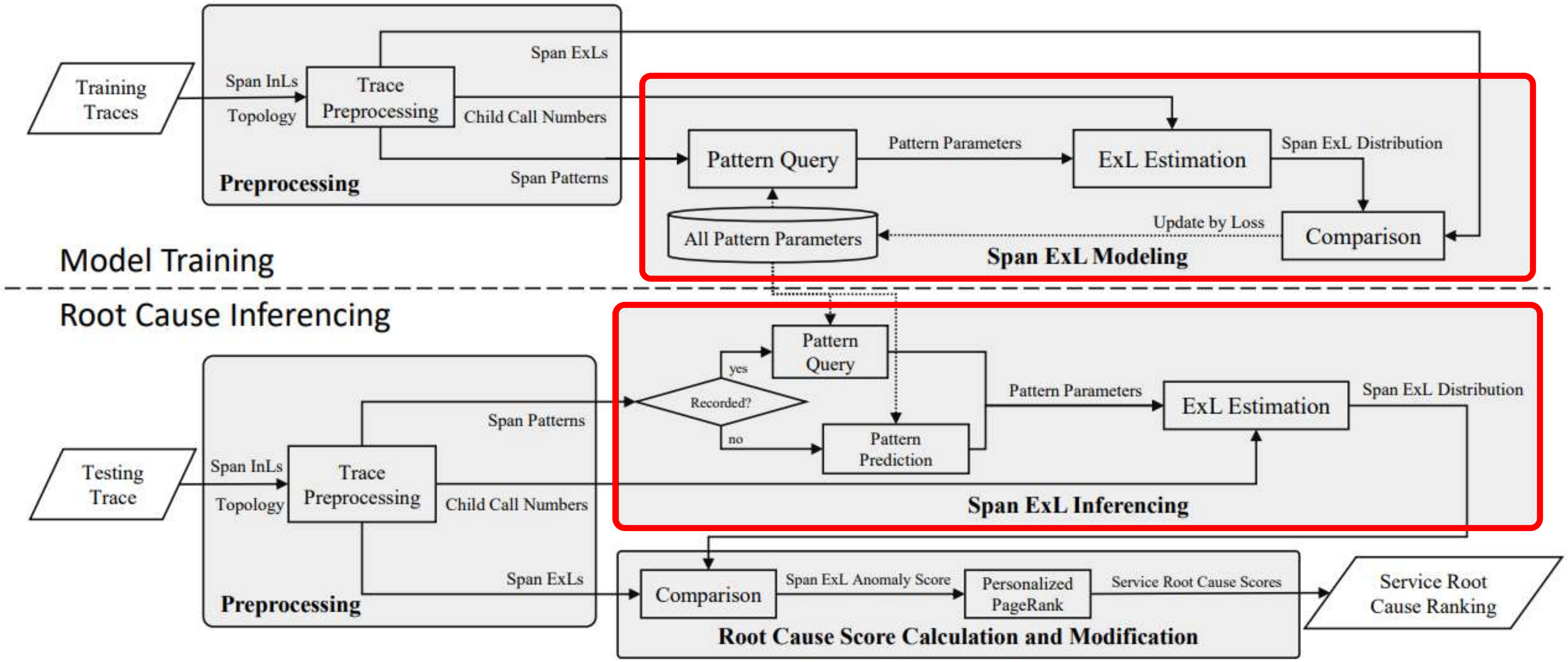
The latency of about 60% of the spans is significantly related to their call numbers to children

TABLE VIII: Percentage of context-aware spans with ExL relevant and irrelevant to the number of calls to child microservices.

Category	H_0: Irrelevant	p-value	Pct. of Context (%) ANOVA	Kruskal-Wallis
Relevant	Reject	< 0.05	59.6%	63.4%
Irrelevant	Accept	≥ 0.05	40.4%	36.6%

Relevance Hypothesis Test

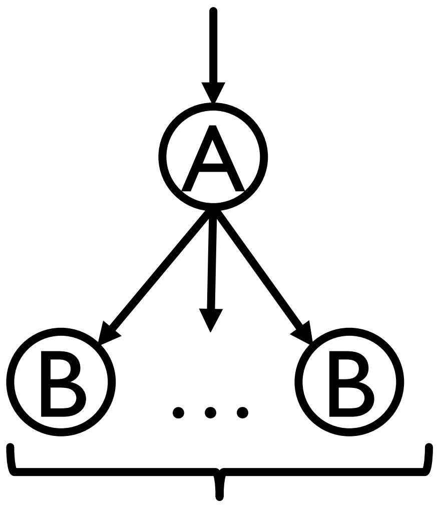
Overview of SparseRCA



Workflow of SparseRCA

Design: Span ExL Modeling & Inferencing

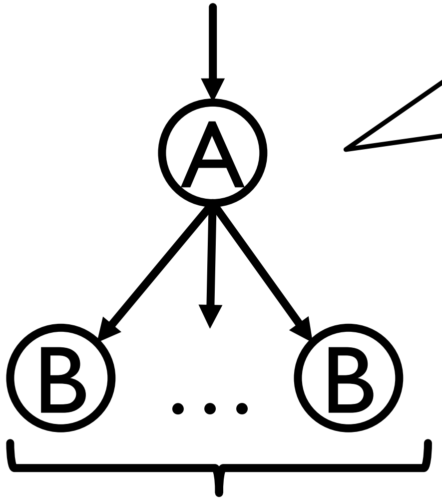
Modeling ExL of a Span: From a Toy Example



Service A which calls Service B for n times

Design: Span ExL Modeling & Inferencing

Modeling ExL of a Span: From a Toy Example



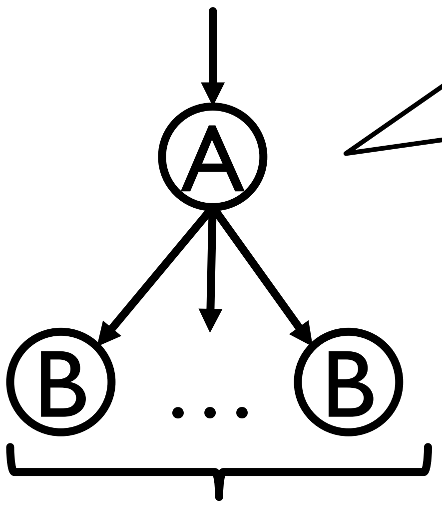
Service A which calls Service B for n times

```
ServiceAOnRequest(request){
        preprocessRequest(request)
        loop n times{
                preprocessBeforeCallChild()
                callMicroserviceB()
                postprocessResponseFromChild()
        }
        postprocessResponse()
        exceptionCaseProcess()
}
```

Pseudocode of Service A

Design: Span ExL Modeling & Inferencing

Modeling ExL of a Span: From a Toy Example



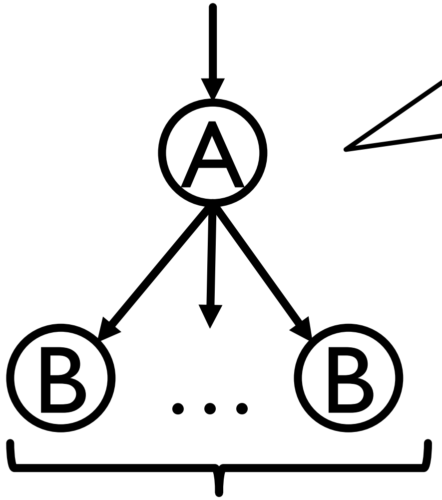
Service A which calls Service B for n times

```
ServiceAOnRequest(request){
        preprocessRequest(request)
        loop n times{
                preprocessBeforeCallChild()
                callMicroserviceB()
                postprocessResponseFromChild()
        }
        postprocessResponse()
        exceptionCaseProcess()
}
```

Pseudocode of Service A

Design: Span ExL Modeling & Inferencing

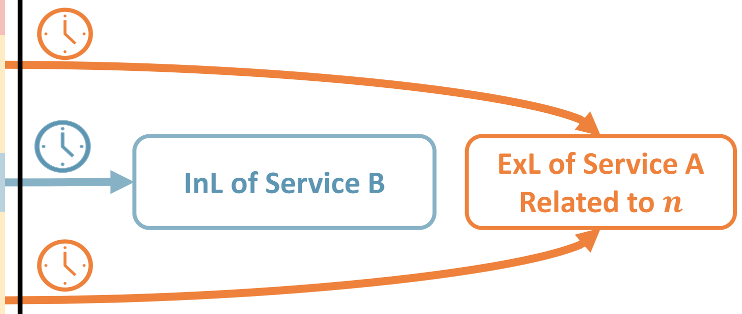
Modeling ExL of a Span: From a Toy Example



Service A which calls Service B for n times

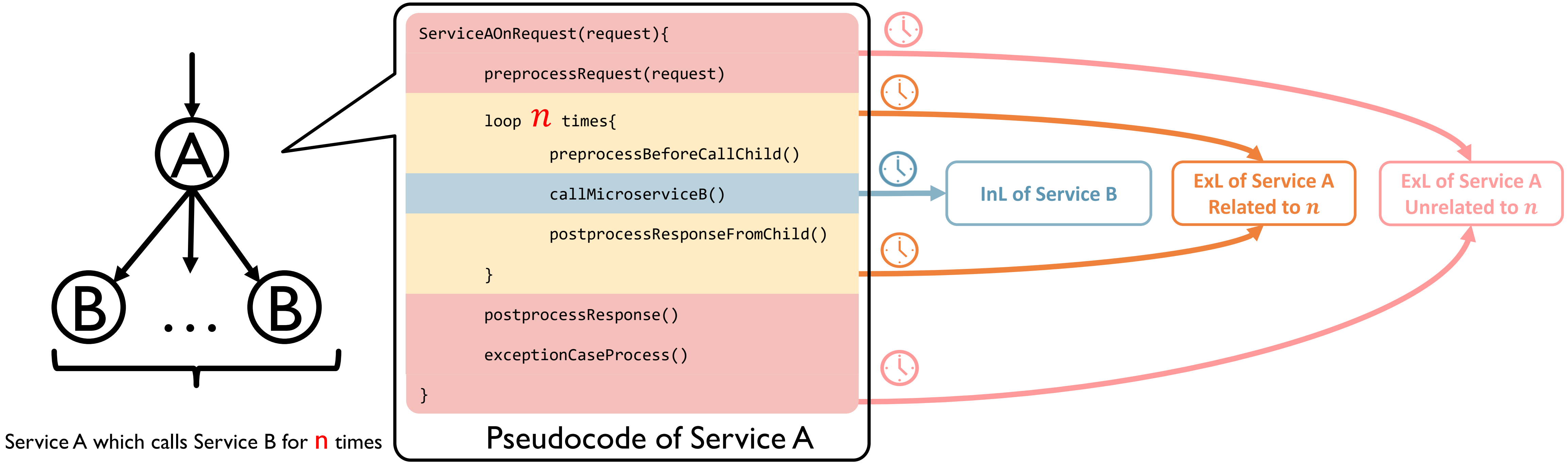
```
ServiceAOnRequest(request){
        preprocessRequest(request)
        loop n times{
                preprocessBeforeCallChild()
                callMicroserviceB()
                postprocessResponseFromChild()
        }
        postprocessResponse()
        exceptionCaseProcess()
}
```

Pseudocode of Service A



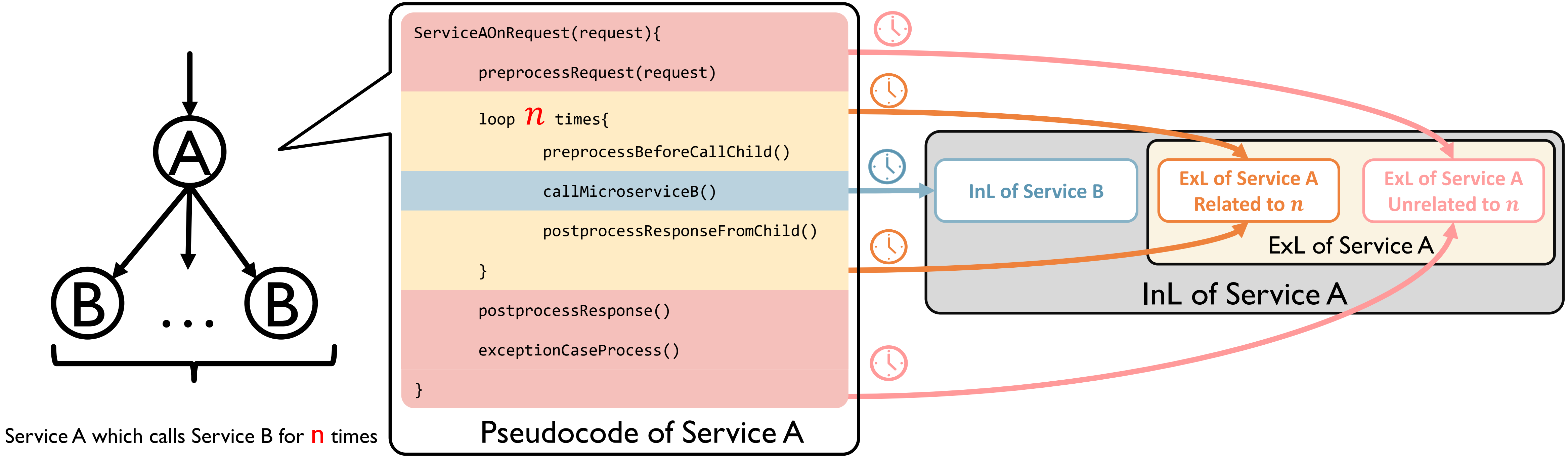
Design: Span ExL Modeling & Inferencing

Modeling ExL of a Span: From a Toy Example



Design: Span ExL Modeling & Inferencing

Modeling ExL of a Span: From a Toy Example

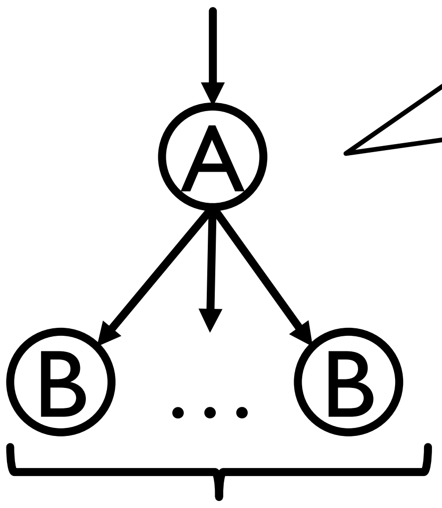


Design: Span ExL Modeling & Inferencing

Modeling ExL of a Span: From a Toy Example

There are Two parts of ExL:

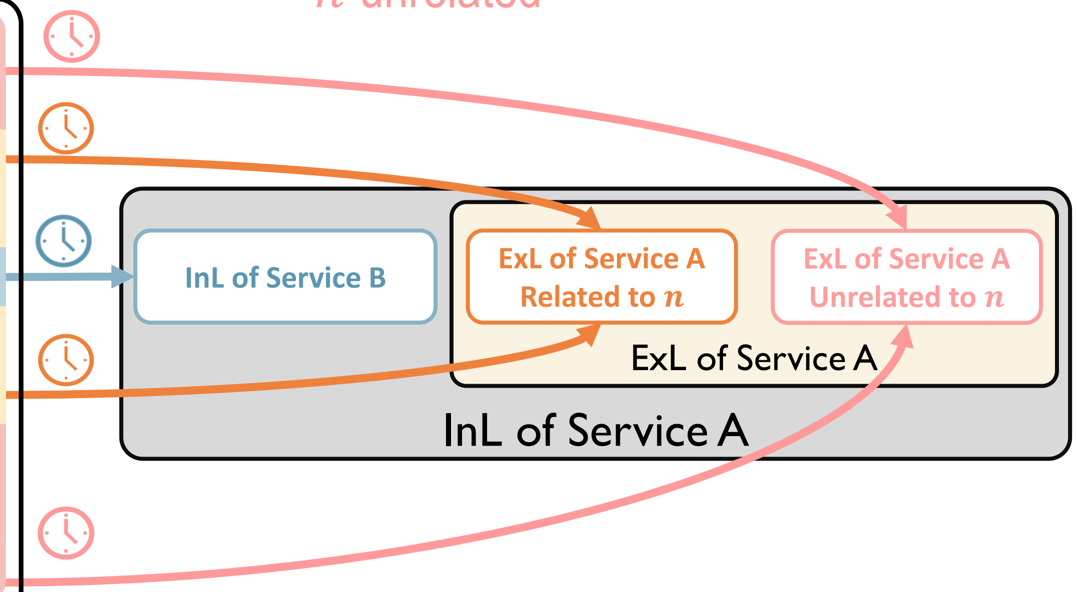
- n-related
- n-unrelated



Service A which calls Service B for n times

```
ServiceAOnRequest(request){
        preprocessRequest(request)
        loop n times{
                preprocessBeforeCallChild()
                callMicroserviceB()
                postprocessResponseFromChild()
        }
        postprocessResponse()
        exceptionCaseProcess()
}
```

Pseudocode of Service A

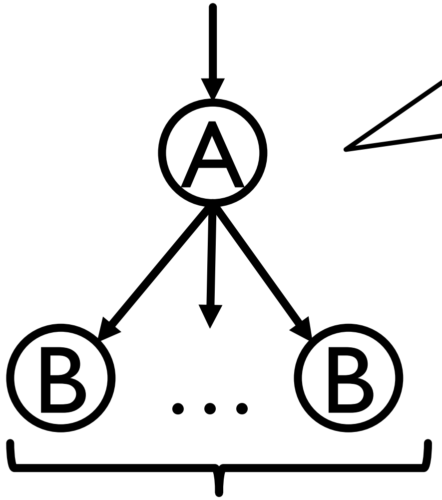


Design: Span ExL Modeling & Inferencing

Modeling ExL of a Span: From a Toy Example

There are Two parts of ExL:

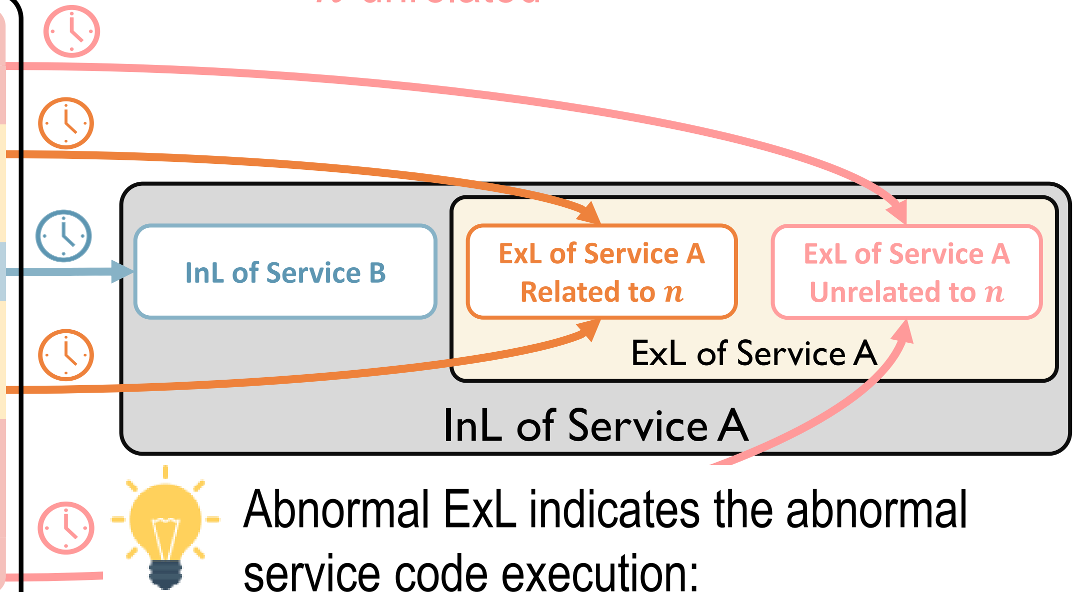
- n-related
- n-unrelated



Service A which calls Service B for n times

```
ServiceAOnRequest(request){
    preprocessRequest(request)
    loop n times{
        preprocessBeforeCallChild()
        callMicroserviceB()
        postprocessResponseFromChild()
    }
    postprocessResponse()
    exceptionCaseProcess()
}
```

Pseudocode of Service A

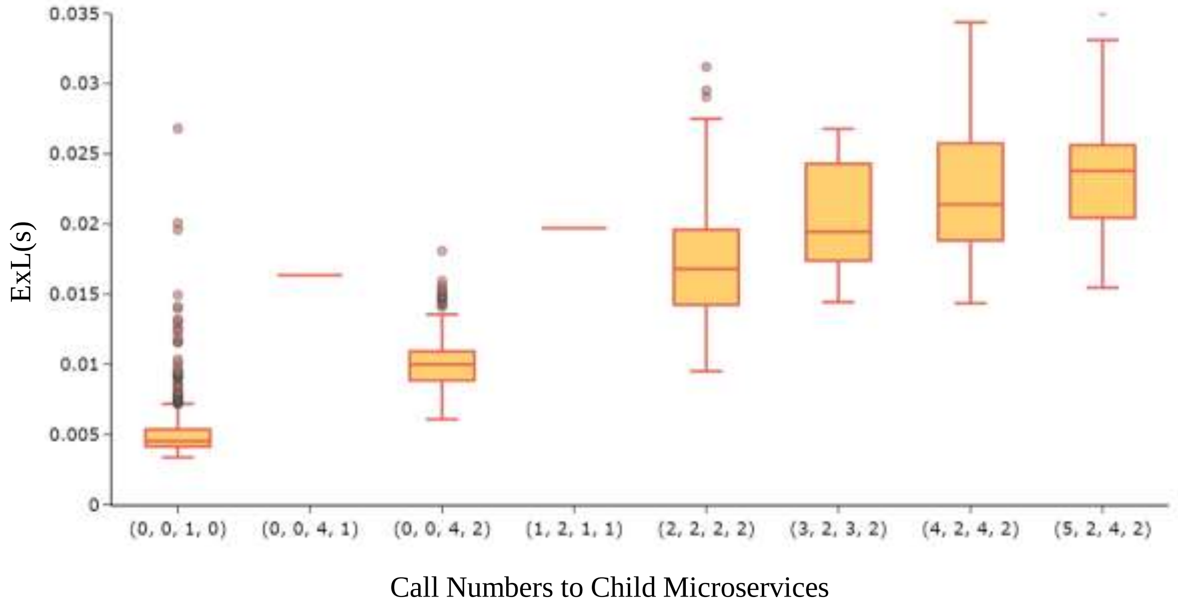


Abnormal ExL indicates the abnormal service code execution:

- Unexpected code branch
- Unexpected queueing delays
- Unexpected abortion
- …

Design: Span ExL Modeling & Inferencing

Modeling ExL of a Span: Verified in Real-world Case



There are Two parts of ExL:

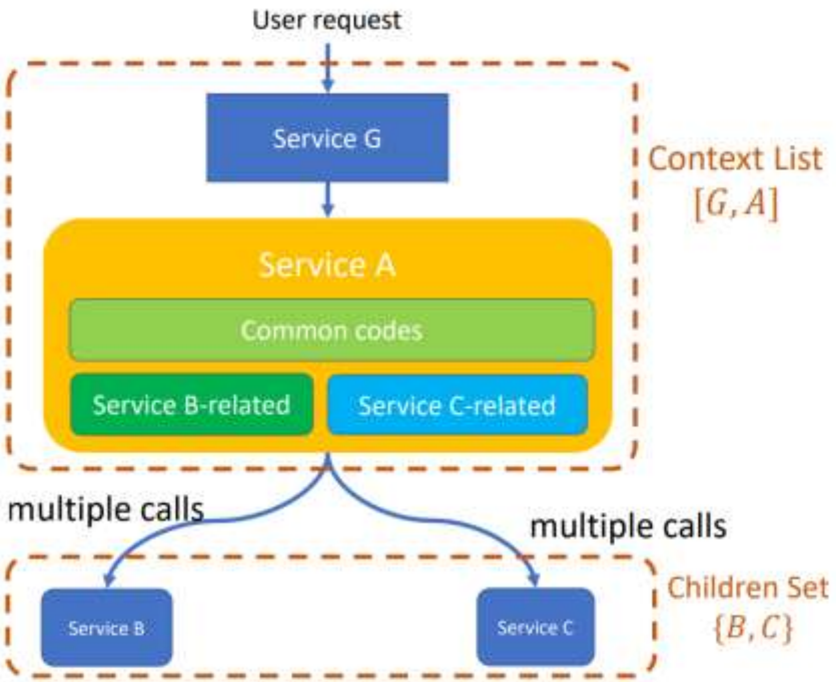
- n-related
- n-unrelated

Abnormal ExL indicates the abnormal service code execution:

- Unexpected code branch
- Unexpected queueing delays
- Unexpected early abortion
- …

Design: Span ExL Modeling & Inferencing

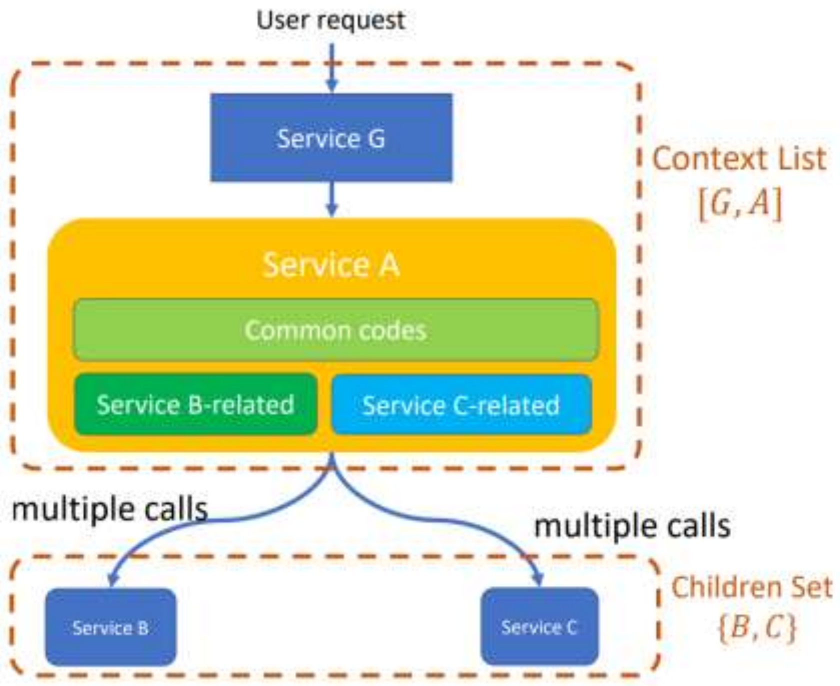
Modeling the ExL of span S_i



Design: Span ExL Modeling & Inferencing

Modeling the ExL of span S_i

$$\mathrm{ET}(S_i) = R(\theta(P(S_i))) + \mathbf{C}(\theta(P(S_i))) \cdot \mathbf{N}(S_i)$$

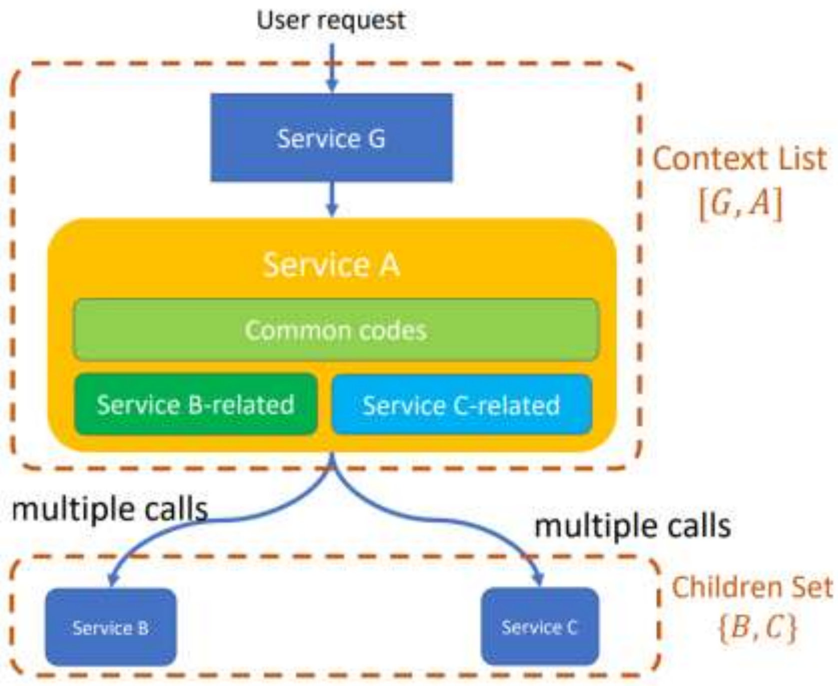


Design: Span ExL Modeling & Inferencing

Modeling the ExL of span S_i

$$\mathrm{ET}(S_i) = R(\theta(P(S_i))) + \mathbf{C}(\theta(P(S_i))) \cdot \mathbf{N}(S_i)$$

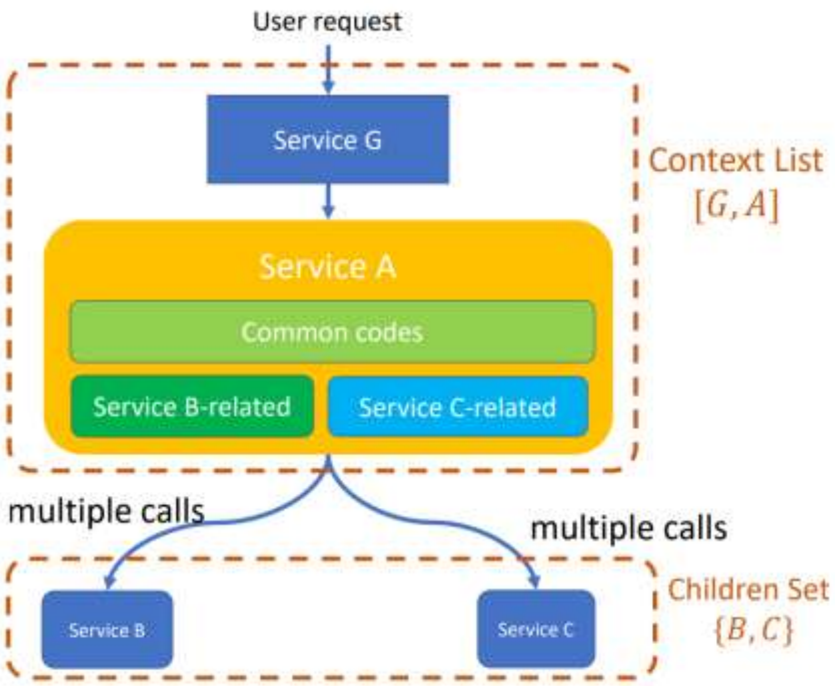
- $\mathrm{ET}(S_i)$: the Expected ExL distribution of span S_i
- $\mathrm{P}(S_i)$: the pattern of S_i, $\mathrm{P}(S_i) = ([G, A], \{B, C\})$
- R : the ExL components unrelated to the downstream call numbers
- $\mathbf{C}$: the ExL components related to the call numbers to each of the downstream nodes
- $\mathbf{N}(S_i)$: call numbers to each of the child microservices
- $\theta(\mathrm{P}(S_i))$: the pattern parameters (learned during training)



Design: Span ExL Modeling & Inferencing

Modeling the ExL of span S_i

$$\mathrm{ET}(S_i) = R(\theta(P(S_i))) + \mathbf{C}(\theta(P(S_i))) \cdot \mathbf{N}(S_i)$$



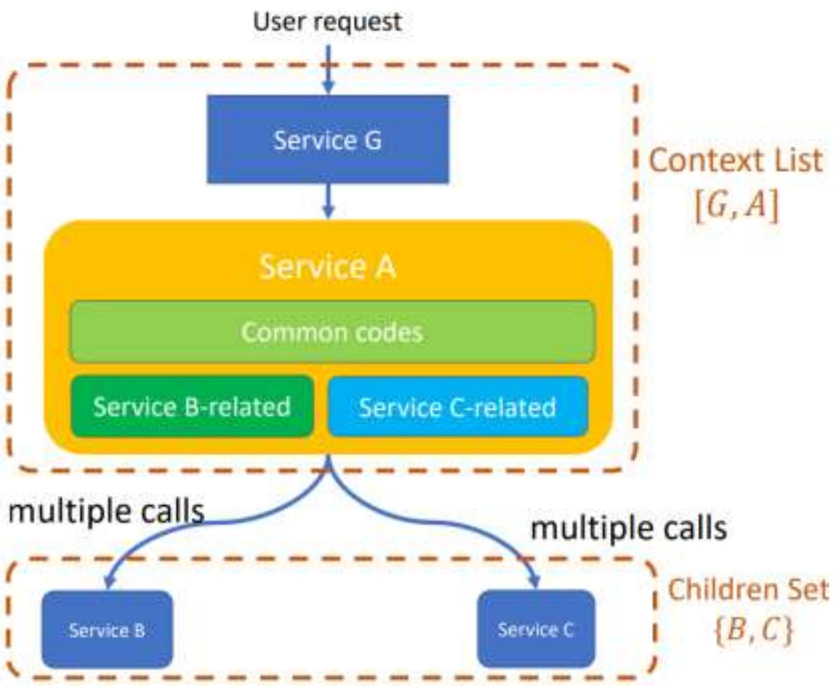
Design: Span ExL Modeling & Inferencing

Modeling the ExL of span S_i

$$\mathrm{ET}(S_i) = R(\theta(P(S_i))) + \mathbf{C}(\theta(P(S_i))) \cdot \mathbf{N}(S_i)$$

Gaussian Noise Assumption

$$R \sim \mathcal{N}(t_R(P(S_i))), \sigma_R^2(P(S_i)))$$
$$\mathbf{C} \sim \mathcal{N}(\mathbf{t_C}(P(S_i)), \sigma_{\mathbf{C}}^2(P(S_i)))$$



Design: Span ExL Modeling & Inferencing

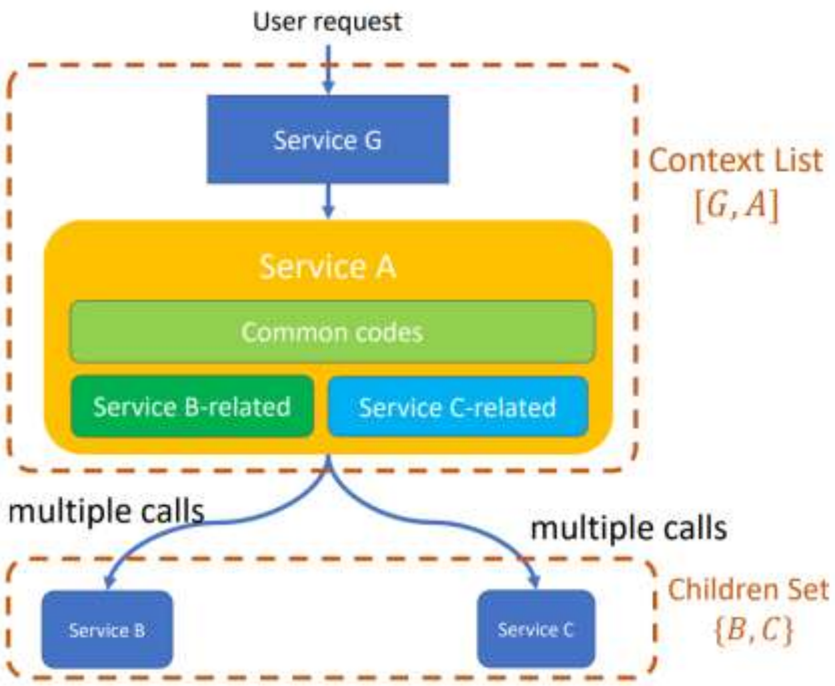
Modeling the ExL of span S_i

$$\mathrm{ET}(S_i) = R(\theta(P(S_i))) + \mathbf{C}(\theta(P(S_i))) \cdot \mathbf{N}(S_i)$$

Gaussian Noise Assumption

$$R \sim \mathcal{N}\left(\boxed{t_R(P(S_i))}, \sigma_R^2(P(S_i))\right)$$
$$\mathbf{C} \sim \mathcal{N}\left(\boxed{\mathbf{t_C}(P(S_i))}, \sigma_{\mathbf{C}}^2(P(S_i))\right)$$

Theoretically derived by LSM



Design: Span ExL Modeling & Inferencing

Modeling the ExL of span S_i

$$\mathrm{ET}(S_i) = R(\theta(P(S_i))) + \mathbf{C}(\theta(P(S_i))) \cdot \mathbf{N}(S_i)$$

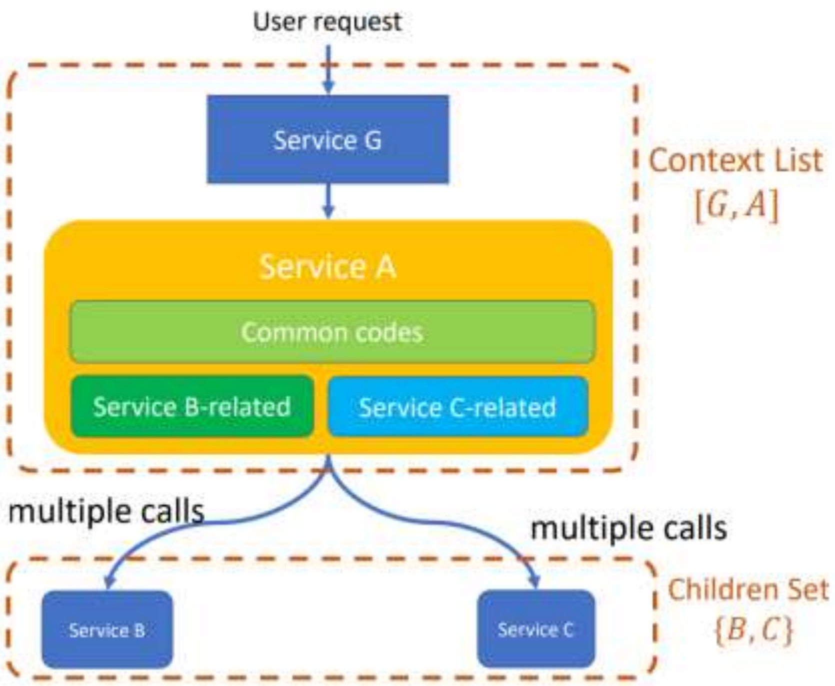
Gaussian Noise Assumption

$$R \sim \mathcal{N}\left(\boxed{t_R(P(S_i))}, \boxed{\sigma_R^2(P(S_i))}\right)$$

$$\mathbf{C} \sim \mathcal{N}\left(\boxed{\mathbf{t_C}(P(S_i))}, \boxed{\sigma_\mathbf{C}^2(P(S_i))}\right)$$

Theoretically derived by LSM

Approximated by EM



Design: Span ExL Modeling & Inferencing

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Design: Span ExL Modeling & Inferencing

known unknown

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Design: Span ExL Modeling & Inferencing

known unknown

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Learned Patterns $R(P(S_i))$ $C(P(S_i))$

Design: Span ExL Modeling & Inferencing

known unknown

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Learned Patterns $R(P(S_i))$ $C(P(S_i))$

Inference Stage

Design: Span ExL Modeling & Inferencing

known unknown

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Learned Patterns $R(P(S_i))$ $C(P(S_i))$

Inference Stage

Ideal:

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Design: Span ExL Modeling & Inferencing

known   unknown

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Learned Patterns $R(P(S_i))$ $C(P(S_i))$

Inference Stage

Ideal:

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Reality:

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Design: Span ExL Modeling & Inferencing

known unknown

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Learned Patterns $R(P(S_i))$ $C(P(S_i))$

Inference Stage

Ideal:

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Reality:

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Testing Environment

The environment is frequently updated

- New microservices and calls emerge frequently

Design: Span ExL Modeling & Inferencing

known unknown

Training Stage

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Learned Patterns $R(P(S_i))$ $C(P(S_i))$

Inference Stage

Reality:

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$

Testing Environment

The environment is frequently updated

- New microservices and calls emerge frequently

Modeling for Pattern Unrecorded

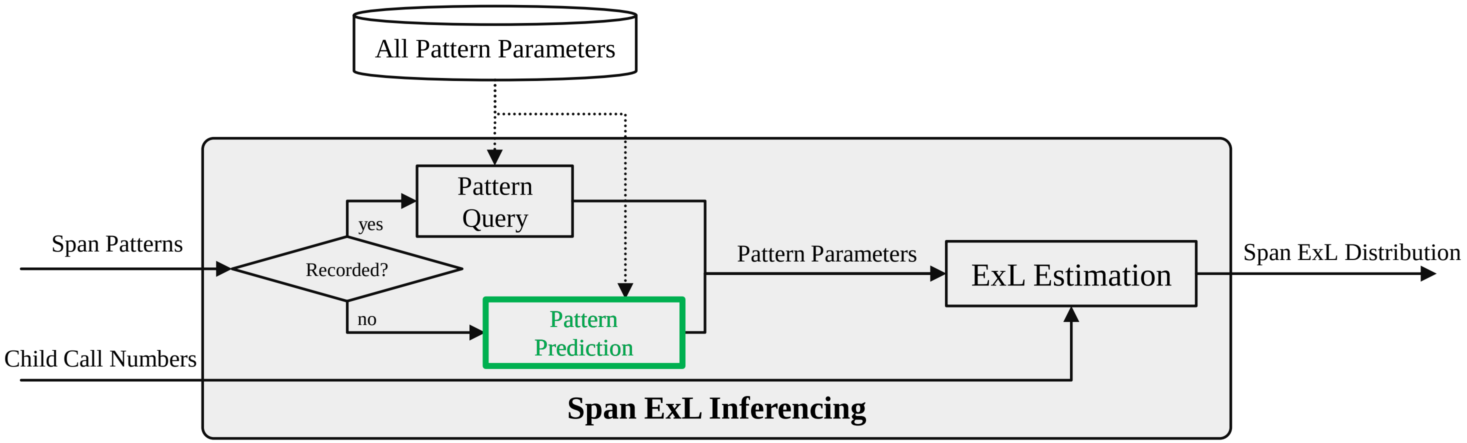
Design: Span ExL Modeling & Inferencing

known unknown

Training Stage

$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$

Learned Patterns $R(P(S_i))$ $C(P(S_i))$



Inference Stage

Reality:

$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$

Testing Environment

The environment is frequently updated

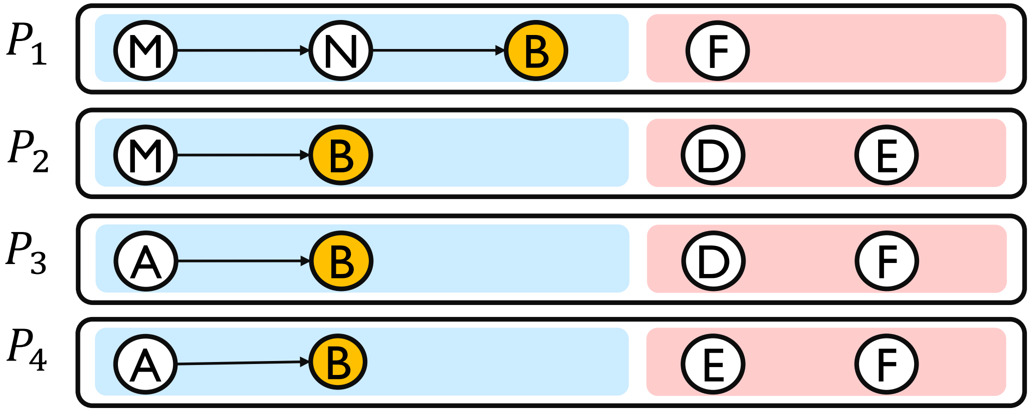
- New microservices and calls emerge frequently

Modeling for Pattern Unrecorded

Design: Span ExL Modeling & Inferencing

Predict Unseen Pattern Parameters

Pattern | **Context** | **Children** | **Known Parameters** | **Unknown Parameters**



Recorded Span Patterns and Parameters

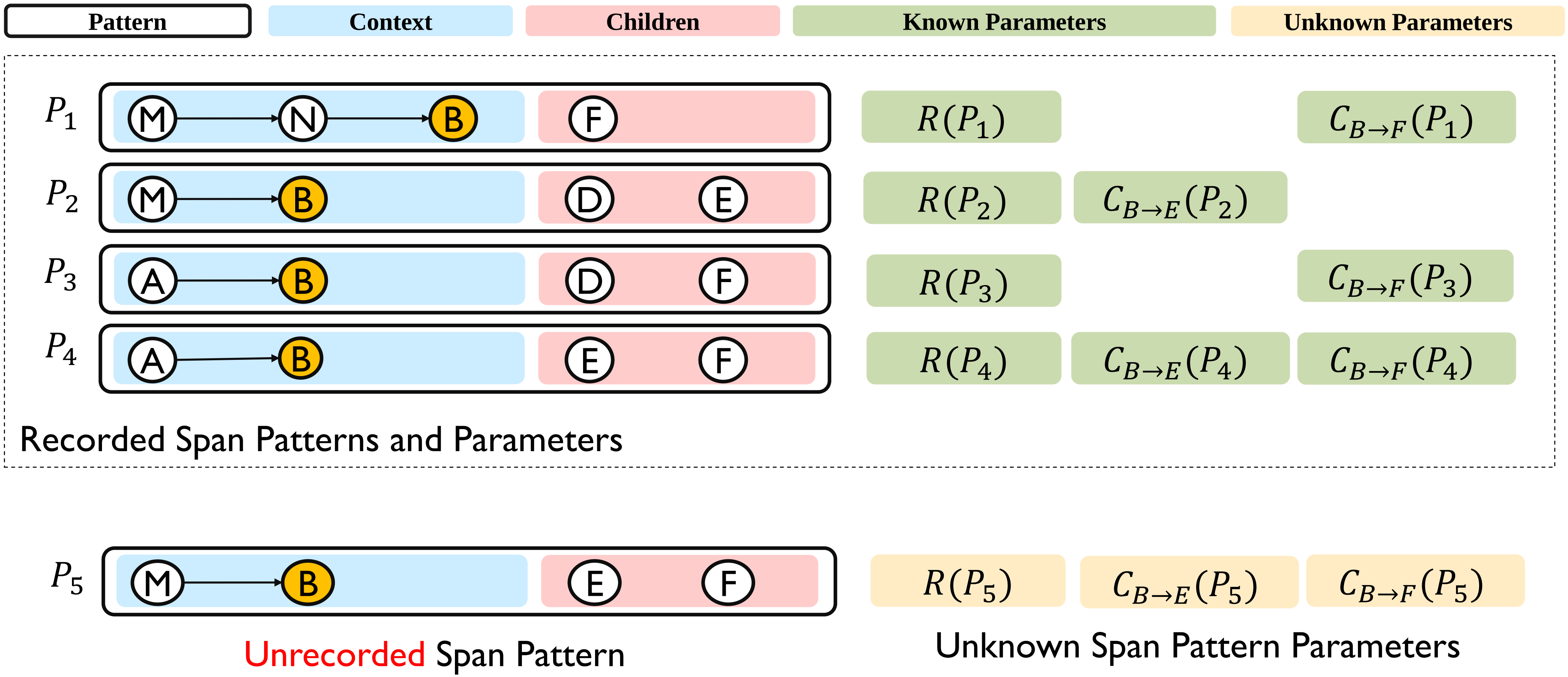
Unrecorded Span Pattern

$R(P_5)$ $C_{B\to E}(P_5)$ $C_{B\to F}(P_5)$

Unknown Span Pattern Parameters

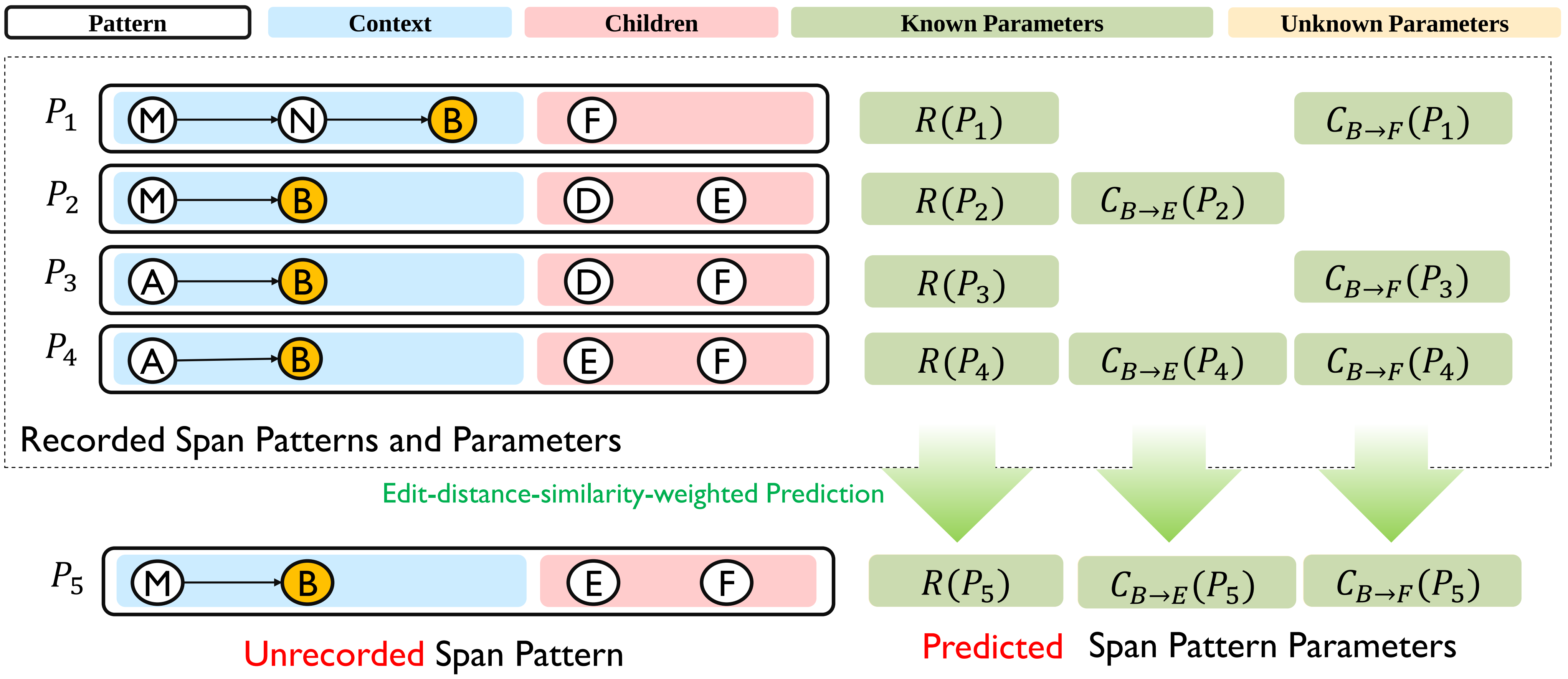
Design: Span ExL Modeling & Inferencing

Predict Unseen Pattern Parameters



Design: Span ExL Modeling & Inferencing

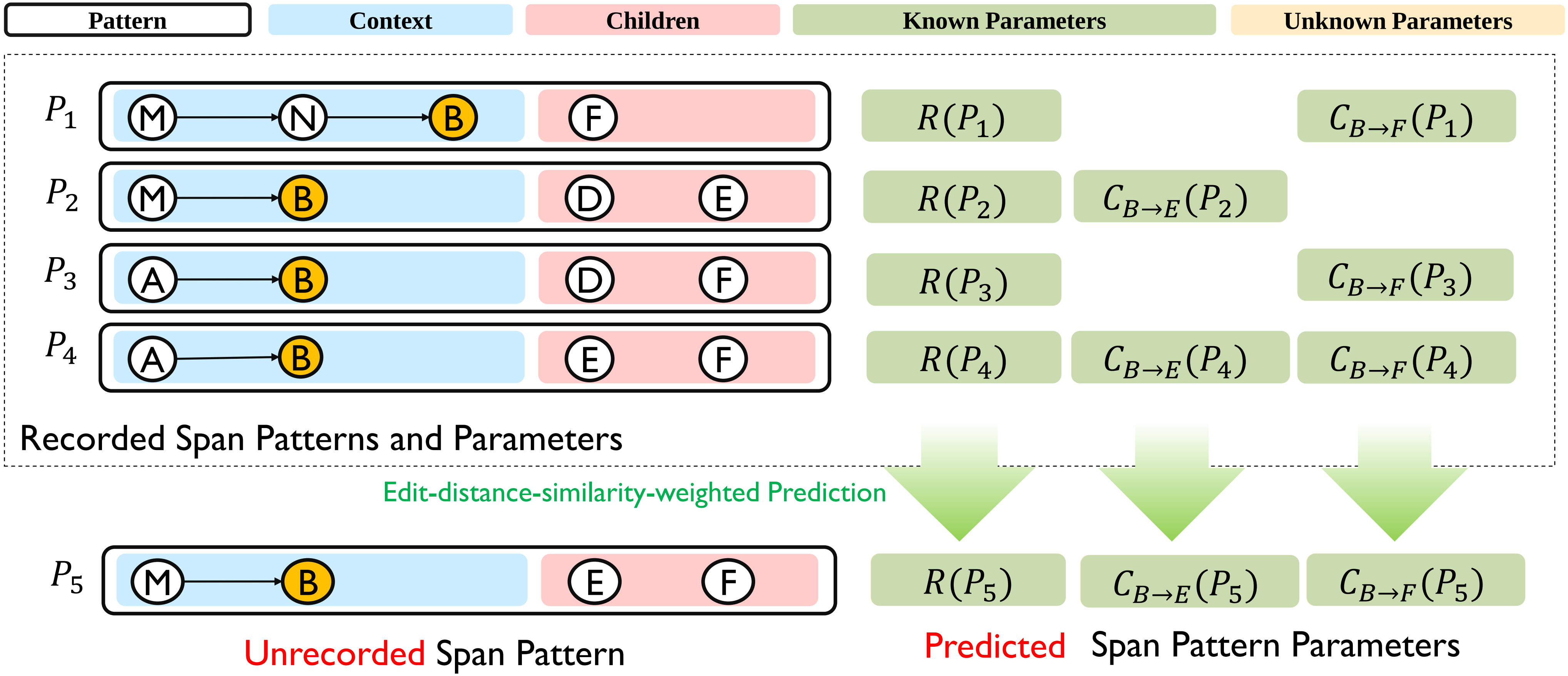
Predict Unseen Pattern Parameters



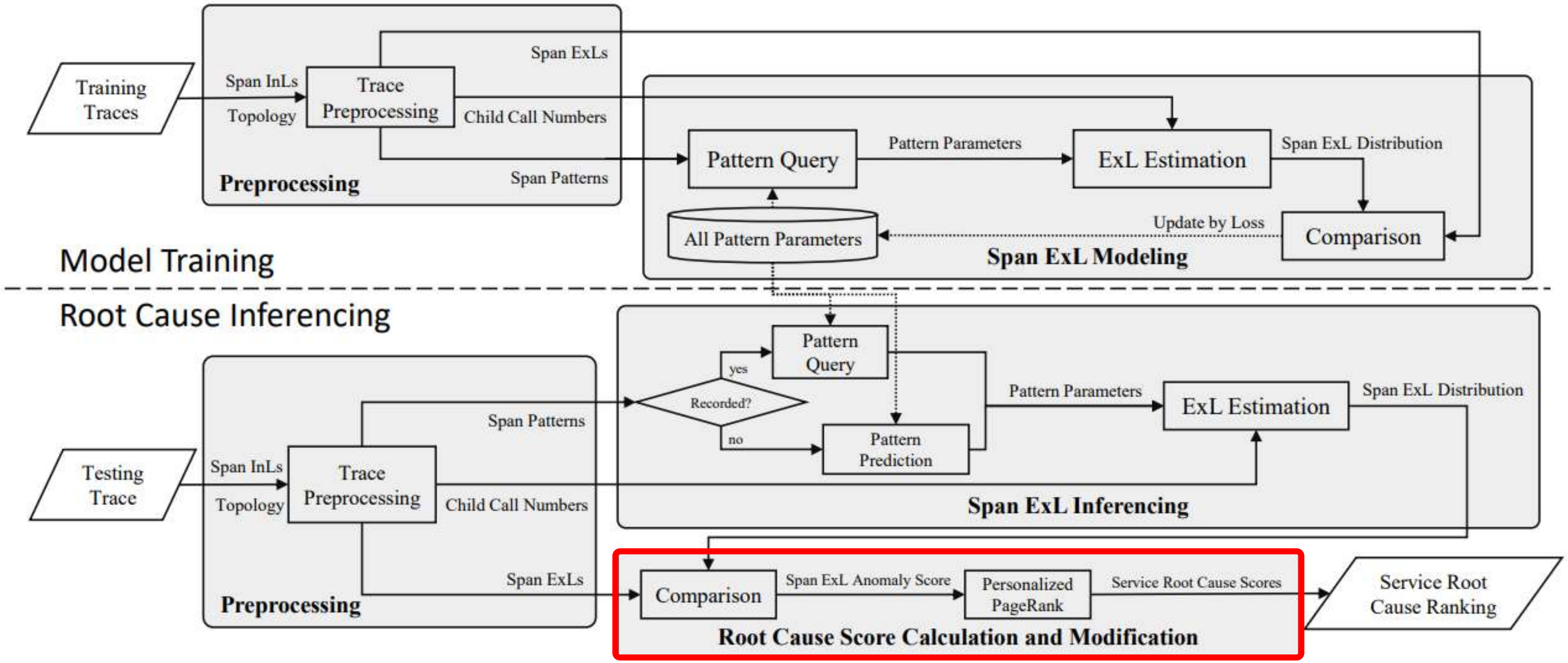
Design: Span ExL Modeling & Inferencing

Predict Unseen Pattern Parameters

$$ET(S_i) = R(P(S_i)) + C(P(S_i)) \cdot N(S_i)$$



Overview of SparseRCA



Workflow of SparseRCA

Design: Deriving and Optimizing Root Cause Scores

Naïve Root Cause Score

Design: Deriving and Optimizing Root Cause Scores

Naïve Root Cause Score

Standard Deviation ✖ Relative Deviation

$$\mathrm{Y}_{raw}(S_i) = \frac{(\mathrm{ET}(S_i) - \mu(S_i))^2}{(\mu(S_i) + \epsilon) * (\sigma(S_i) + \epsilon)}$$

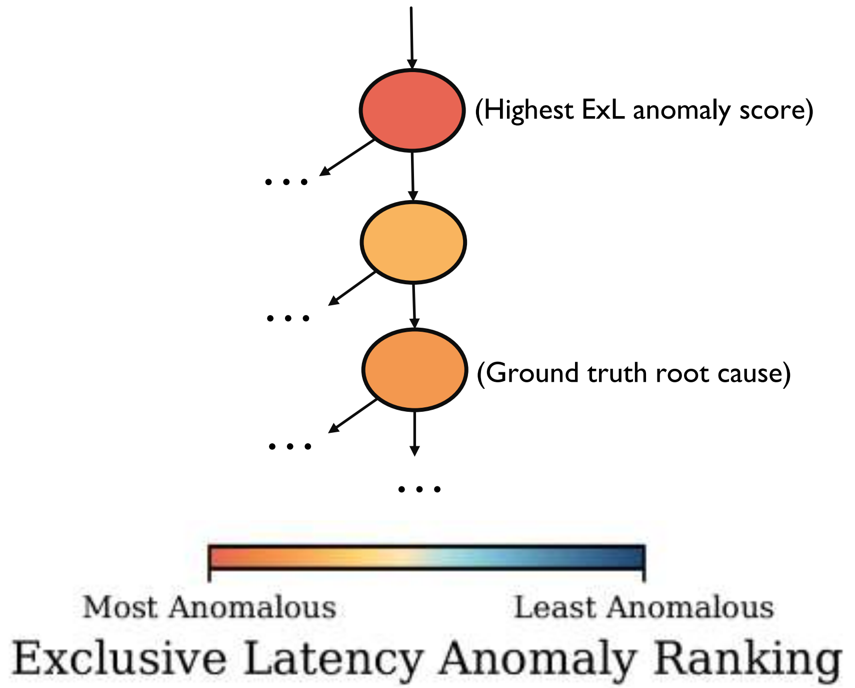
Design: Deriving and Optimizing Root Cause Scores

Naïve Root Cause Score

Standard Deviation ✖ Relative Deviation

$$Y_{raw}(S_i) = \frac{(\mathrm{ET}(S_i) - \mu(S_i))^2}{(\mu(S_i) + \epsilon) * (\sigma(S_i) + \epsilon)}$$

RCA cannot fully depend on ExL anomalies because anomalous high ExLs may also appear in ancestor nodes



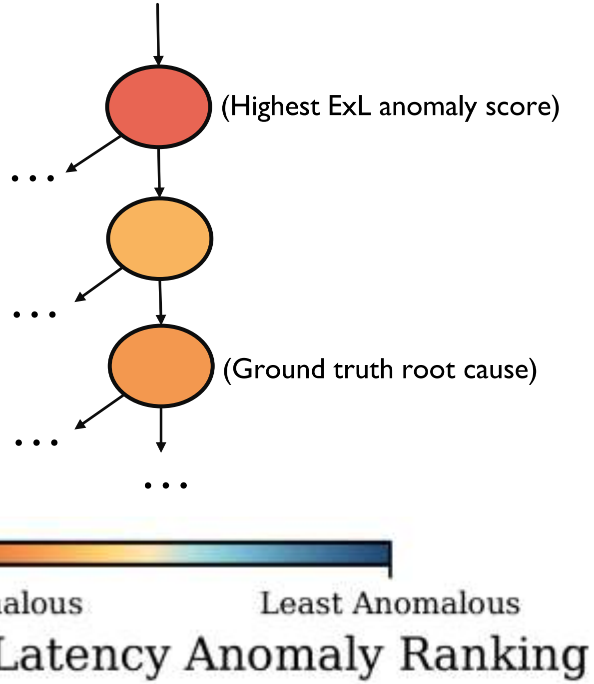
Design: Deriving and Optimizing Root Cause Scores

Naïve Root Cause Score

Standard Deviation ✖ Relative Deviation

$$Y_{raw}(S_i) = \frac{(\mathrm{ET}(S_i) - \mu(S_i))^2}{(\mu(S_i) + \epsilon) * (\sigma(S_i) + \epsilon)}$$

RCA cannot fully depend on ExL anomalies because anomalous high ExLs may also appear in ancestor nodes



Root Cause Score Modification

Personalized PageRank — Redistribute the root cause scores

$$\mathbf{Y}_{mod}^{(t+1)}(\mathbf{S}) = \alpha \mathbf{Y}_{raw}(\mathbf{S}) + (1-\alpha)\mathbf{M} \times \mathbf{Y}_{mod}^{(t)}(\mathbf{S})$$

$$\mathbf{Y}_{mod}^{(0)}(\mathbf{S}) = \mathbf{Y}_{raw}(\mathbf{S})$$

$$\mathbf{Y}_{final}(\mathbf{S}) = \mathbf{Y}_{mod}^{(L)}(\mathbf{S})$$

Design: Deriving and Optimizing Root Cause Scores

Naïve Root Cause Score

Standard Deviation ✖ Relative Deviation

$$Y_{raw}(S_i) = \frac{(\mathrm{ET}(S_i) - \mu(S_i))^2}{(\mu(S_i) + \epsilon) * (\sigma(S_i) + \epsilon)}$$

Root Cause Score Modification

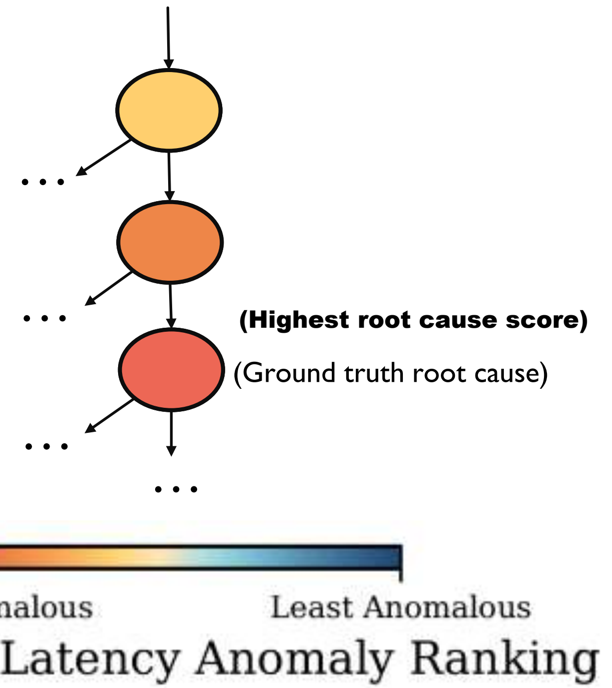
Personalized PageRank — Redistribute the root cause scores

$$\mathbf{Y}_{mod}^{(t+1)}(\mathbf{S}) = \alpha \mathbf{Y}_{raw}(\mathbf{S}) + (1-\alpha)\mathbf{M} \times \mathbf{Y}_{mod}^{(t)}(\mathbf{S})$$

$$\mathbf{Y}_{mod}^{(0)}(\mathbf{S}) = \mathbf{Y}_{raw}(\mathbf{S})$$

$$\mathbf{Y}_{final}(\mathbf{S}) = \mathbf{Y}_{mod}^{(L)}(\mathbf{S})$$

RCA cannot fully depend on ExL anomalies because anomalous high ExLs may also appear in ancestor nodes



Outline

- Background
- Design
- Evaluation
- Summary

Evaluation Research Questions

RQ1:Effectiveness

How does SparseRCA improve the accuracy of RCA in real-world TE dataset?

RQ2:Ablation Study

What is the individual and combined contribution of some designs in SparseRCA?

RQ3:Trace Sparsity Robustness

How does our model perform with even sparser traces?

Evaluation Dataset

Experiment Dataset

- Collected from the real-world testing environment in a datacenter in Hangzhou of Ant Group
- 6k+ traces (SLO-satisfied and SLO-violated) as the training set
- The root cause of 120 SLO-violated traces manually labeled serve as the test set
- The dataset covers 29 days and involves 500+ services

#Traces (train)	#Traces (test)	Overall (Duration)	#Services	AvgSrv (per trace)	AvgSpn (per trace)
6,080	120	29 days	507	10.6	38.7

Dataset Overview

Evaluation RQ1: Effectiveness

Evaluation Metric

- Average Expected Top-k accuracy
- Tie-breaking with probability strategy

$$a_i = \begin{cases} min(m, k-n+1)/m & \text{if } n \leq k \\ 0 & \text{if } n > k \end{cases}$$

$$\text{A@k} = \frac{1}{T}\sum_{i=1}^{T} a_i \times 100\%$$

Comparison with Baselines

TABLE IV: Performance of SparseRCA and Baselines

	Model	Category	A@1 (%)	A@3 (%)	A@5 (%)
Baselines	MicroHECL	Stat-based Topology	19.3	26.4	39.5
	AutoMap	Stat-based Causality	40.7	50.6	61.5
	MicroScope	Stat-based Causality	40.3	66.3	73.0
	MicroRank	Trace InL & Spectrum	61.2	67.6	73.0
Ours	SparseRCA	Trace ExL & Topology	**66.1**	**86.4**	**88.1**

Evaluation RQ1: Effectiveness

Evaluation Metric

- Average Expected Top-k accuracy
- Tie-breaking with probability strategy

$$a_i = \begin{cases} min(m, k-n+1)/m & \text{if } n \leq k \\ 0 & \text{if } n > k \end{cases}$$

$$\text{A@k} = \frac{1}{T}\sum_{i=1}^{T} a_i \times 100\%$$

Comparison with Baselines

TABLE IV: Performance of SparseRCA and Baselines

	Model	Category	A@1 (%)	A@3 (%)	A@5 (%)
Baselines	MicroHECL	Stat-based Topology	19.3	26.4	39.5
	AutoMap	Stat-based Causality	40.7	50.6	61.5
	MicroScope	Stat-based Causality	40.3	66.3	73.0
	MicroRank	Trace InL & Spectrum	61.2	67.6	73.0
Ours	SparseRCA	Trace ExL & Topology	**66.1**	**86.4**	**88.1**

SparseRCA outperforms baselines in TE

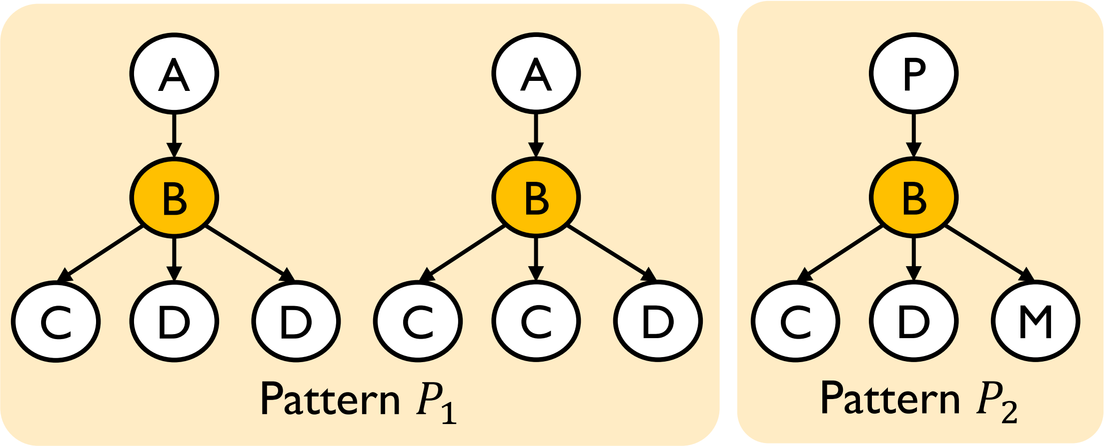
Evaluation RQ2: Ablation Study

Analyzed Components

Evaluation RQ2: Ablation Study

Analyzed Components

PBM The Pattern-Based Modeling of the span ExLs (instead of the call-based)

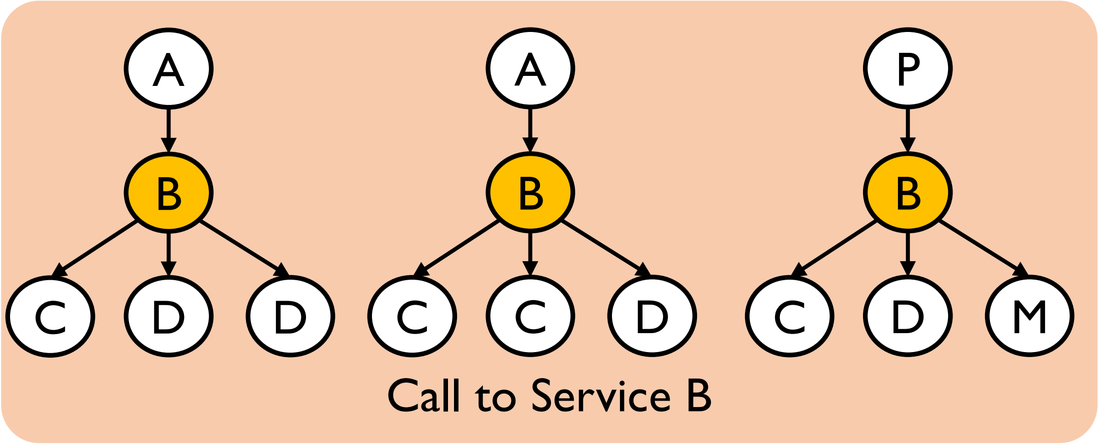


PBM **Distinguished Patterns**

Evaluation RQ2: Ablation Study

Analyzed Components

PBM The Pattern-Based Modeling of the span ExLs (instead of the call-based)



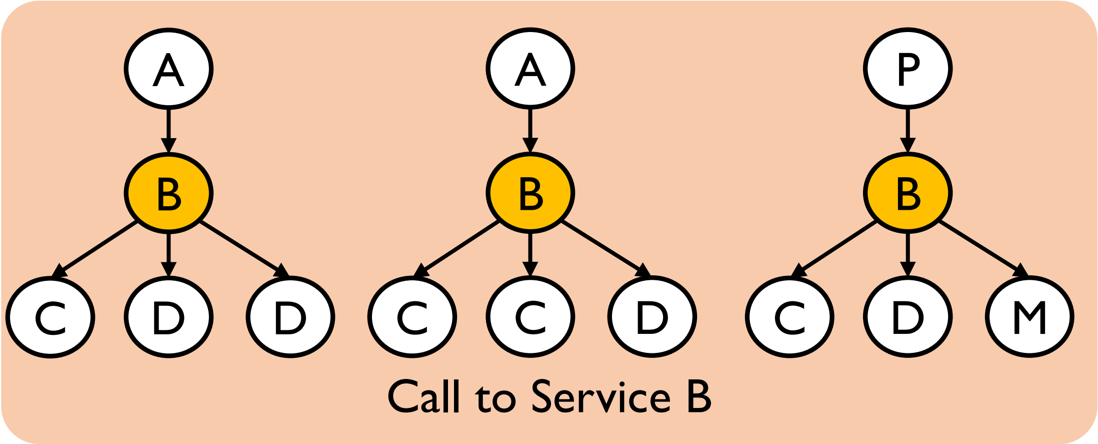
~~**PBM**~~ **Same Calls**

Evaluation RQ2: Ablation Study

Analyzed Components

PBM The Pattern-Based Modeling of the span ExLs (instead of the call-based)

DBA The Distribution-Based Anomaly score of ExL (instead of expectation-based)



$$R \sim \mathcal{N}(t_R(P(S_i))), \sigma_R^2(P(S_i)))$$
$$\mathbf{C} \sim \mathcal{N}(\mathbf{t_C}(P(S_i)), \sigma_{\mathbf{C}}^{\mathbf{2}}(P(S_i)))$$

$$\mathrm{Y}_{raw}(S_i) = \frac{(\mathrm{ET}(S_i) - \mu(S_i))^2}{(\mu(S_i) + \epsilon) * (\sigma(S_i) + \epsilon)}$$

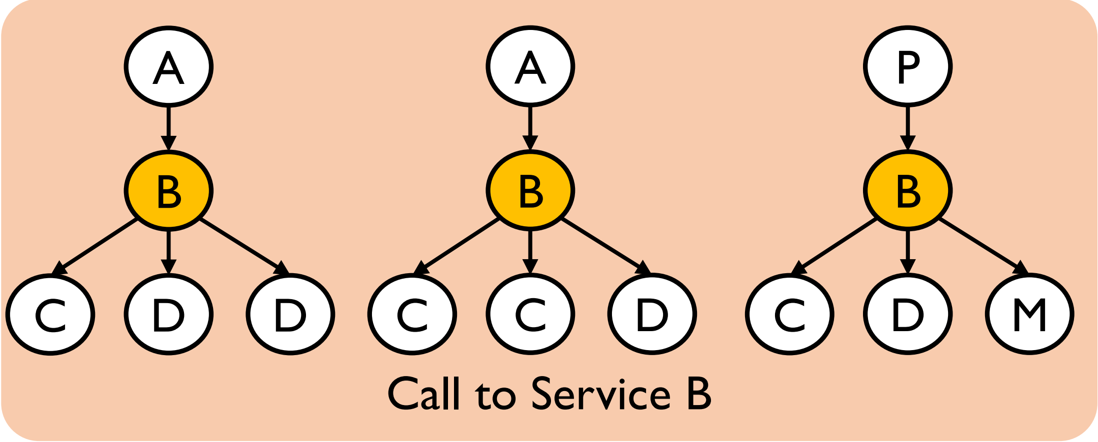
~~PBM~~ **Same Calls** **DBA**

Evaluation RQ2: Ablation Study

Analyzed Components

PBM The Pattern-Based Modeling of the span ExLs (instead of the call-based)

DBA The Distribution-Based Anomaly score of ExL (instead of expectation-based)



$$R \sim \mathcal{N}(t_R(P(S_i))), \sigma_R^2(\cancel{P}(S_i)))$$

$$\mathbf{C} \sim \mathcal{N}(\mathbf{t_C}(P(S_i)), \sigma_{\mathbf{C}}^2(\cancel{P}(S_i)))$$

$$Y'_{raw}(S_i) = \frac{|\mathrm{ET}(S_i) - \mu(S_i)|}{\mu(S_i) + \epsilon}$$

~~**PBM**~~ **Same Calls** ~~**DBA**~~

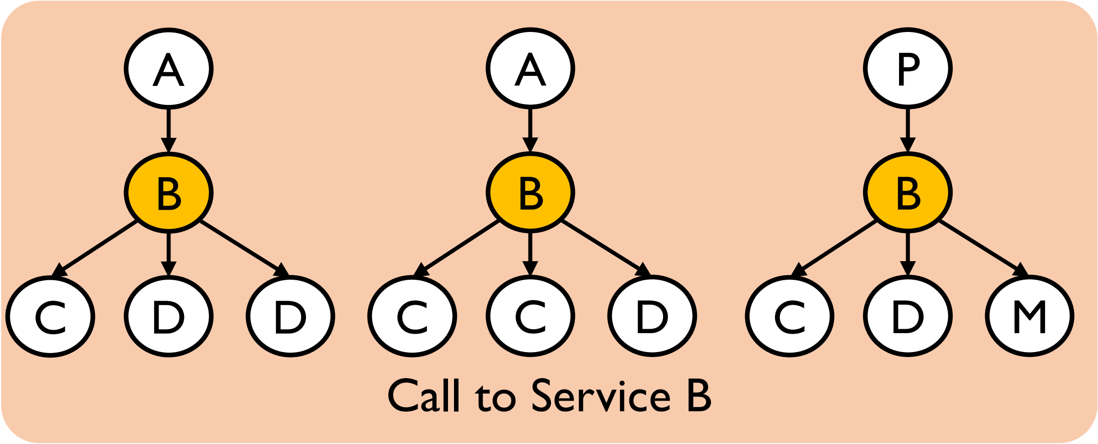
Evaluation RQ2: Ablation Study

Analyzed Components

PBM The Pattern-Based Modeling of the span ExLs (instead of the call-based)

DBA The Distribution-Based Anomaly score of ExL (instead of expectation-based)

RCM The topology-based Root Cause Modification through personalized PageRank



~~PBM~~

Same Calls

$$R \sim \mathcal{N}(t_R(P(S_i))), \sigma_R^2(\cancel{P}(S_i)))$$

$$\mathbf{C} \sim \mathcal{N}(\mathbf{t_C}(P(S_i)), \sigma_{\mathbf{C}}^2(\cancel{P}(S_i)))$$

$$Y'_{raw}(S_i) = \frac{|\mathrm{ET}(S_i) - \mu(S_i)|}{\mu(S_i) + \epsilon}$$

~~DBA~~

$$\mathbf{Y}_{mod}^{(t+1)}(\mathbf{S}) = \alpha \mathbf{Y}_{raw}(\mathbf{S}) + (1-\alpha)\mathbf{M} \times \mathbf{Y}_{mod}^{(t)}(\mathbf{S})$$
$$\mathbf{Y}_{mod}^{(0)}(\mathbf{S}) = \mathbf{Y}_{raw}(\mathbf{S})$$
$$\mathbf{Y}_{final}(\mathbf{S}) = \mathbf{Y}_{mod}^{(L)}(\mathbf{S})$$

RCM

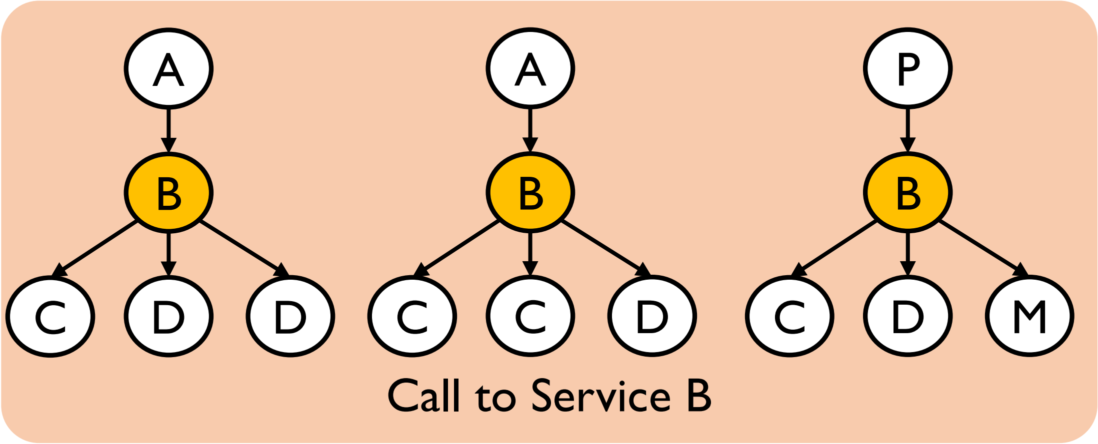
Evaluation RQ2: Ablation Study

Analyzed Components

PBM The Pattern-Based Modeling of the span ExLs (instead of the call-based)

DBA The Distribution-Based Anomaly score of ExL (instead of expectation-based)

RCM The topology-based Root Cause Modification through personalized PageRank



PBM

Same Calls

$$R \sim \mathcal{N}(t_R(P(S_i))), \sigma_R^2(P(S_i)))$$
$$\mathbf{C} \sim \mathcal{N}(\mathbf{t_C}(P(S_i)), \sigma_{\mathbf{C}}^2(P(S_i)))$$

$$Y'_{raw}(S_i) = \frac{|\mathrm{ET}(S_i) - \mu(S_i)|}{\mu(S_i) + \epsilon}$$

DBA

$$\mathbf{Y}^{(t+1)}_{mod}(\mathbf{S}) = \alpha \mathbf{Y}_{raw}(\mathbf{S}) + (1-\alpha)\mathbf{M} \times \mathbf{Y}^{(t)}_{mod}(\mathbf{S})$$
$$\mathbf{Y}^{(0)}_{mod}(\mathbf{S}) = \mathbf{Y}_{raw}(\mathbf{S})$$
$$\mathbf{Y}_{final}(\mathbf{S}) = \mathbf{Y}^{(\infty)}_{mod}(\mathbf{S})$$

RCM

Evaluation RQ2: Ablation Study

Experiment Results

	Model	PBM	DBA	RCM	A@1	A@3	A@5
Complete	SparseRCA	✓	✓	✓	**66.1**	**86.4**	**88.1**
	w/o (RCM)	✓	✓	×	49.2	72.9	72.9
	w/o (DBA)	✓	×	✓	59.3	81.4	84.7
	w/o (PBM)	×	✓	✓	61.0	84.7	88.1
Partial	w/o (DBA,RCM)	✓	×	×	42.4	69.5	72.9
	w/o (PBM,DBA)	×	✓	×	44.1	69.5	74.6
	w/o (PBM,DBA)	×	×	✓	47.5	72.9	84.7
	w/o (PBM,DBA,RCM)	×	×	×	27.1	61.0	72.9

Evaluation RQ2: Ablation Study

Experiment Results

	Model	PBM	DBA	RCM	A@1	A@3	A@5
Complete	SparseRCA	✓	✓	✓	**66.1**	**86.4**	**88.1**
Partial	w/o (RCM)	✓	✓	×	49.2	72.9	72.9
	w/o (DBA)	✓	×	✓	59.3	81.4	84.7
	w/o (PBM)	×	✓	✓	61.0	84.7	88.1
	w/o (DBA,RCM)	✓	×	×	42.4	69.5	72.9
	w/o (PBM,DBA)	×	✓	×	44.1	69.5	74.6
	w/o (PBM,DBA)	×	×	✓	47.5	72.9	84.7
	w/o (PBM,DBA,RCM)	×	×	×	27.1	61.0	72.9

Remove from complete model

Add to partial model

Evaluation RQ2: Ablation Study

Experiment Results

	Model	PBM	DBA	RCM	A@1	A@3	A@5
Complete	SparseRCA	✓	✓	✓	**66.1**	**86.4**	**88.1**
	w/o (RCM)	✓	✓	×	49.2	72.9	72.9
	w/o (DBA)	✓	×	✓	59.3	81.4	84.7
	w/o (PBM)	×	✓	✓	61.0	84.7	88.1
Partial	w/o (DBA,RCM)	✓	×	×	42.4	69.5	72.9
	w/o (PBM,DBA)	×	✓	×	44.1	69.5	74.6
	w/o (PBM,DBA)	×	×	✓	47.5	72.9	84.7
	w/o (PBM,DBA,RCM)	×	×	×	27.1	61.0	72.9

Remove from complete model

Add to partial model

The designs are all proved effective

Evaluation RQ2: Ablation Study

Experiment Results

	Model	PBM	DBA	RCM	A@1	A@3	A@5
Complete	SparseRCA	✓	✓	✓	**66.1**	**86.4**	**88.1**
Partial	w/o (RCM)	✓	✓	×	49.2	72.9	72.9
	w/o (DBA)	✓	×	✓	59.3	81.4	84.7
	w/o (PBM)	×	✓	✓	61.0	84.7	88.1
	w/o (DBA,RCM)	✓	×	×	42.4	69.5	72.9
	w/o (PBM,DBA)	×	✓	×	44.1	69.5	74.6
	w/o (PBM,DBA)	×	×	✓	47.5	72.9	84.7
	w/o (PBM,DBA,RCM)	×	×	×	27.1	61.0	72.9

Remove from complete model

Add to partial model

The designs are all proved effective

The RCM improves the performance most significantly

Evaluation RQ2: Ablation Study

Experiment Results

	Model	PBM	DBA	RCM	A@1	A@3	A@5
Complete	SparseRCA	✓	✓	✓	**66.1**	**86.4**	**88.1**
Partial	w/o (RCM)	✓	✓	×	49.2	72.9	72.9
	w/o (DBA)	✓	×	✓	59.3	81.4	84.7
	w/o (PBM)	×	✓	✓	61.0	84.7	88.1
	w/o (DBA,RCM)	✓	×	×	42.4	69.5	72.9
	w/o (PBM,DBA)	×	✓	×	44.1	69.5	74.6
	w/o (PBM,DBA)	×	×	✓	47.5	72.9	84.7
	w/o (PBM,DBA,RCM)	×	×	×	27.1	61.0	72.9

Remove from complete model

Add to partial model

The designs are all proved effective

The RCM improves the performance most significantly

While our ExL modeling can roughly identify root causes (A@5), the analyzed components can effectively optimize the ranking within the highly suspicious root cause candidates

Evaluation RQ2: Ablation Study

Experiment Results

	Model	PBM	DBA	RCM	A@1	A@3	A@5
Complete	SparseRCA	✓	✓	✓	**66.1**	**86.4**	**88.1**
Partial	w/o (RCM)	✓	✓	×	49.2	72.9	72.9
	w/o (DBA)	✓	×	✓	59.3	81.4	84.7
	w/o (PBM)	×	✓	✓	61.0	84.7	88.1
	w/o (DBA,RCM)	✓	×	×	42.4	69.5	72.9
	w/o (PBM,DBA)	×	✓	×	44.1	69.5	74.6
	w/o (PBM,DBA)	×	×	✓	47.5	72.9	84.7
	w/o (PBM,DBA,RCM)	×	×	×	27.1	61.0	72.9

Remove from complete model

Add to partial model

The designs are all proved effective

The RCM improves the performance most significantly

While our ExL modeling can roughly identify root causes (A@5), the analyzed components can effectively optimize the ranking within the highly suspicious root cause candidates

Evaluation RQ3: Trace Sparsity Evaluation

With Sparser Traces

TABLE VI: Accuracy of SparseRCA Under Sparser Traces.

trainset used (%)	Model	A@1	A@3	A@5
100	MicroRank	61.2	67.6	73.0
100	SparseRCA	66.1	86.4	88.1
50	SparseRCA	66.1	78.0	84.7
40	SparseRCA	66.1	79.7	83.1
25	SparseRCA	55.9	72.9	79.7
20	SparseRCA	59.3	71.2	79.7
15	SparseRCA	54.2	69.5	78.0
10	SparseRCA	54.2	66.1	74.6
5	SparseRCA	42.4	52.5	69.5

Evaluation RQ3: Trace Sparsity Evaluation

With Sparser Traces

TABLE VI: Accuracy of SparseRCA Under Sparser Traces.

trainset used (%)	Model	A@1	A@3	A@5
100	MicroRank	61.2	67.6	73.0
100	SparseRCA	66.1	86.4	88.1
50	SparseRCA	66.1	78.0	84.7
40	SparseRCA	66.1	79.7	83.1
25	SparseRCA	55.9	72.9	79.7
20	SparseRCA	59.3	71.2	79.7
15	SparseRCA	54.2	69.5	78.0
10	SparseRCA	54.2	66.1	74.6
5	SparseRCA	42.4	52.5	69.5

Evaluation RQ3: Trace Sparsity Evaluation

With Sparser Traces

TABLE VI: Accuracy of SparseRCA Under Sparser Traces.

trainset used (%)	Model	A@1	A@3	A@5
100	MicroRank	61.2	67.6	73.0
100	SparseRCA	66.1	86.4	88.1
50	SparseRCA	66.1	78.0	84.7
40	SparseRCA	66.1	79.7	83.1
25	SparseRCA	55.9	72.9	79.7
20	SparseRCA	59.3	71.2	79.7
15	SparseRCA	54.2	69.5	78.0
10	SparseRCA	54.2	66.1	74.6
5	SparseRCA	42.4	52.5	69.5

SparseRCA is more robust and tolerate in scenarios with even sparser traces

Summary

SparseRCA

- The trace sparsity and system volatility in testing environments
- An unsupervised trace-based RCA method performing robust single-trace inference and capable of processing unseen span patterns

High top-k accuracy of RCA evaluated with real-world dataset

Key Designs of SparseRCA

- Span-pattern-based modeling of span ExL distributions
- Predicting unseen patterns with pattern similarity
- Topology-based Optimization with personalized PageRank

Proved effectiveness of the key components in ablation study

清華大學
Tsinghua University

中国科学院
计算机网络信息中心
Computer Network Information Center,
Chinese Academy of Sciences

蚂蚁集团
ANT GROUP

Thank You!

SparseRCA: Unsupervised Root Cause Analysis in Sparse Microservice Testing Traces

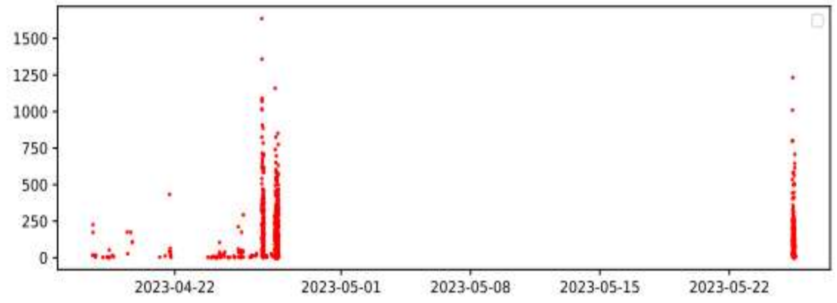
Zhenhe Yao[1], Haowei Ye, Changhua Pei[2], Guang Cheng, Guangpei Wang, Zhiwei Liu,
Hongwei Chen, Hang Cui, Zeyan Li, Jianhui Li, Gaogang Xie, Dan Pei

Q: Why do you model the ExL with linear gaussian assumption?

Trace Sparsity in Testing Environment limits the training of some DL-based methods

- VAE
- GNN
- …

Non-convergence



#Trace/min	0	1	[2,10)	[10,30)	[30,50)	50+
Minute Pct. (%)	89.97	3.95	3.85	1.50	0.53	0.19
Abnormal Pct. (%)	/	11.11	8.01	6.16	7.12	2.04

Testing Environment

Traces from manually constructed test cases

- Sparse traces (mostly less than 10/min at peak)
- Centralized in testing period

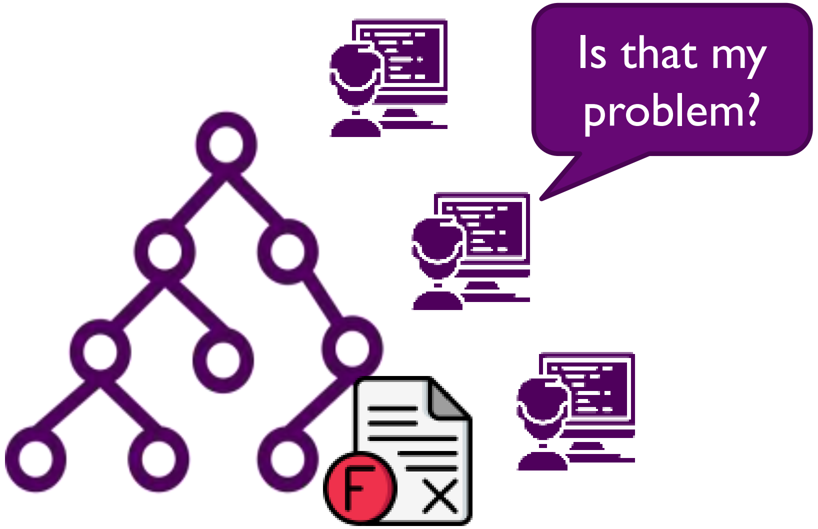
Q: Can SparseRCA be applied to trace-dense scenarios?

1. **SparseRCA addresses the challenges for sparse scenarios**
   - Theoretically, SparseRCA can be applied dense scenario. The design insights holds for trace-dense scenarios.
   - But in dense environment where more data are available and some important weaknesses are not strong, the performance improvement by SparseRCA might not be as significant as in sparse environment.

2. **Some of the implementations could be altered with larger data input**
   - With large volumes of data, we usually train the models batch by batch, avoiding processing the data together.
   - Some of the SparseRCA designs could be altered if being applied to dense scenario. E.g., we could:
     - Utilize complex models like VAE to derive the variance
     - Introduce mixed distributions

Q: How does SparseRCA compare with statistical code analysis tasks?

- **SparseRCA focuses the end-to-end testing scenarios**
  - Multiple teams/developers maintain different microservices, with little knowledge about the other teams' code details.
  - It's hard for a developer to perform code analysis for his upstream or downstream service



Q: Why do you choose the best metric instead of utilizing all together?

- **Sparsity introduces compromises**
  - In scenarios with sparse samples, simply introducing more features as new input channels usually results in worse performance.
    - Model easily overfits to bad RCA indicator metric
  - General machine learning principle
  - Incorporating additional metrics would be preferred for scenarios with dense traces