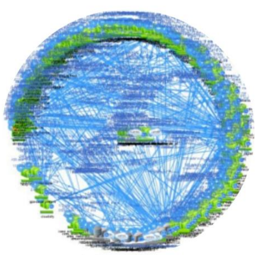
Sieve: Attention-based Sampling of End-to-End Trace Data in Distributed Microservice Systems

Zicheng Huang, Pengfei Chen, Guangba Yu, Hongyang Chen, Zibin Zheng

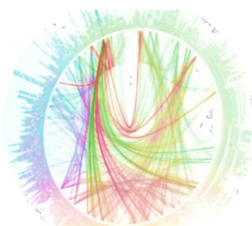
Distributed Tracing

Microservice architecture

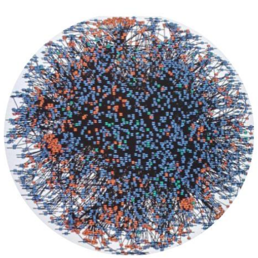
- loosely coupled
- fine-grained services
- complex interactions



Netflix



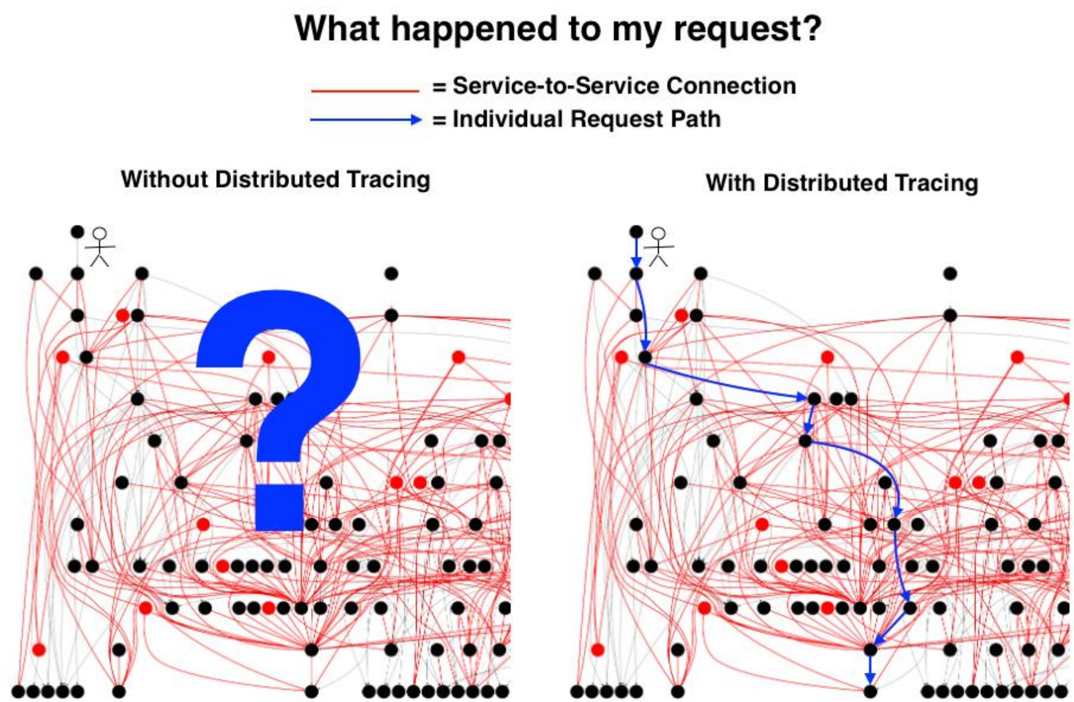
Twitter



Amazon

Social Network

Distributed request tracing



Plays an important role in profiling, diagnosing, and debugging microservice systems.

Trace Data Sampling

Trace data are often produced in a large volume and costly for storage.

$10^{10}\sim10^{11}$ reqeusts

dozens of TB trace data per day

Sample the trace data to reduce the storage overhead.

Head-based Sampling

- make sampling decision at the beginning of the trace
- reduce tracing overhead
- trace are sampled randomly

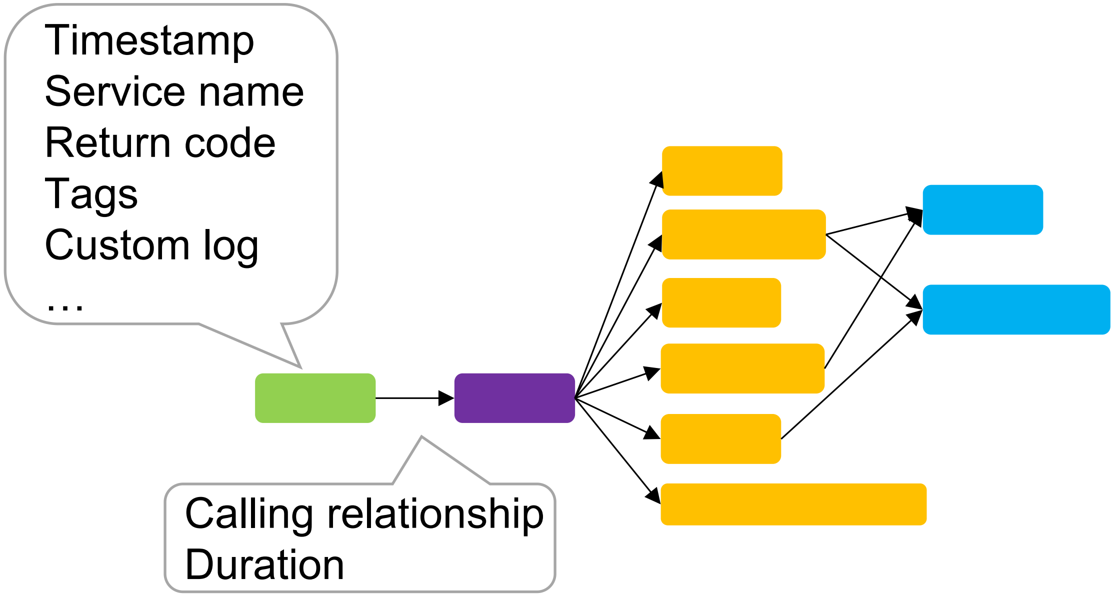
Tail-based Sampling

- make sampling decision when request finishes
- more overhead to generate the completed trace
- capture more informative traces

Challenges to Trace Sampling

Trace data contain rich and complex information.

- Trace is composed of numerous spans.
- Span contains timestamp, service name, calling relationship, etc.



Discard the uninformative traces and keep the informative traces.

Motivation

The number of trace data is enormous but most are similar and redundant.

Only a fraction of traces helpful to operators are those from corner cases.

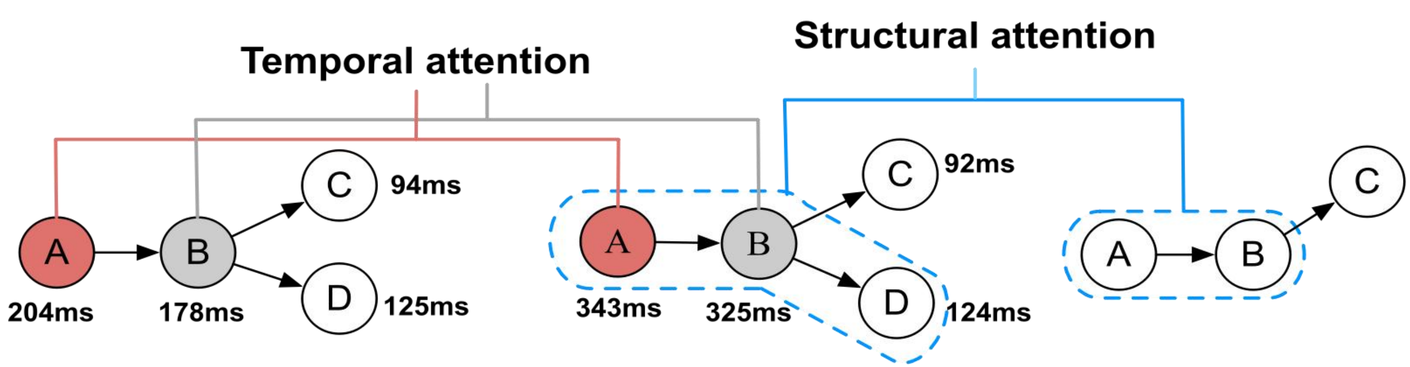
Two indicators of uncommon traces

Structural information
- calling relationship
- number of spans

Structural Attention

temporal information
- request latency

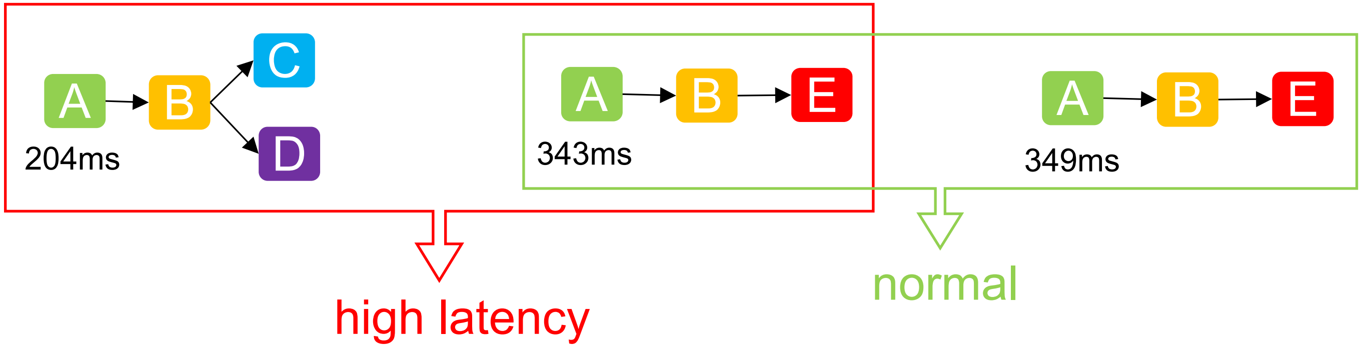
Temporal Attention



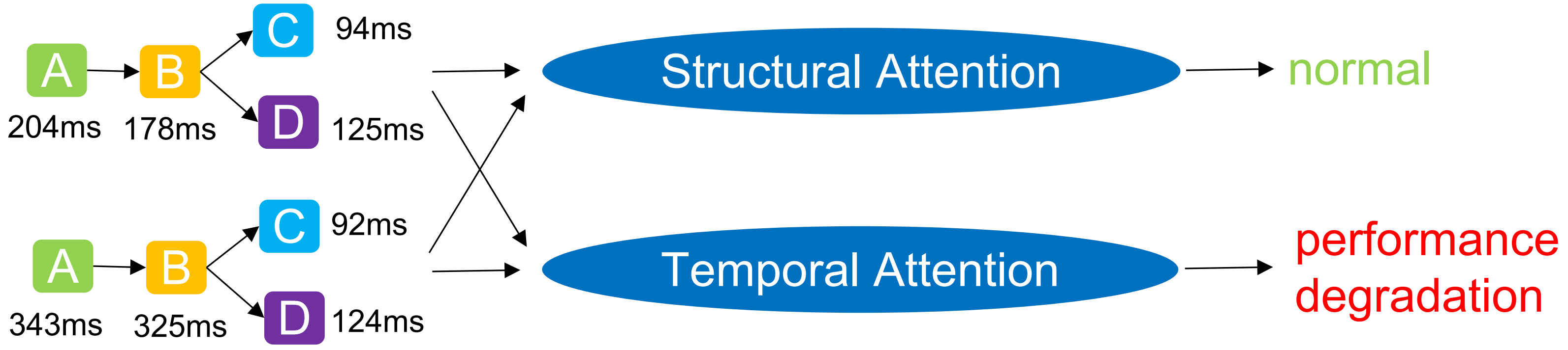
Motivation

Considering only structural attention or temporal attention can be misleading.

- Traces with high latency are not necessarily anomalous.

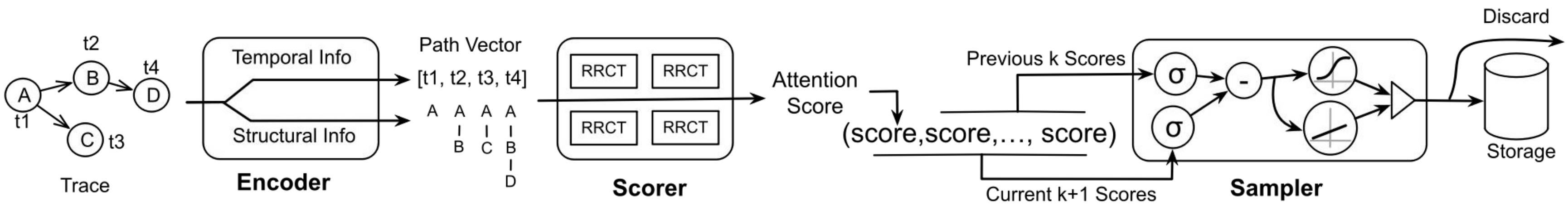


- Traces with a usual structure are not necessarily normal.



Consider the structural attention and the temporal attention at the same time.

Sampling Workflow of Sieve

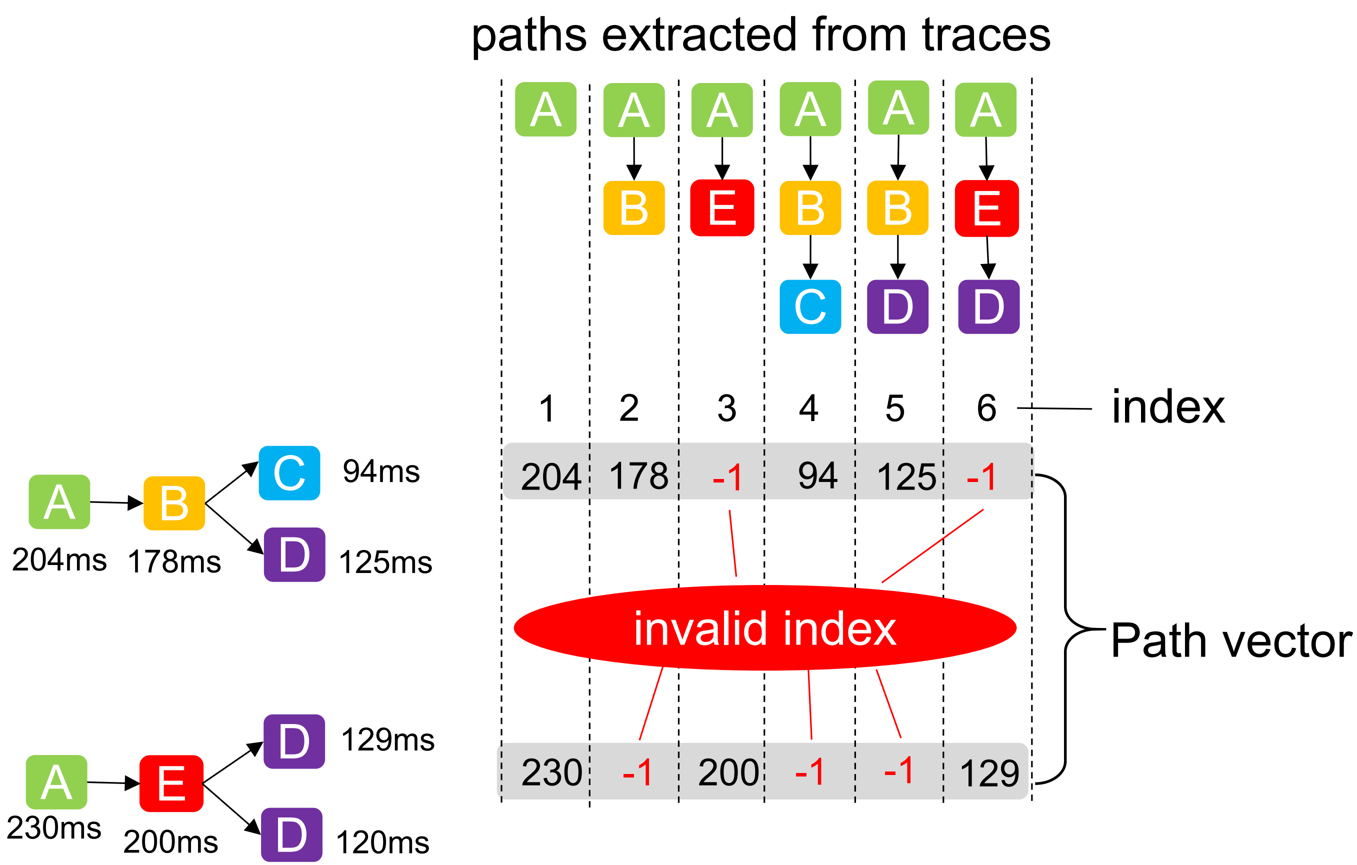


Sieve: an online sampler that biases sampling towards uncommon traces

- Encode a trace into a path vector that incorporates its structural and temporal information.
- Build a robust random cut forest model to evaluate the path vector and get an attention score.
- Calculate the sampling probability according to the historical attention scores.

Path Vector Encoder

Extract all the paths starting from the root span and their latency.
Associate each path with an index in the vector.
Assign the latency of the tail span of the path to the associated index.
Assign -1 to the associated index if the trace does not contain the path.

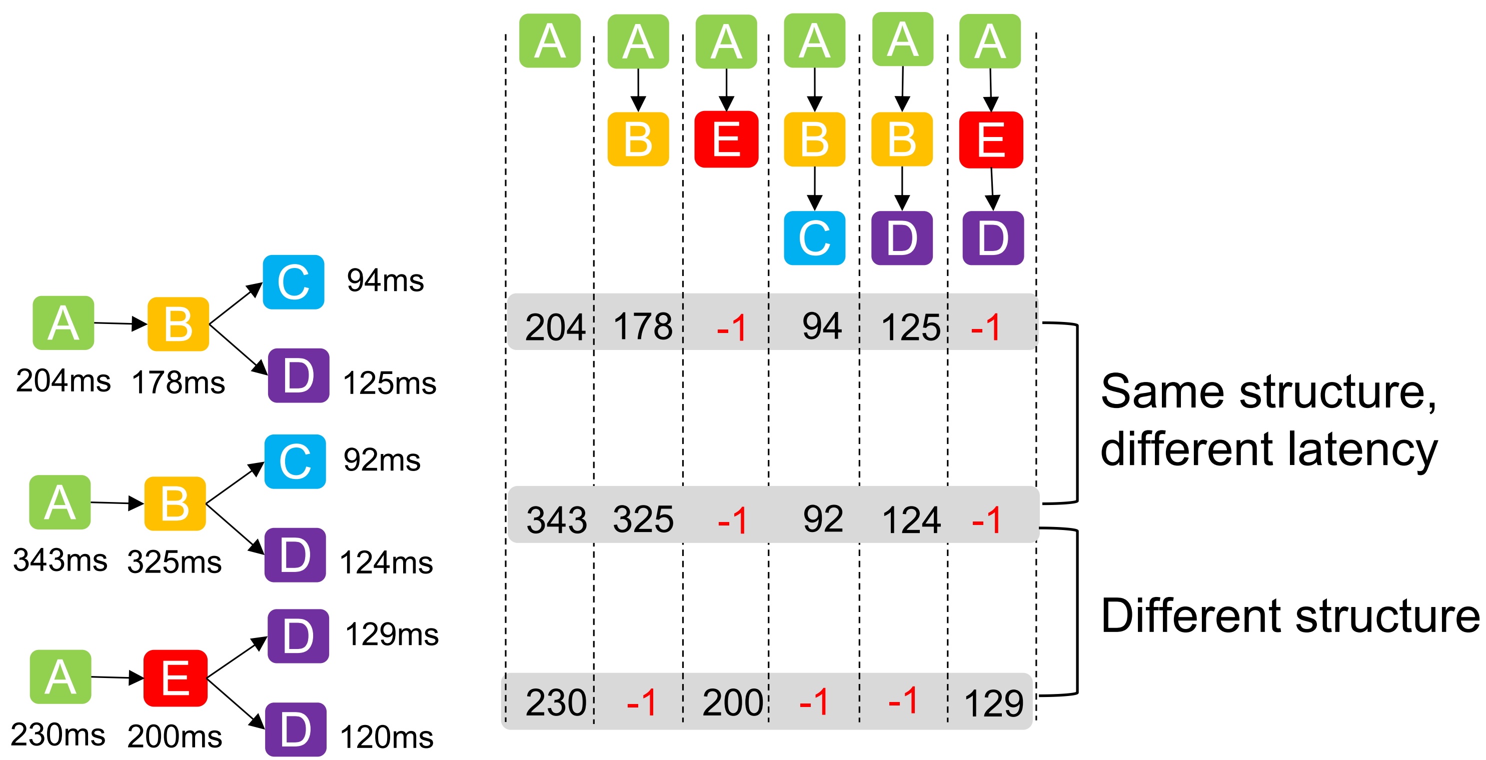


Path Vector Encoder

The structural and temporal information are incorporated into a vector.

Traces with different structures will be distinguishable.

Traces with same structures but different latency will be distinguishable.



Adaptive Scorer

Isolate the uncommon from the common traces by partition.

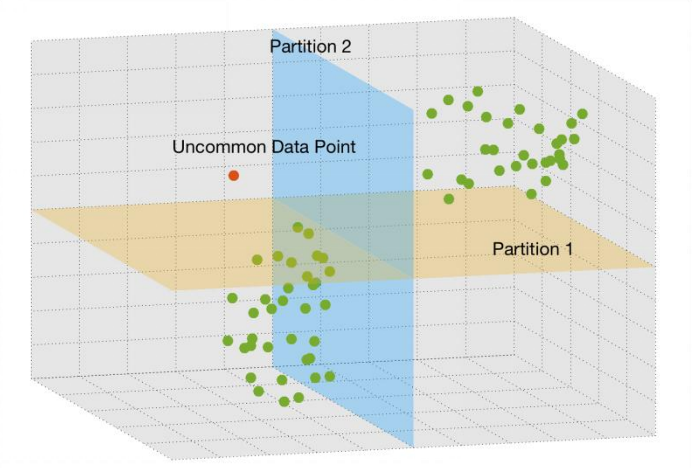
Uncommonness

↓

Deviation from the majority

↓

Easier to be isolated



Isolation Forest

- Partition the data recursively from the root and build iTrees.
- Each data point is in the leaf node.
- The degree of susceptibility is measured by the path length.

Robust Random Cut Forest (RRCF)

- Handle the outlier detection in the streaming data.
- More effective cutting dimension selection.
- The degree of susceptibility is measured in a robust way.

Adaptive Scorer

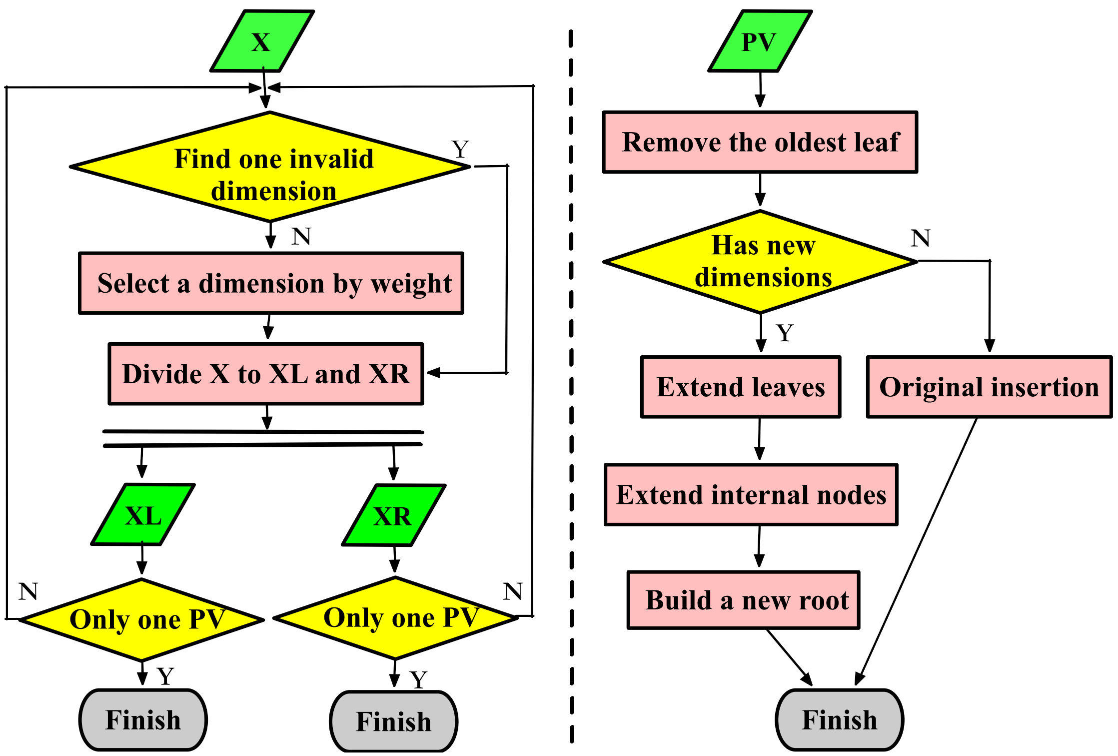
Adopt RRCF to achieve the isolation of uncommon traces.

Construction Stage (Left)

Build multiple Robust Random Cut Trees (RRCT) with different partition schemes.

Maintenance Stage (Right)

Remove the oldest leaf before new data insertion and keep the size of RRCT unchanged.

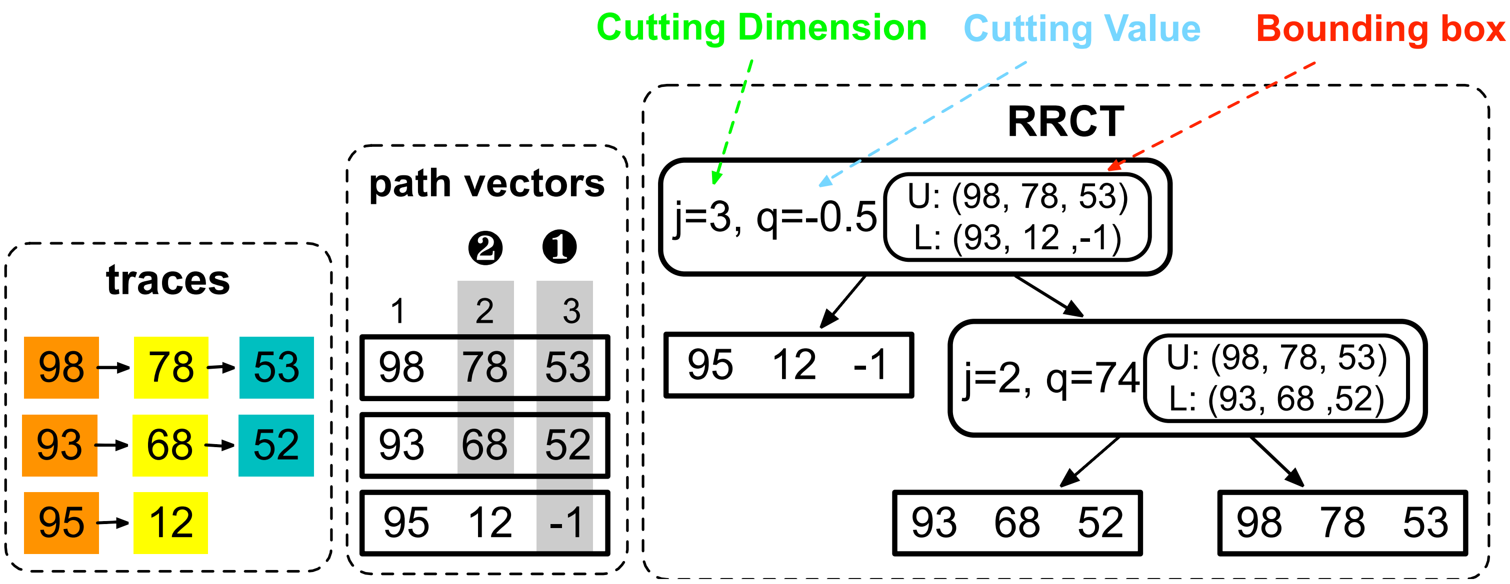


Adaptive Scorer

Improved Cutting Dimension Selection

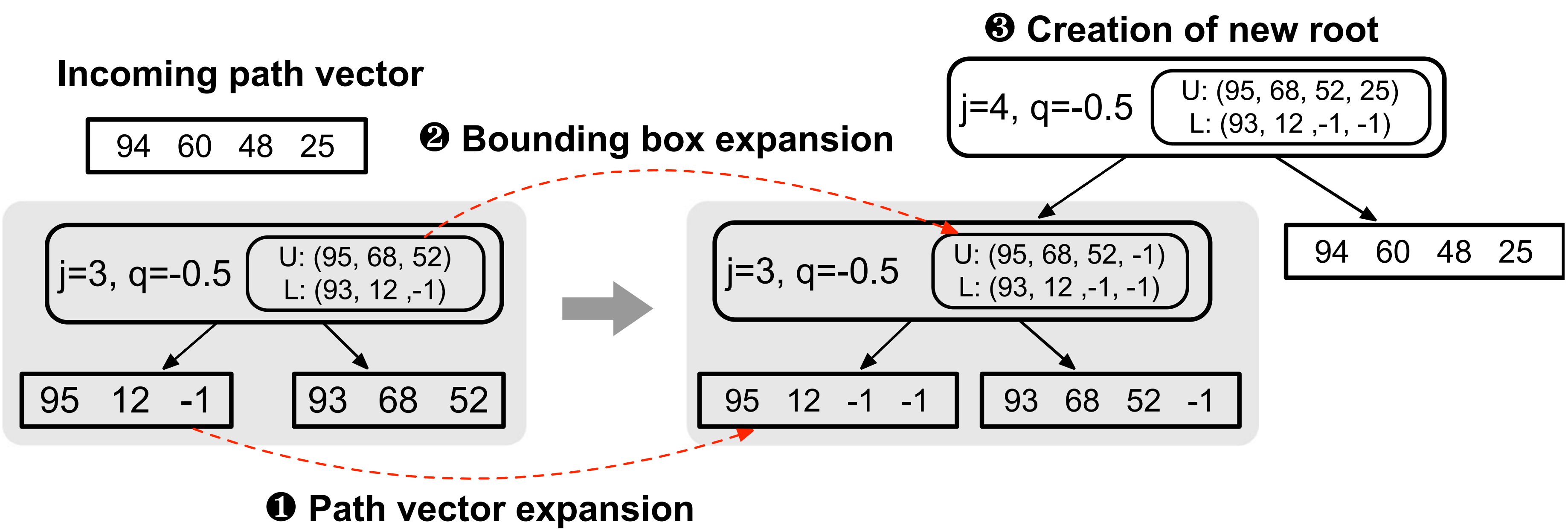
❶ Select the invalid dimensions first to give priority to structural attention.

❷ No invalid dimensions, the dimension with largest difference is most likely to be selected.



Adaptive Scorer

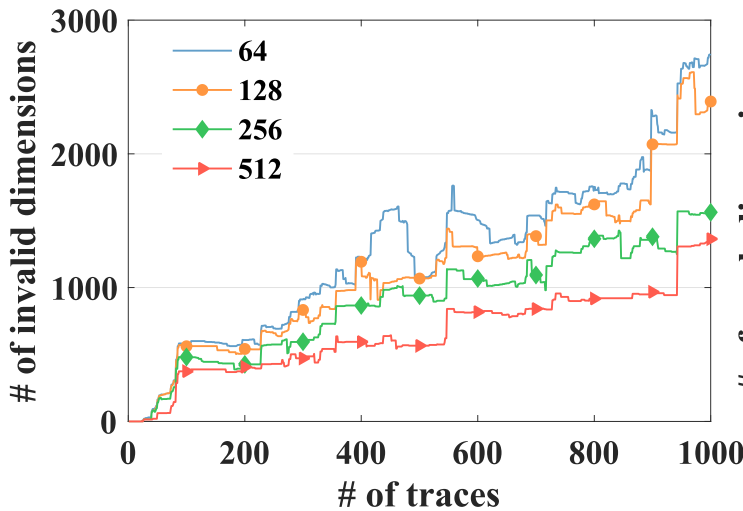
Adaptive to new traces by path vector expansion.

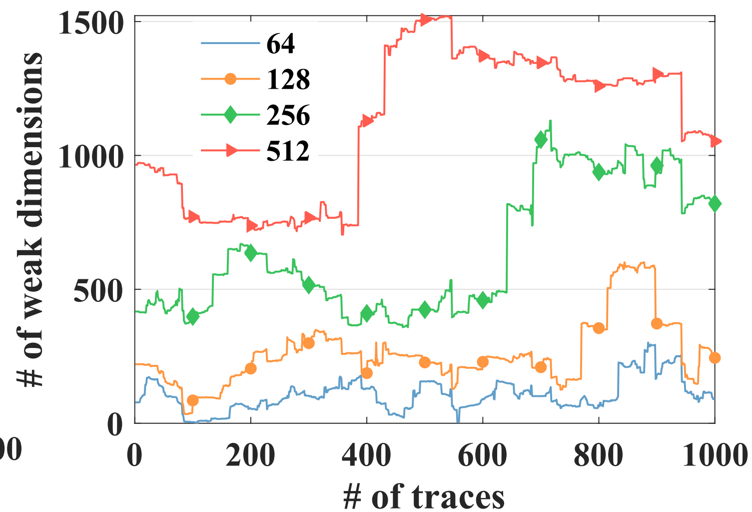


Unfortunately, the continuous expansion will result in the curse of dimensionality.

Adaptive Scorer

The ever-increasing number of invalid dimensions and weak dimensions.





Dimension Reduction

- Remove the dimensions that is invalid for all the path vectors in the tree.
- Remove the dimensions in which the variance of the values is less than 0.1.

Adaptive Scorer

Get the attention score of a trace according to its depth in the tree.

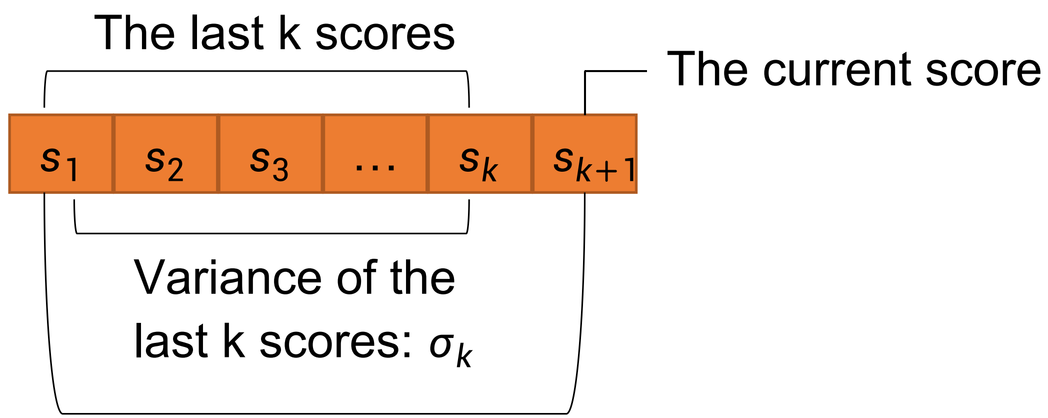
$$score = \frac{depth\ of\ the\ tree}{depth\ of\ the\ leaf\ contains\ the\ trace}$$

Integrate the scores from each RRCT into the final attention score.

$$final\ score = \frac{Sum\ of\ scores\ obtained\ from\ each\ tree}{number\ of\ tree}$$

Biased Sampler

Calculate the sampling probability according to the historical scores.



variance and mean of the k+1 scores: σ_{k+1}, μ_{k+1}

$$\sigma_{k+1} - \sigma_k > h$$

Ye
s

No

converge to 1 rapidly, biased sampling towards the uncommon

$$\frac{1}{1 + e^{2\mu_{k+1} - s_{k+1}}}$$

$$\frac{s_{k+1}}{\sum_{i=1}^{k+1} s_i}$$

linear growth of sampling probability

Evaluation

Dataset

VWR	a simulated microservice system	34167 traces
AIOps	a Real-world microservice system	168432 traces
Boutique	a microservice benchmark	2000 traces
TC	real production traces	6561 traces

Experiments

- effectiveness for temporally and structurally uncommon traces
- sensitivity to the degree of uncommonness and parameter settings
- performance comparison
- representative sampling
- overhead

Effectiveness

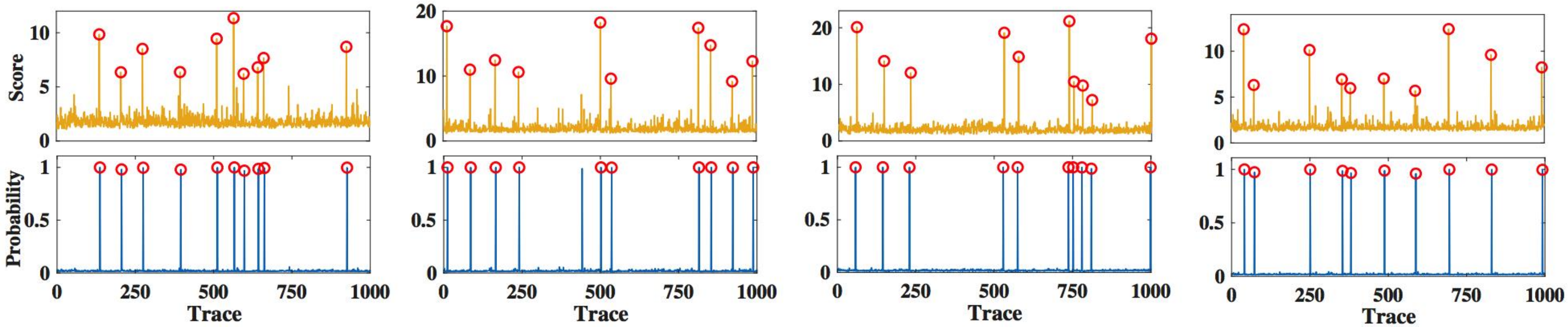
effectiveness of temporal attention

990 common traces
10 uncommon traces

Experiment Setting

Dataset	Spans	Latency	
		Common	*Uncommon*
VWR	6	$200 \sim 400$ms	> 100000ms
AIOps	58	$100 \sim 200$ms	$500 \sim 1000$ms
Boutique	28	< 200ms	> 500ms
TC	15	$22 \sim 30$ms	$50 \sim 66$ms

Experiment Results



(a) VWR (b) AIOps (c) Boutique (d) TC

- sampling probability of the common: < 0.02
- sampling probability of the uncommon: > 0.99

Effectiveness

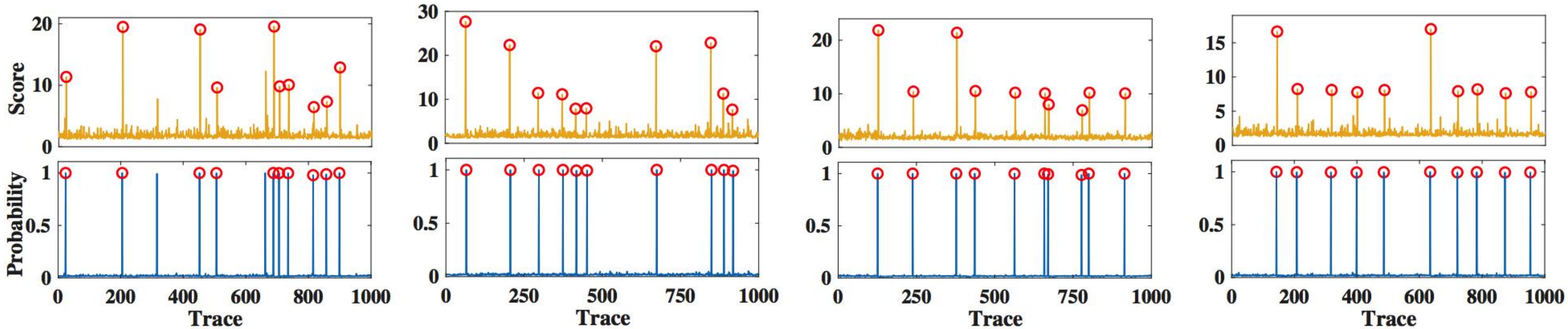
effectiveness of structural attention

990 common traces
10 uncommon traces

Experiment Setting

Dataset	Latency	Spans	
		Common	*Uncommon*
VWR	< 200ms	6	{4,5}
AIOps	$400 \sim 450$ms	58	59
Boutique	< 200ms	28	18
TC	$25 \sim 35$ms	15	14

Experiment Results



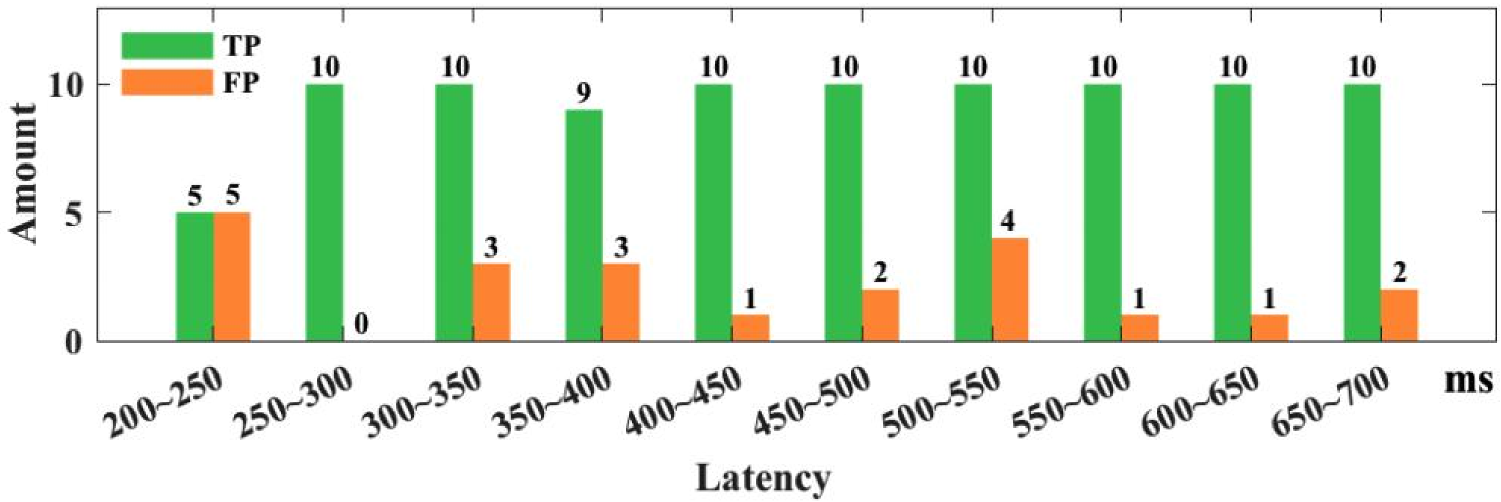
(a) VWR (b) AIOps (c) Boutique (d) TC

- Detect all the uncommon traces, even those with subtle difference in structure.

Sensitivity

sensitivity to the degree of temporal uncommonness

990 common traces (latency < 200ms)
10 uncommon traces



As the difference becomes more distinguishable, the detection ability is improved rapidly and keeps the false positive rate at a low level.

Sensitivity

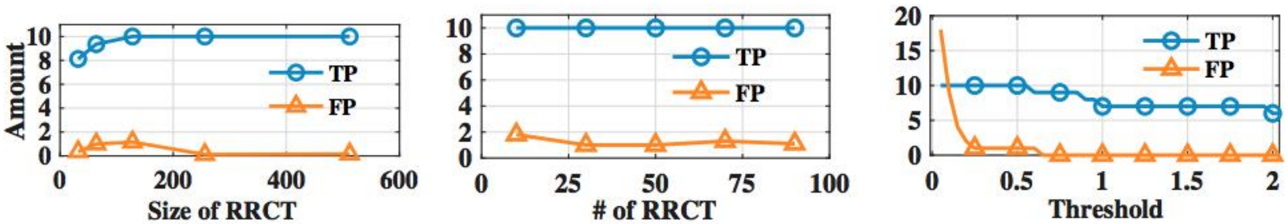
sensitivity to the parameter settings

Experiment setting

990 common traces
10 uncommon traces

Type	Spans	Latency	Amount
common	58	$0 \sim 300ms$	990
uncommon	58	$> 400ms$	5
	59	$0 \sim 300ms$	2
	7	$0 \sim 300ms$	3

Experiment Results



(a) Size of RRCT (b) Number of RRCT (c) Threshold

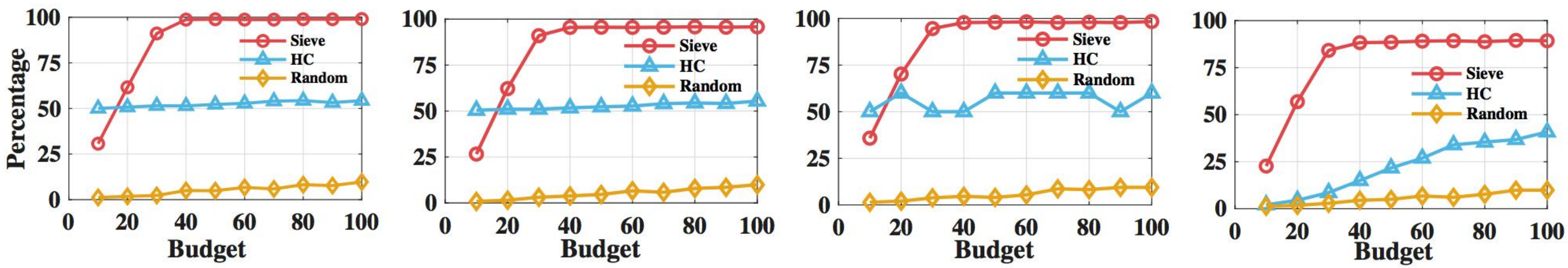
Performance

Experiment setting

Type	VWR		AIOps		Boutique		TC		Amount
	Spans	Latency	Spans	Latency	Spans	Latency	Spans	Latency	
common	6	$0 \sim 200ms$	58	$0 \sim 300ms$	28	$0 \sim 200ms$	29	$0 \sim 40ms$	990
uncommon	6	$> 100000ms$	58	$> 400ms$	28	$> 500ms$	29	$60 \sim 90ms$	5
	5	$0 \sim 200ms$	59	$0 \sim 300ms$	18	$0 \sim 200ms$	30	$0 \sim 40ms$	5
	7	$0 \sim 300ms$	4	$0 \sim 200ms$					

contrast methods: Hierarchical Clustering, Random Sampling

Experiment results



(a) VWR (b) AIOps (c) Boutique (d) TC

Sieve retains the uncommon traces with a low sample size.

Representative Sampling

Experiment results

Increase the proportion of the minority in the samples.

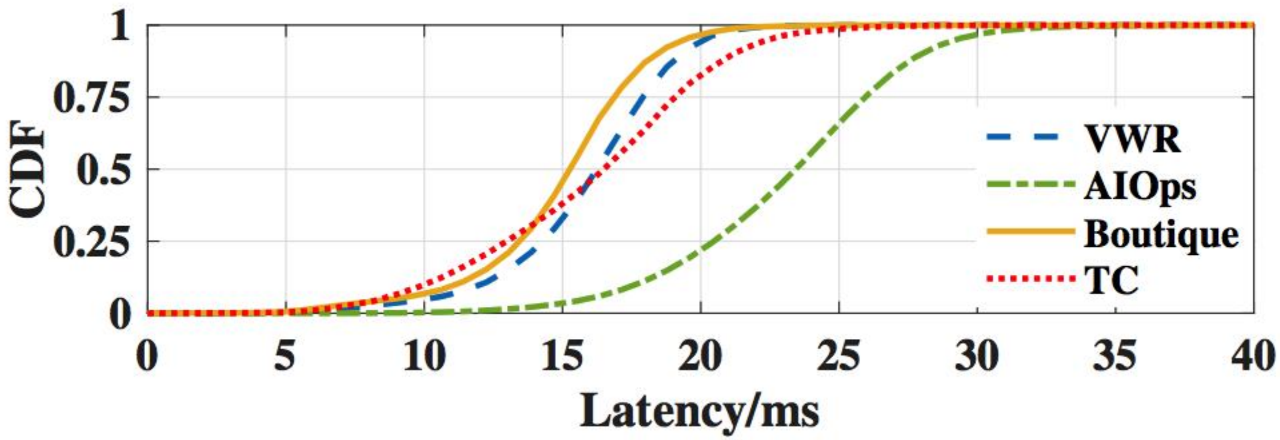
	API-1	API-2	API-3	API-4	API-5	API-6	API-7	API-8	API-9	API-10
Population	11	16	48	119	135	210	325	571	607	4519
Sieve	11	16	41.2	64	51	31.2	9.4	15.2	10.8	65.4
HC	5.9	16	5.44	17.22	19.44	32.64	42.24	45.92	42.84	87.56
Random	0.42	0.68	2.3	5.38	6.34	10.78	16.12	27.28	29.7	217

The reduction of storage is up to 97.5%.

	Population	Sample	Sampling Rate
VWR	34167	85	2.5%
AIOps	168432	7602	4.5%
Boutique	2000	114	5.9%

Overhead

Experiment results



Sampling latency varies between 3ms and 33ms.

Conclusion

- Sieve leverages the attention mechanism to bias sampling towards the temporally and structurally uncommon traces.
- Sieve incorporates temporal and structural information of a trace into a path vector.
- Sieve builds an enhanced RRCF model to evaluate the attention score of a trace.
- Sieve biases sampling by comparing the attention score with the historical scores.
- Sieve retains the informative traces and saves storage up to 97.5%.

Thank you for watching!