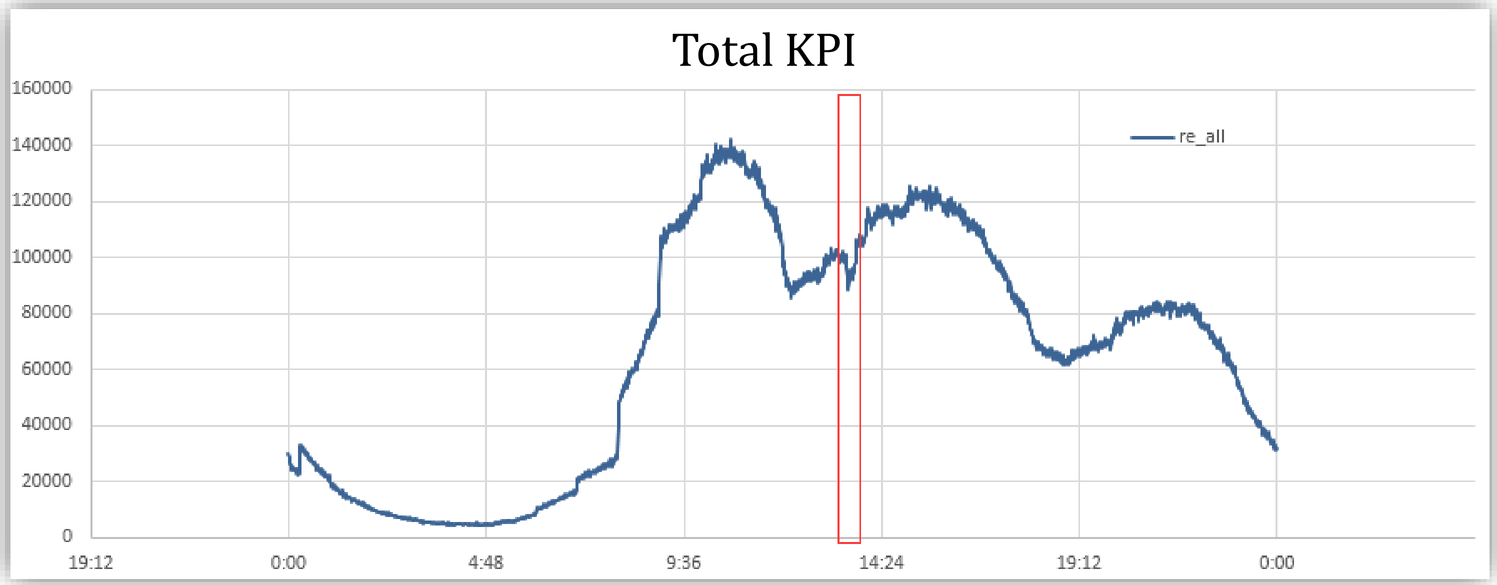
HotSpot: Anomaly Localization for Additive KPIs with Multi-Dimensional Attributes

Yongqian Sun

2018/11/21

Motivation

- **When the total KPI is anomaly, we need to localize the root cause of fine-grained indicators.**



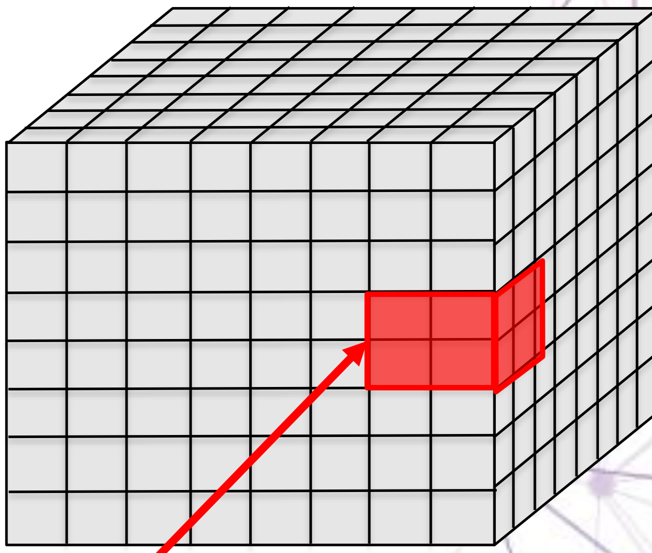
Anomaly sample

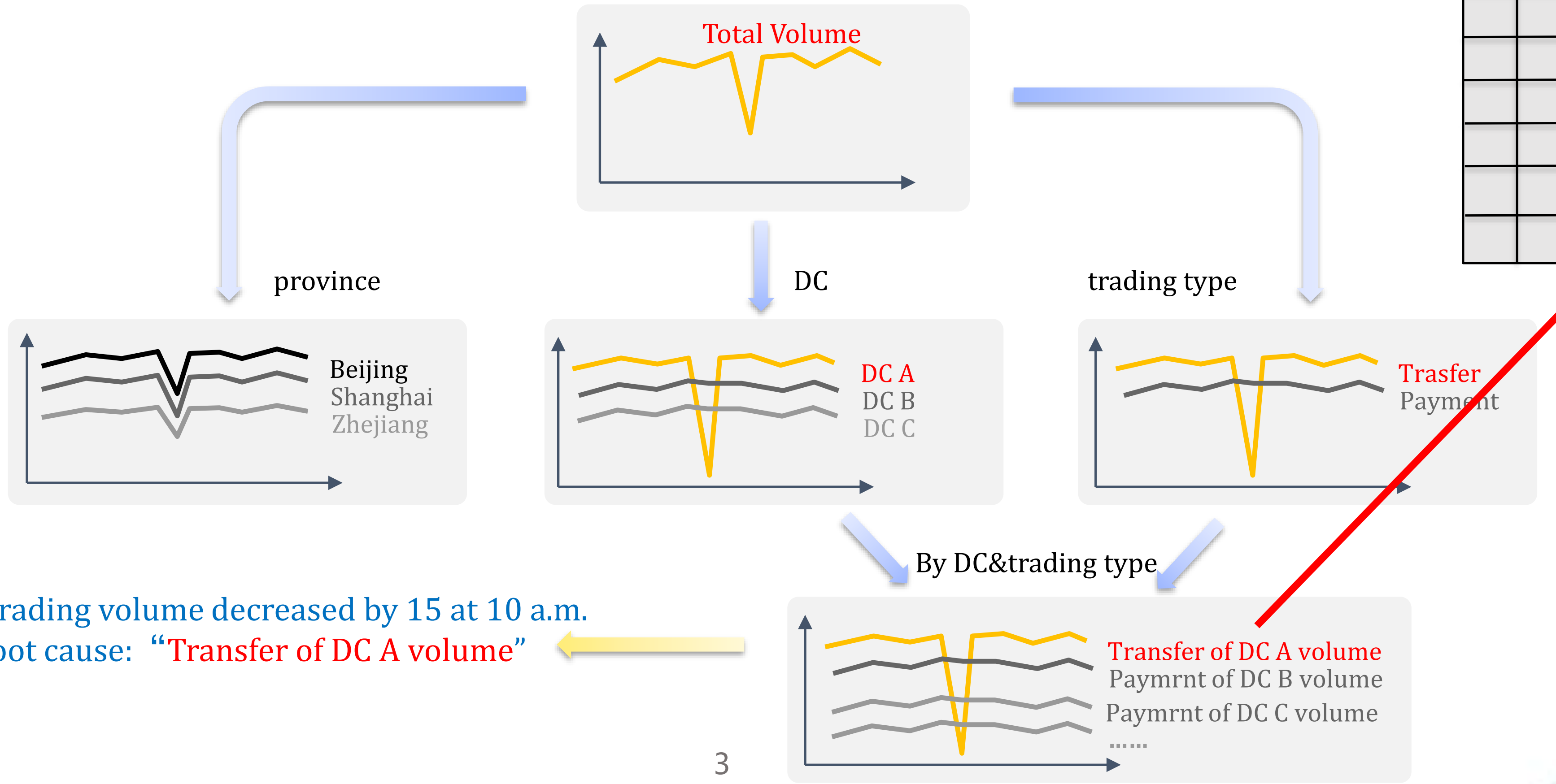
KPI: Key performance indicator

A case

time	trading volume	province	DC	trading type	ISP	Device	……
2018/05/07 10:00	235600	Beijing	A	transfer	ChinaNet	iphone	……

Multi-dimensional anomaly localization: When the total amount of a multi-dimensional attribute KPI is anomaly, it is necessary to localize the specific element (or a set of elements) where the root cause lies.

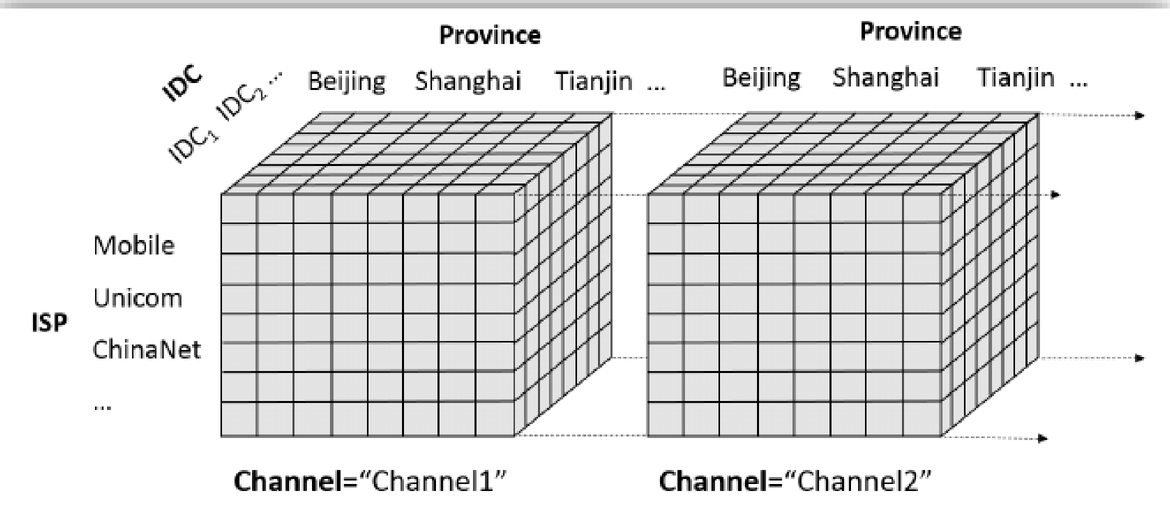




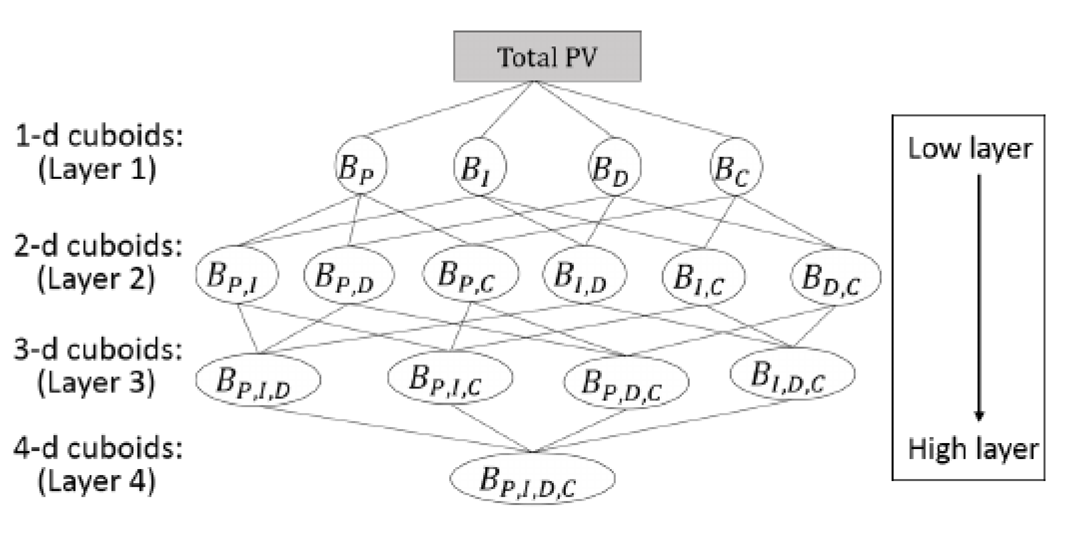
Total trading volume decreased by 15 at 10 a.m.
Root cause: “Transfer of DC A volume”

Problem

- **Date cube**



4-d data cube, represented as a series of 3-d data cubes



Cuboids in a 4-d data cube

Problem: Effectively and efficiently localize the most potential root cause, i.e., a subset of elements of one specific cuboid Bi, for a total KPI value anomaly. The root cause set RSet ⊆ E(Bi).

Related work 1

- **Adtributor [NSDI14]:**

Step1. For each dimension, find a set of changed elements based on **Explanation Power**.

Step2. Find the sets that are the most succinct.

Step3. Compare **Surprise** of the sets.

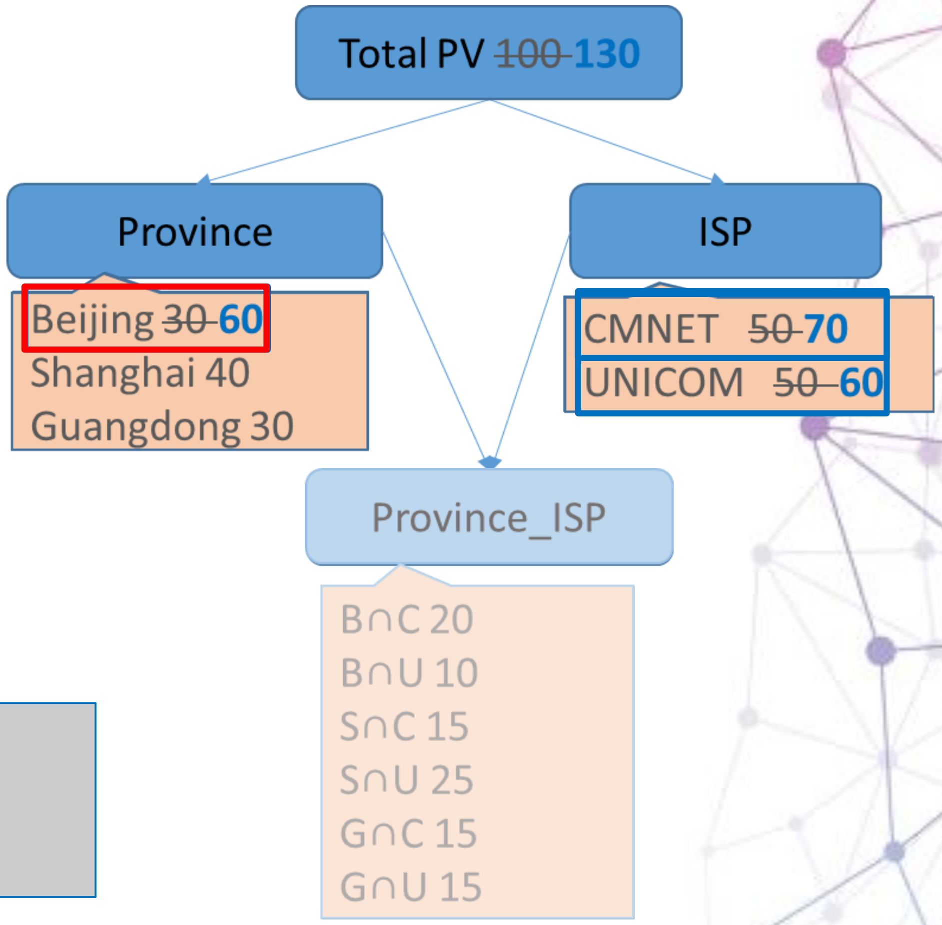
Surprise of “beijing”:

p=bejing_forcast/totalPV_forecast;

q=bejing_real/totalPV_real;

S=0.5*(p*log(2*p/(p+q))+ q*log(2*q/(p+q)))

S_bejing=0.025; S_CMNET=0.001



Disadvantages:

1. Can't handle cross dimensions.
2. Solution appears to be ad hoc and weak.

Related work 2

- **iDice [ICSE16]:**

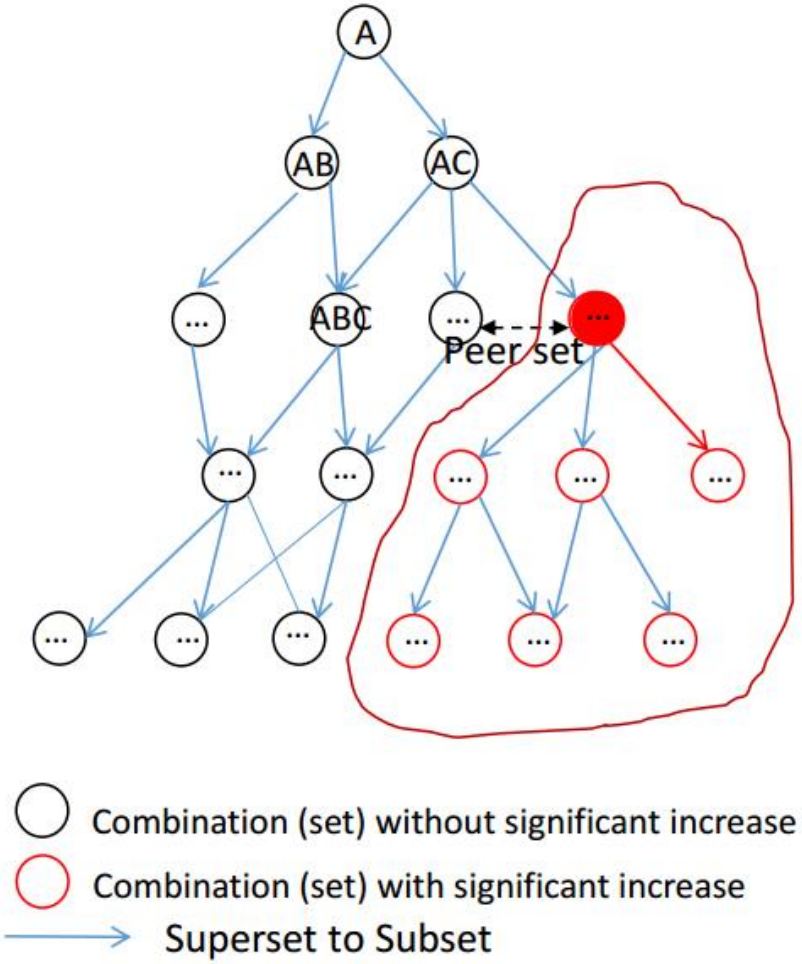
Step1. Impact based pruning.

Step2. Change detection based pruning.

Step3. **Isolation Power** based pruning.

$$IP(X) = -\frac{1}{\overline{\Omega_a} + \overline{\Omega_b}} \left(\overline{X_a} \ln \frac{1}{P(a|X)} + \overline{X_b} \ln \frac{1}{P(b|X)} + (\overline{\Omega_a} - \overline{X_a}) \ln \frac{1}{P(a|\overline{X})} + (\overline{\Omega_b} - \overline{X_b}) \ln \frac{1}{P(b|\overline{X})} \right)$$

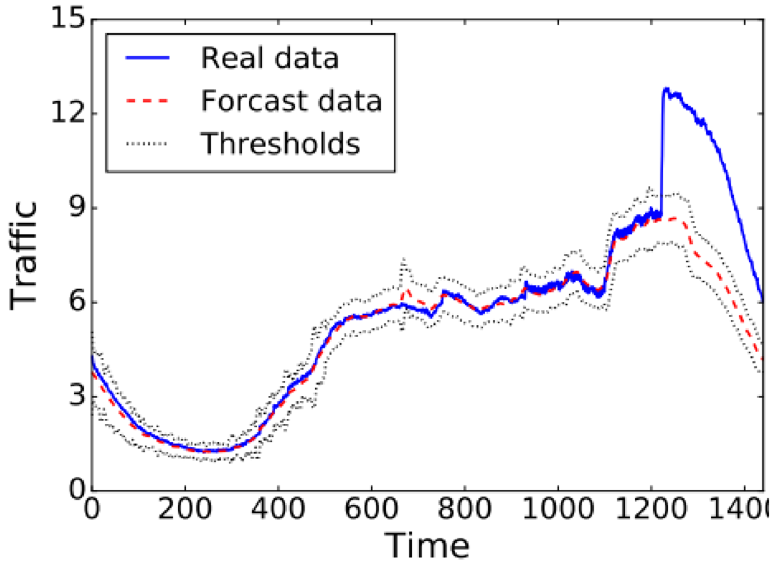
Step4. Ranking results with a fisher distance.



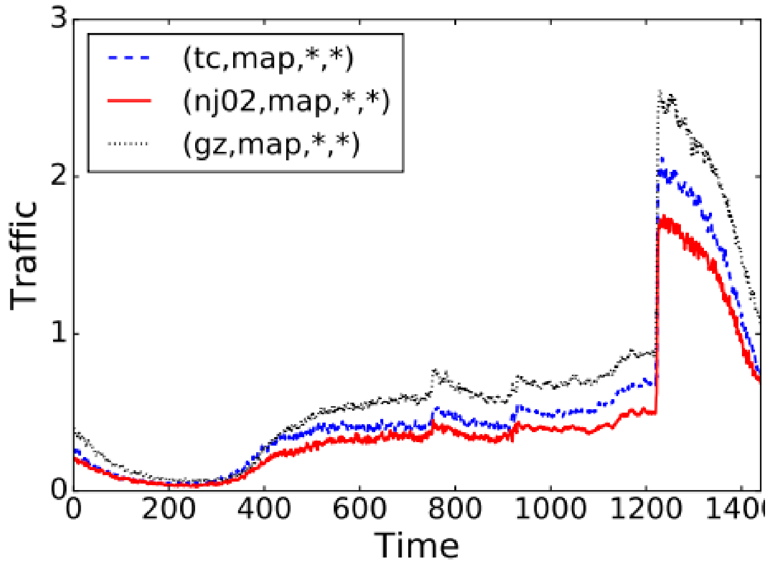
Disadvantages:
1. Brute-force pruning may lead to loss of precision.
2. The result will be very poor when there are **more than two elements** in the root cause set.

Related work 2

- **Why to concern about more than two elements in the root cause set?**
  - **A case of baidu:**



(a) Total



(b) Root cause elements

Huawei、Tencent、Ant Financial have implemented this algorithm。

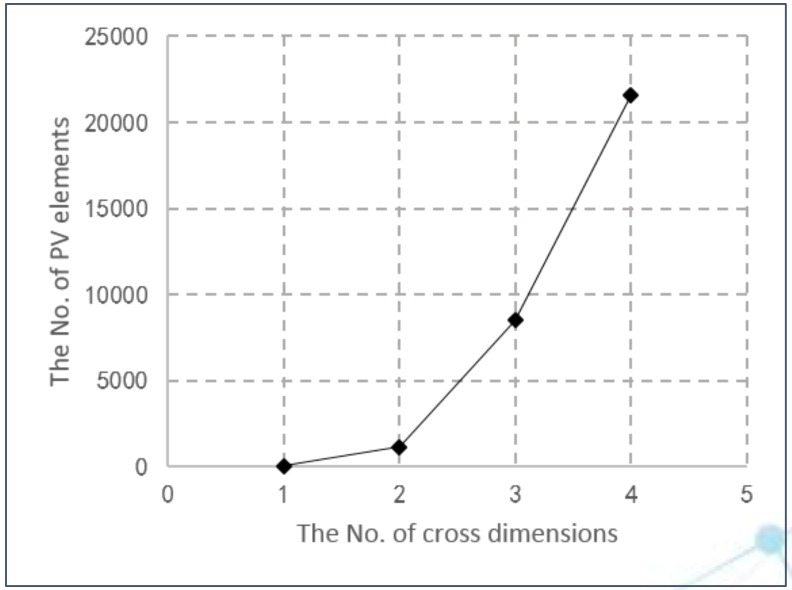
Challenges

1. There exist complex relationships among elements, then it is different to measure the potential of an element set.

$f(p,i) \rightarrow v(p,i)$		Province(p)			
		Beijing	Shanghai	Guangdong	*
ISP (i)	Mobile	20→14	15→9	10→10	45→33
	Unicom	10→7	25→15	20→20	55→42
	*	30→21	40→24	30→30	100→75

2. Too many elements in multi-dimensional system.

Layer	Dimensions(The elements number)					
Layer 1	IDC (6)		P(36)	ISP(10)	C(10)	
Layer 2	IDC_P (216)	IDC_ISP (60)	IDC_C (60)	P_ISP (360)	P_C (360)	ISP_C (100)
Layer 3	IDC_P_ISP (2160)		IDC_P_C (2160)	IDC_ISP_C (600)	P_ISP_C (3600)	
Layer 4	IDC_P_ISP_C(21600)					
Total	31338					



Potential score

- **A 2-d case**

$v(p,i)$		Province(P)			
		Beijing	Shanghai	Guangdong	*
ISP (I)	Mobile	20	15	10	45
	Unicom	10	25	20	55
	*	30	40	30	100(Total)

$f(p,i)$ $v(p,i)$		Province			
		Beijing	Shanghai	Guangdong	*
ISP	Mobile	20 → 8	15 15	10 10	45 → 33
	Unicom	10 → 4	25 25	20 20	55 → 49
	*	30 → 12	40 40	30 30	100 → 82

- **The Ripple effect**
  - The anomaly e changes d, e_i related to e in the most fine-grained cross dimension:

$$r(e_i) = f(e_i) + d\frac{f(e_i)}{\sum_j f(e_j)}$$

Core idea

- **Potential Score (*ps*)**
  – Measure the potential of an element set.

$$Potential\ Score = max(1 - \frac{d(\vec{v}, \vec{a})}{d(\vec{v}, \vec{f})}, 0)$$

- **A case:**

$f(p,i) \to v(p,i)$		Province(p)			
		Beijing	Shanghai	Guangdong	*
ISP (i)	Mobile	20→14	15→9	10→10	45→33
	Unicom	10→7	25→15	20→20	55→42
	*	30→21	40→24	30→30	100→75

- **Calculate and choose the largest *ps***

$\vec{f}$ = (20, 15, 10, 10, 25, 20)

$\vec{v}$= (14, 9, 10, 7, 15, 20)

– Examples

$\vec{a}\{(Beijing,*)\}$=(14, 15, 10, 7, 25, 20)

$ps\{(Beijing,*)\}=0.13$ $ps\{(*; Mobile);(*;Unicom)\}=0.13$ $ps\{(Beijing,*), (\{Shanghai,*\})\}=1$

– Traverse all possible sets to find the root cause

$ps\{(Beijing,*), (\{Shanghai,*\})\}=1$

Process

Traverse all sets → Calc and compare potential score

4 dimensions:

- L1, 4 dimensions:
  - A1(n1), A2(n2), A3(n3), A4(n4)
- L2, C_4^2 attributes:
  - A1A2(n1*n2), A1A3(n1*n3), A1A4(n1*n4), A2A3(n2*n3), A2A4(n2*n3), A3A4(n3*n4)
- L3, C_4^3 attributes:
  - A1A2A3(n1*n2*n3), A1A2A4(n1*n2*n4), A1A3A4(n1*n3*n4), A2A3A4(n2*n3*n4)
- L4, 1 attribute:
  - A1A2A3A4(n1*n2*n3*n4)

No. of all sets:

- $2^{n1} - 1 + \cdots + 2^{n1*n2*n3*n4} - 1 \approx 2^{n1*n2*n3*n4}$
- For example: n1,n2,n3,n4 = 6, 36, 10, 10

No.: 2^21600

The average time cost of a set:

- Average elements No. of attribute:
  - 2089.2
- Average elements No. of a set:
  - 1044.6
- Average cost of an element:
  - 0.3s
- An attribute with 36 elements cost:
  - (2^36-1)*18*0.3 ≈11767years

Cost of compute a distance :

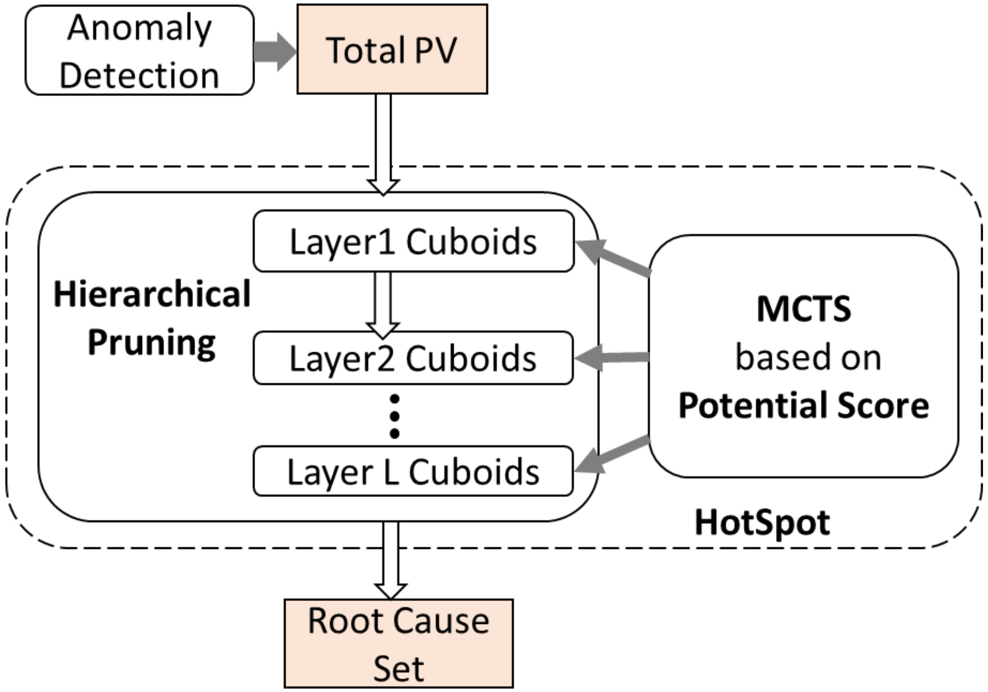
- Euclidean distance： 0.012s

Cost of compute a ps : 0.01s

Bottleneck:
Traverse all sets to calc potential scores.

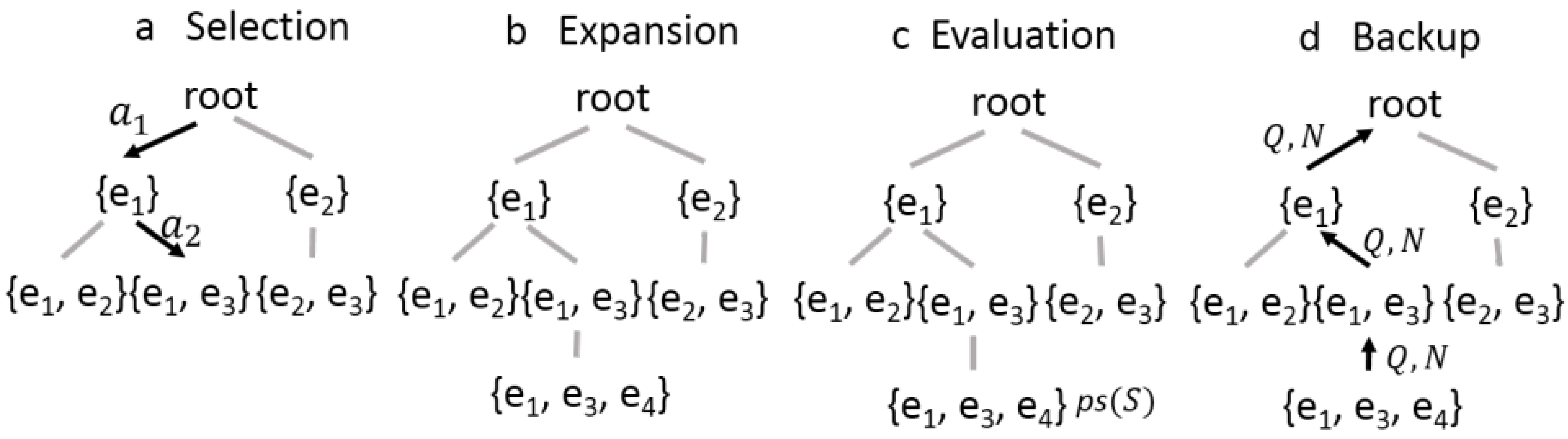
Overview

- **Apply MCTS and hierarchical pruning**

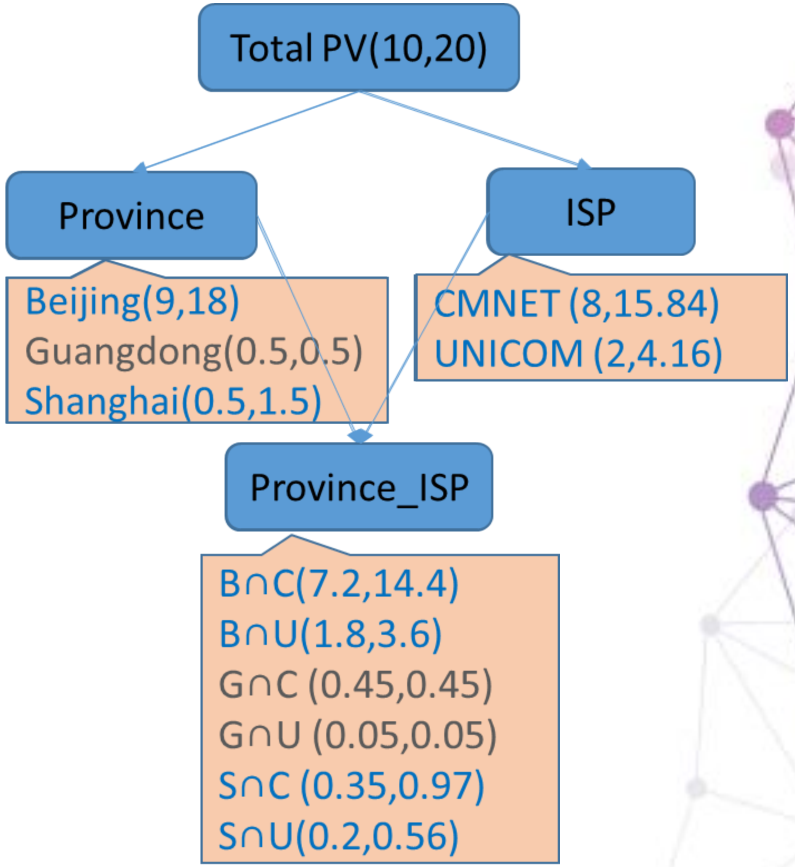


MCTS

- **Monte Carlo tree search (MCTS) is a heuristic search algorithm for some kinds of decision processes, most notably those employed in game play, eg., AlphaGo.**
  - **The decision of the next step position <==> The decision of adding an element to a set**
  - **The steps of one iteration of MCTS :**



- **Hierarchical :**
  - Prune:
    - In layer 1: choose the most potential set in each dimension.
    - Then in layer 2, the options of elements is narrowed down.
  - Anomaly detection:
    - Only detect the anomalies that need to be diagnosed



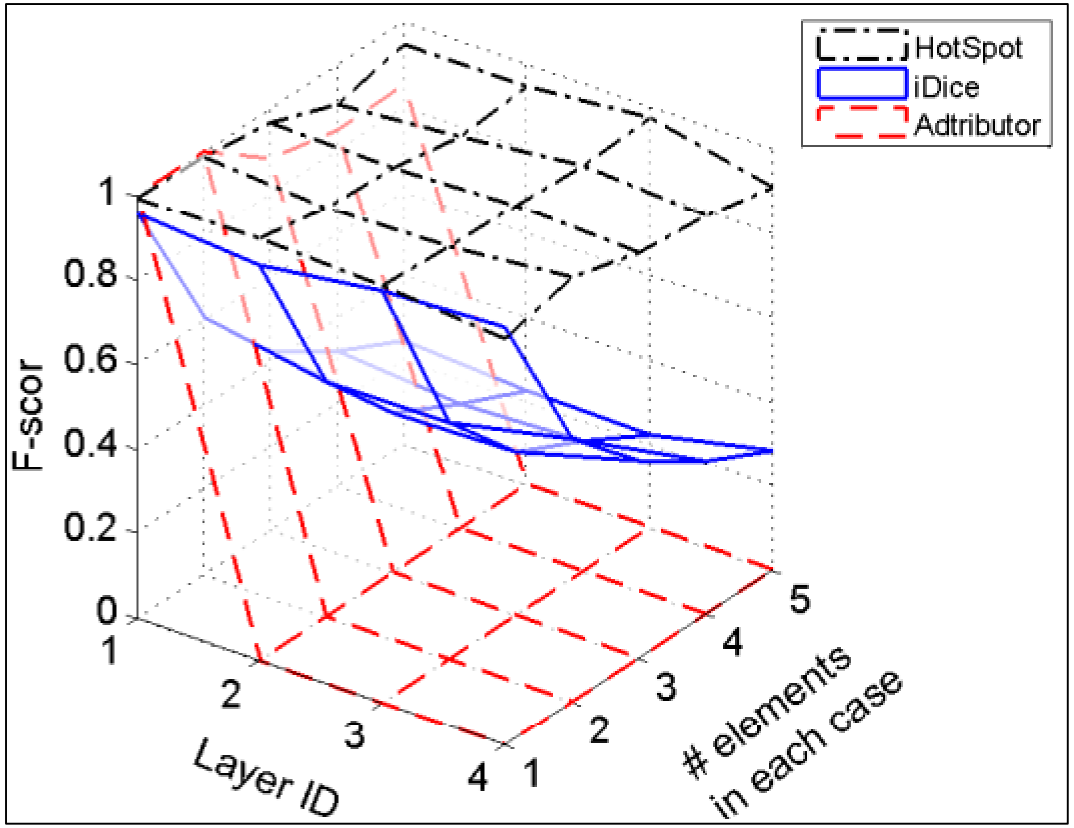
Evaluation 1

- **Metrics:**
  - **Running time**
  - **F-score**
    - $Precision = \frac{True\ Positive}{True\ Positive+False\ Positive}$
    - $Recall = \frac{True\ Positive}{True\ Positive+False\ Negative}$
    - $F-score = \frac{2*precision*recall}{precision+recall}$

  - **True positive: the number of root cause elements correctly localized.**
  - **False positive: the number of the root cause elements wrongly localized.**
  - **True negative: the number of anomaly elements correctly localized.**
  - **False negative: the number of anomaly elements don't be localized.**

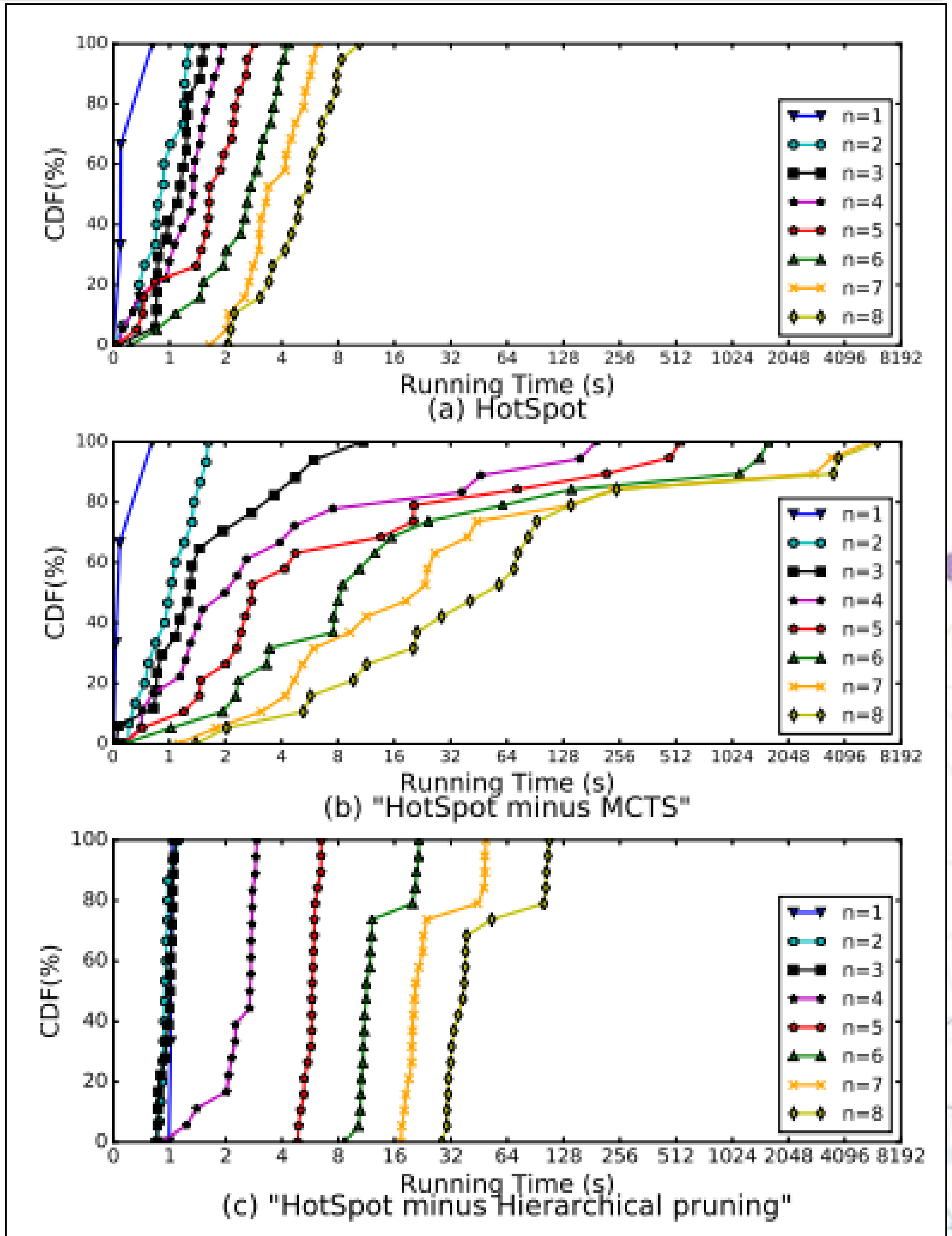
Evaluation 2

- **The F-score comparison of the three algorithms**



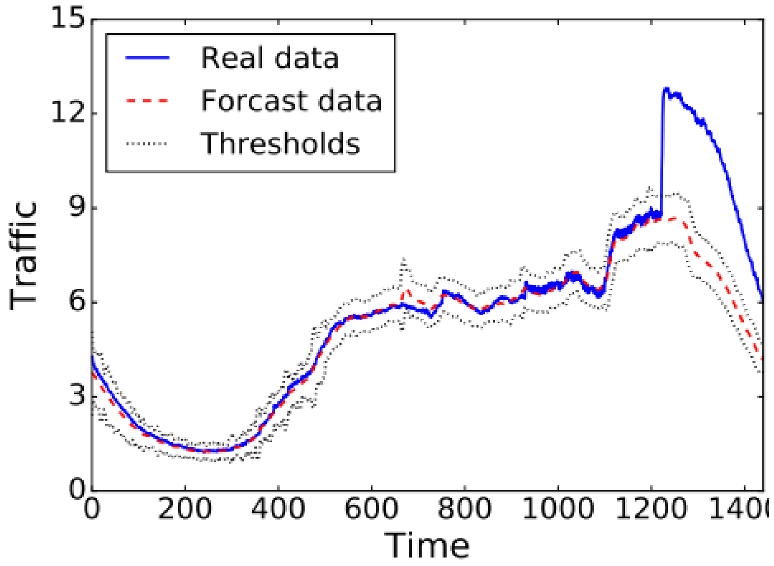
Evaluation 3

- **Comparison of running time of HotSpot, “HotSpot minus MCTS” and “HotSpot minus hierarchical pruning”**

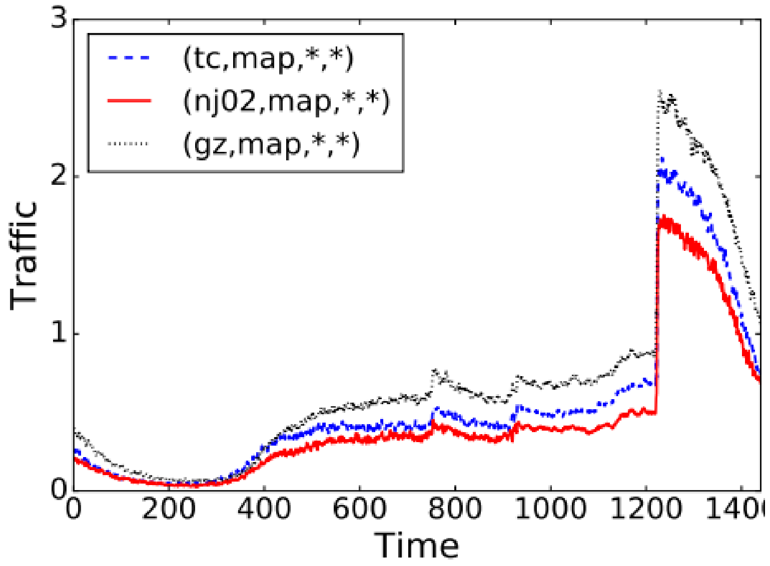


Real-world Case 1

Dimensions: IDC (with 11 elements), Product (182 elements), ISP (7 elements) and Cluster (480 elements).



(a) Total

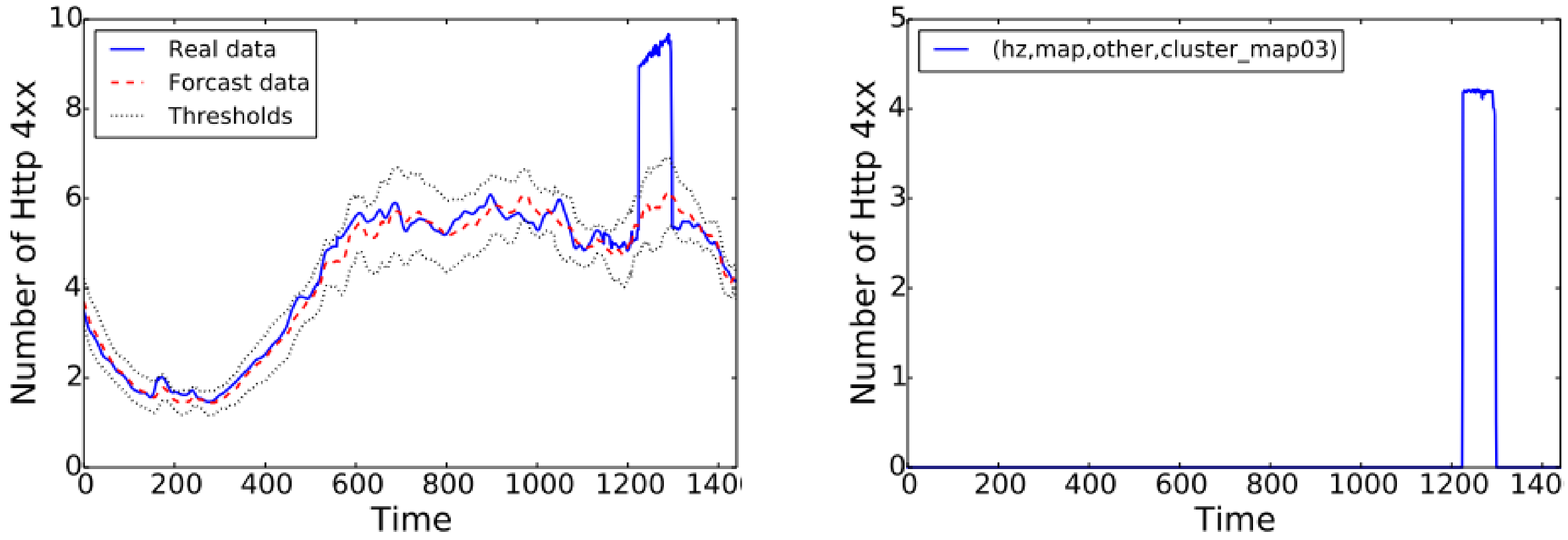


(b) Root cause elements

HotSpot reduces the localization time from about more than 1.5 hours in manual efforts to less than 20 seconds.

Real-world Case 2

Dimensions: IDC (with 11 elements), Product (182 elements), ISP (7 elements) and Cluster (480 elements).



(a) Total (b) Root cause elements

HotSpot reduces the localization time from about more than 1 hour in manual efforts to less than 20 seconds.

Conclusion

- For a multi-dimensional attributes KPI, it is a hard problem to localize the overall KPI’s anomaly to the root caus**e.**
  - Firstly, we considered this anomaly localization as a **search problem** with a huge space.
  - To solve the problem of “complex relationship”, we proposed **potential score** based on the “**ripple effect**”.
  - To deal with the huge search space, HotSpot adopted the **MCTS** and **hierarchical pruning**.

Thanks!