iBTune: Individualized Buffer Tuning for Large-scale Cloud Databases

Jian Tan, Tieying Zhang, Feifei Li, Jie Chen, Qixing Zheng, Ping Zhang, Honglin Qiao, Yue Shi, Wei Cao, Rui Zhang
Alibaba

VLDB 2019

Outline

- Background and Motivation
- Algorithm and System
- Deployment and Result

Background

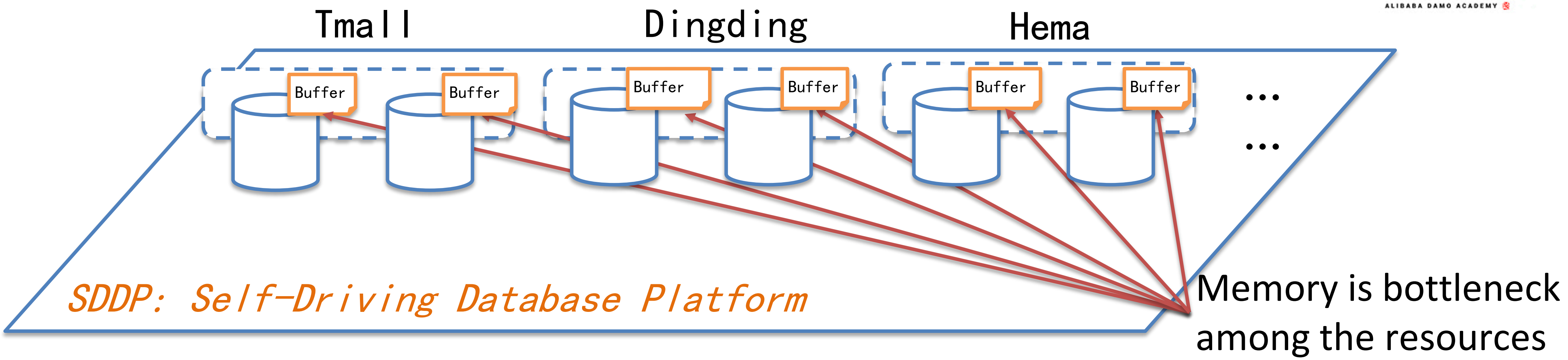


Table 1: Usage of different memory pools

Memory Pool	buffer pool	insert buffer	log buffer	join buffer	key buffer	read buffer	sort buffer
Avg. Size	29609.98M	8.00M	200.00M	0.13M	8.00M	0.13M	1.25M
Percent	99.27%	0.03%	0.67%	0.00%	0.03%	0.00%	0.00%

The memory uses at Alibaba product environment

Buffer pool is the largest memory consumer

Motivation

達摩院
ALIBABA DAMO ACADEMY

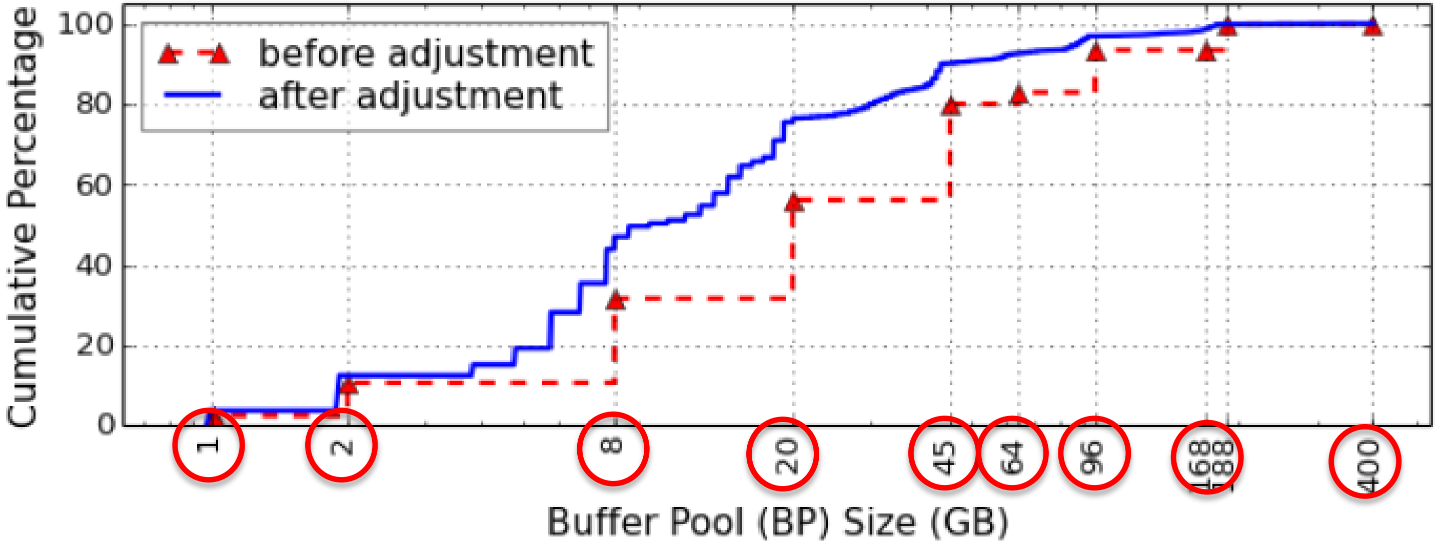
Reduce memory (buffer pool) while guaranteeing SLA (response time).

- DBA manually uses a small number of BP sizes (10 configurations in our case).

- Each instance's BP size might be different as the query workload is different.

- Manual tuning is not scalable for large cloud databases since each instance has different BP size.

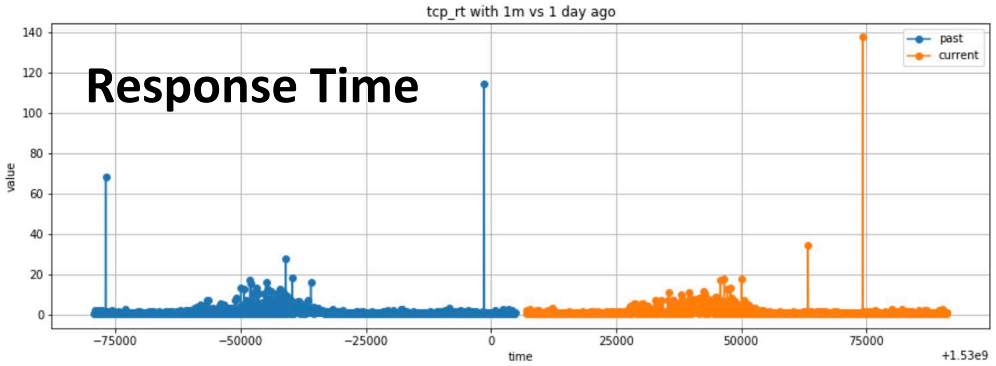
CDF of individual BP sizes
before and after the iBTune applies

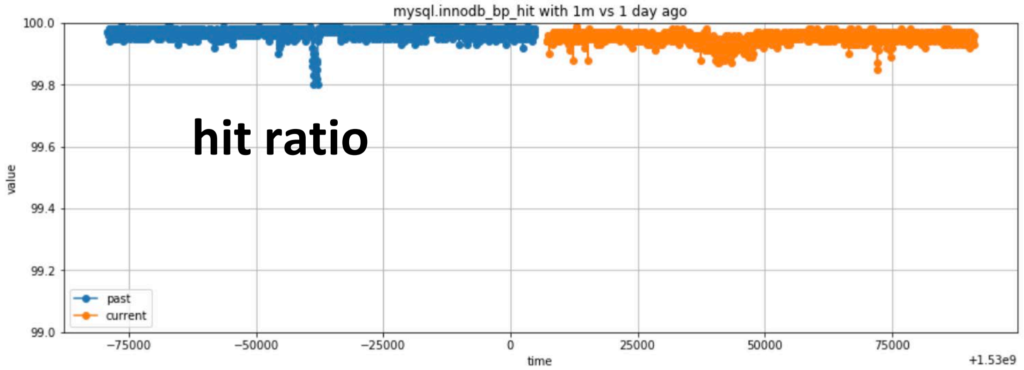


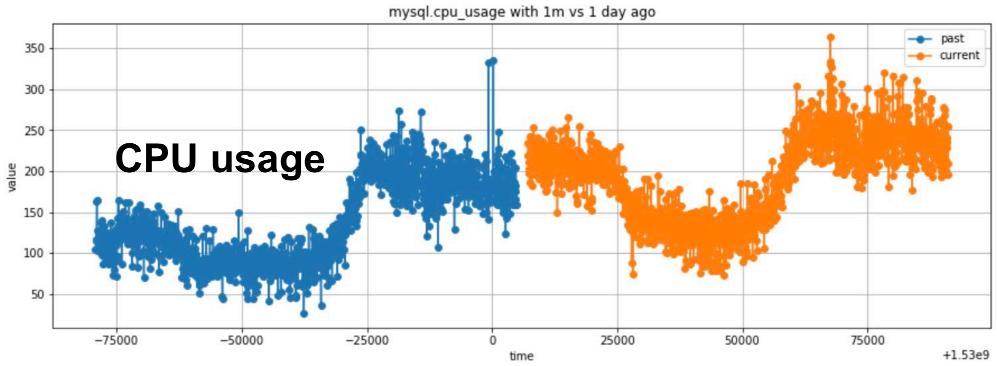
iBTune: Individualized Buffer Tuning for Largescale Cloud Databases

iBTune - Preliminary Attempt

Buffer pool (BP) size is sensitive to miss ratio: BP size is reduced from 188G to 80G when it's hit ratio is from 99.968% to 99.950%







- Challenge: Heuristic method (such as shrinking 10% each time) does not work, since we have to try many times, which makes the system unstable and is unacceptable for mission-critical applications.

Intuition：

- Calculate BP based on hit ratio (miss ratio) to avoid restarting system multiple times
- Confirm whether the BP size meets the requirement of SLA

Outline

Background and Motivation

Algorithm and System

Deployment and Result

iBTune – High level idea

達摩院
ALIBABA DAMO ACADEMY

Practical function $\frac{\log(mr_{target}) - \log(mr_{cur})}{\log(bp_{bp_{target}}) - \log(bp_{cur})} \approx -\alpha_i.$

F1(t_miss_ratio) = New BP size

Apply new BP size

DB instance

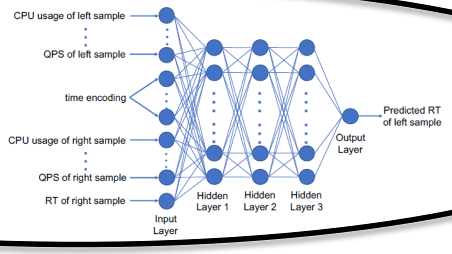
tolerate miss ratio (t_miss_ratio)

DB instance

F2(t_miss_ratio) = Response time <

Safe Response time (SLA)

Pairwise DNN



Guarantee SLA

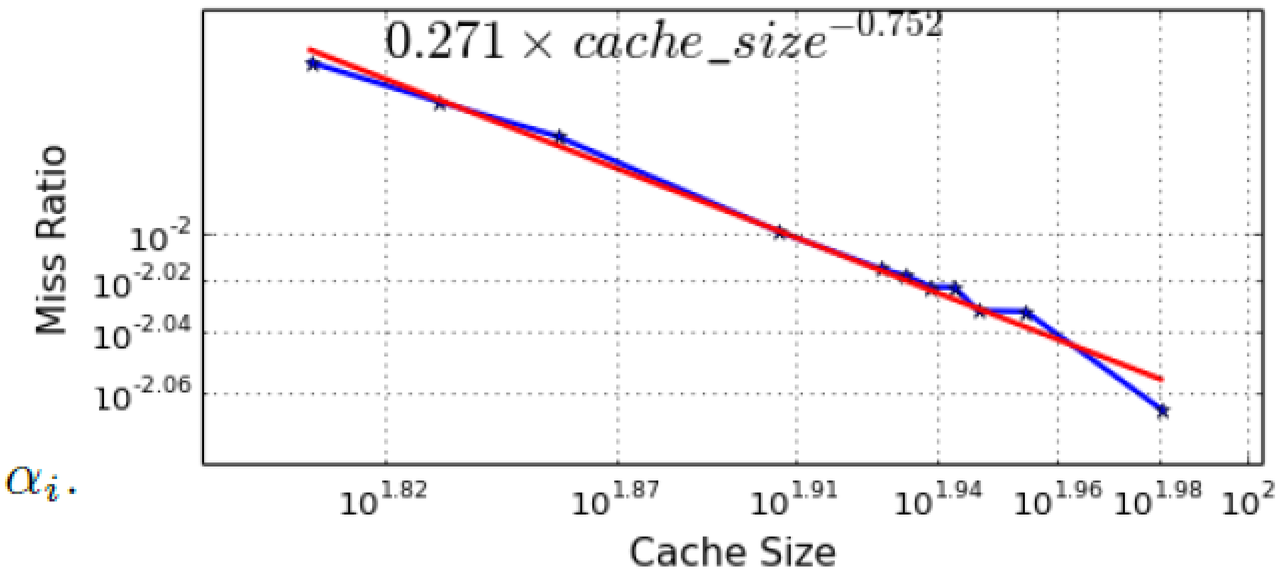
Finding BP=f(miss ratio)

達摩院
ALIBABA DAMO ACADEMY

- A number of empirical measurements on real systems have shown power law popularity distributions and follow that:

$$\frac{\log\left(mr_{target}\right) - \log\left(mr_{cur}\right)}{\log\left(bp_{bp_{target}}\right) - \log\left(bp_{cur}\right)} \approx -\alpha_i.$$

- Parameter α is obtained from the workload, which is 1.2 in our case.



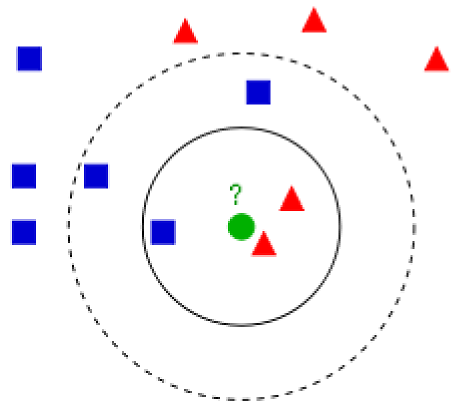
A large class of heavy-tailed requests with popularities following a power law distribution fits in our formulation.

Calculating tolerable miss ratio

達摩院
ALIBABA DAMO ACADEMY

- K-nearest-neighbors (DB instances)
- Find the nearest neighbors
  - Such as 6 in our case
  - Neighbor distance is calculated by similarity
- Calculate tolerable miss ratio
  - The weighted mean of the miss ratios of the k-nearest-neighbors

How to obtain k-nearest-neighbors?



$N(i) \leftarrow$ connected nodes of i on G;
for each instance j in $N(i)$ **do**
  if $mr_{(cur,j)} > mr_{(cur,i)}$ **then**
    $w_{ij} = \exp((E_j - E_i)^2/(2\sigma^2))$;
  else
    $w_{ij} = 0$;
  end
end

$$mr_{(tolerable,i)} \leftarrow \frac{\sum_{j \in N(i)} w_{ij} mr_{(cur,j)}}{\sum_{j \in N(i)} w_{ij}} ;$$

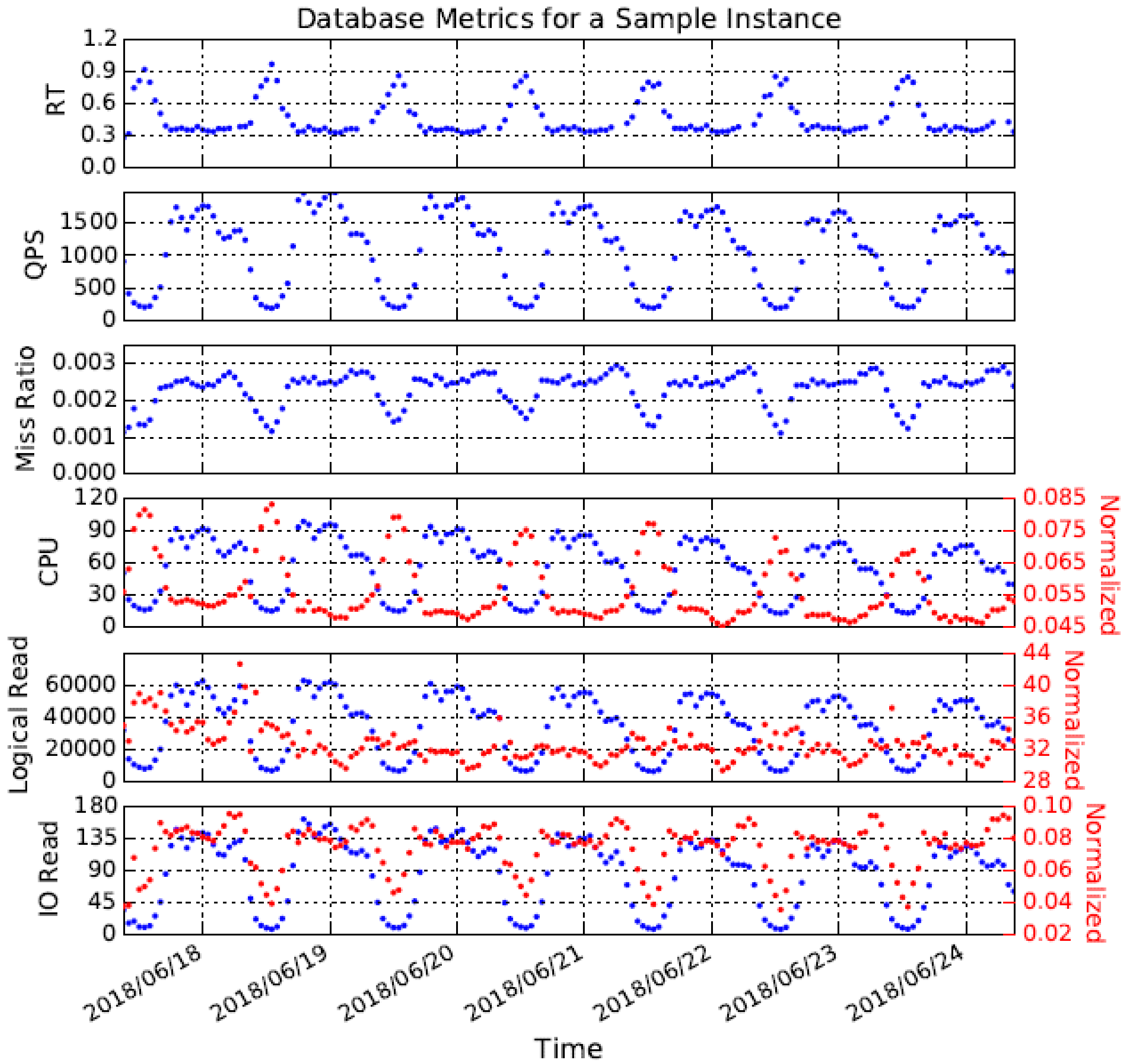
Calculating similarity

- Features
  - RT
  - QPS
  - miss ratio
  - CPU usage
  - logical read
  - io read

The last three metrics are divided by QPS

	Raw	Div By QPS
CPU usage:	0.055	0.093
Logical read:	0.127	0.394
IO read:	0.017	0.117

Pearson correlation coefficients

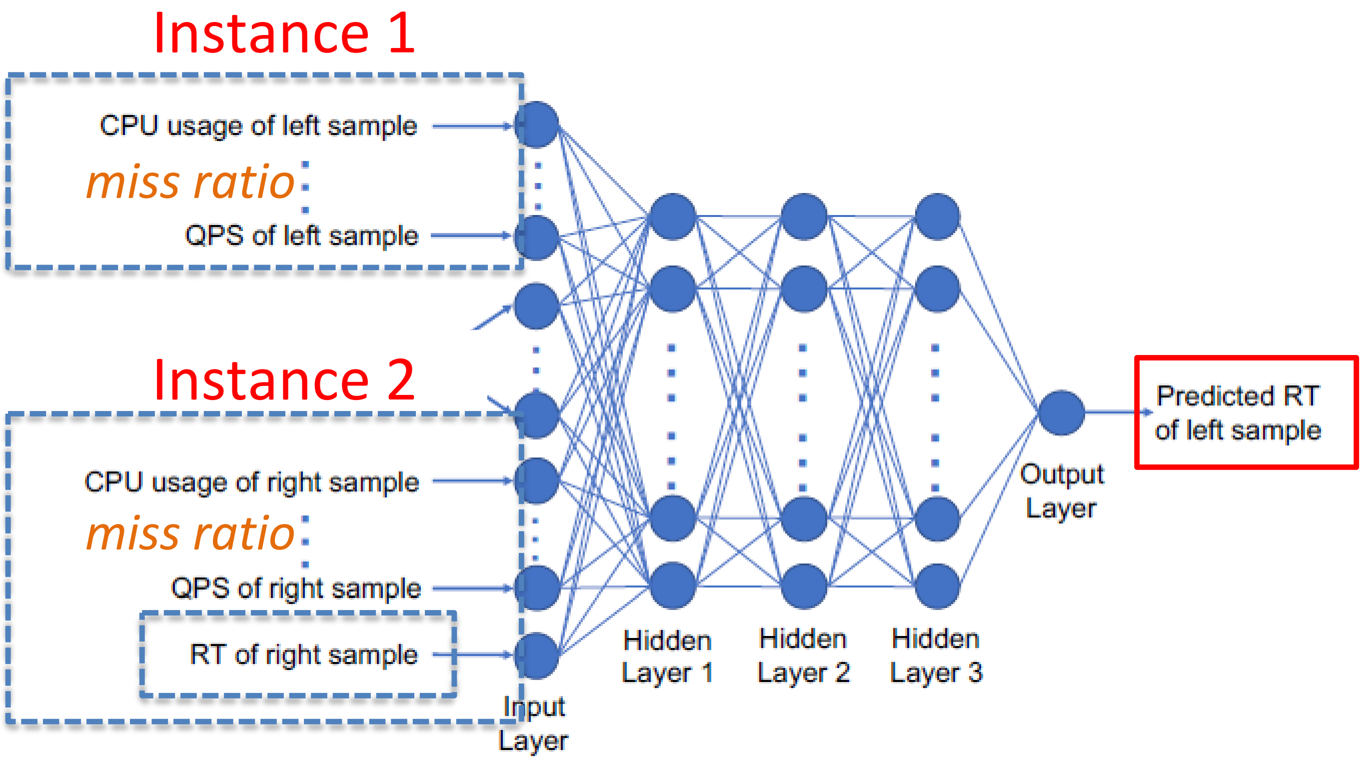


Till now, we can get the k-nearest-neighbors and tolerable miss ratio and calculate the $\boxed{\text{new BP size}}$

Predicting RT

Pairwise DNN: Predict the respond time (RT) based on the tolerable miss ratio

- Training
  - Input: two instances' metrics + right instance's RT
  - Output: Left instance's RT
- Predicting
  - input: A instance's metrics X 2 except miss ratio
  - Output: RT corresponding to tolerate miss ratio
- The granularity of metric is a day

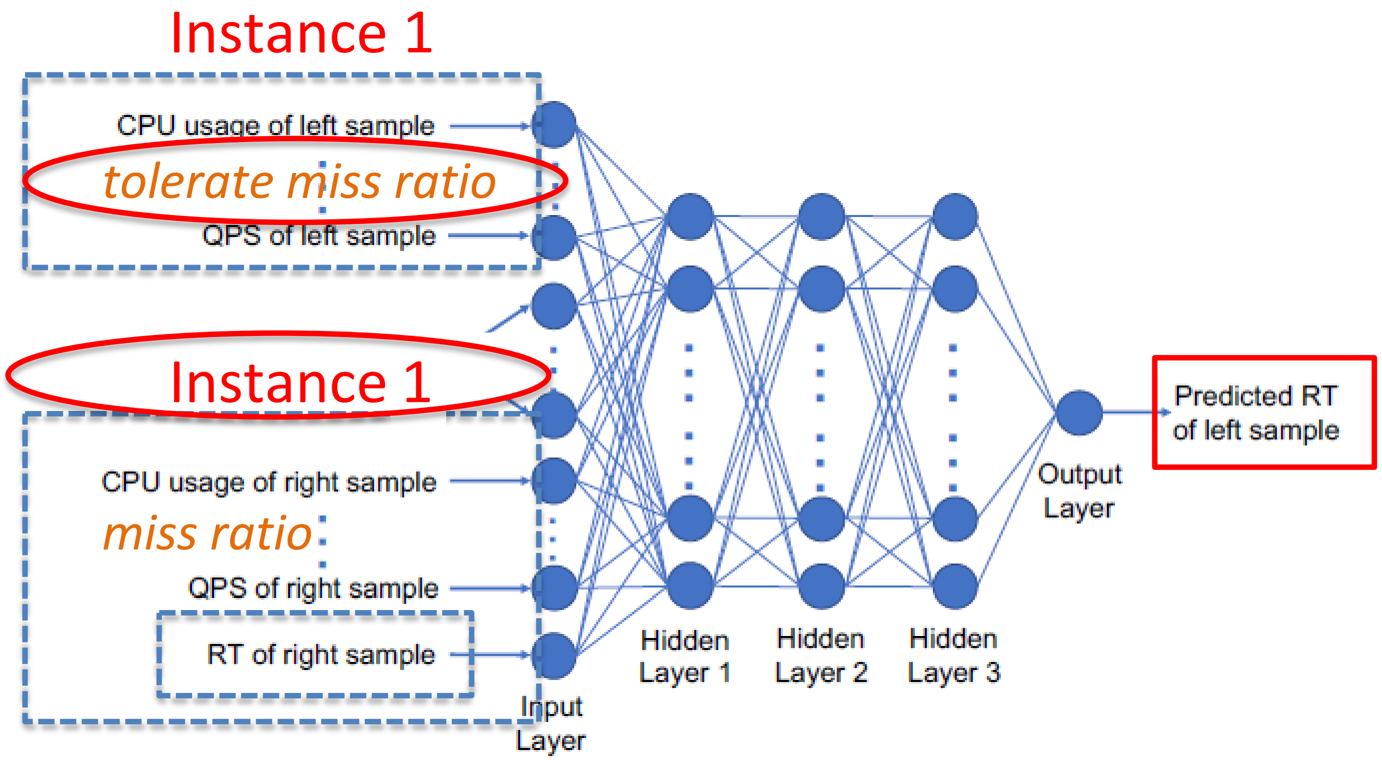


The predicted RT is compared with the safe SLA (RT)

Predicting RT

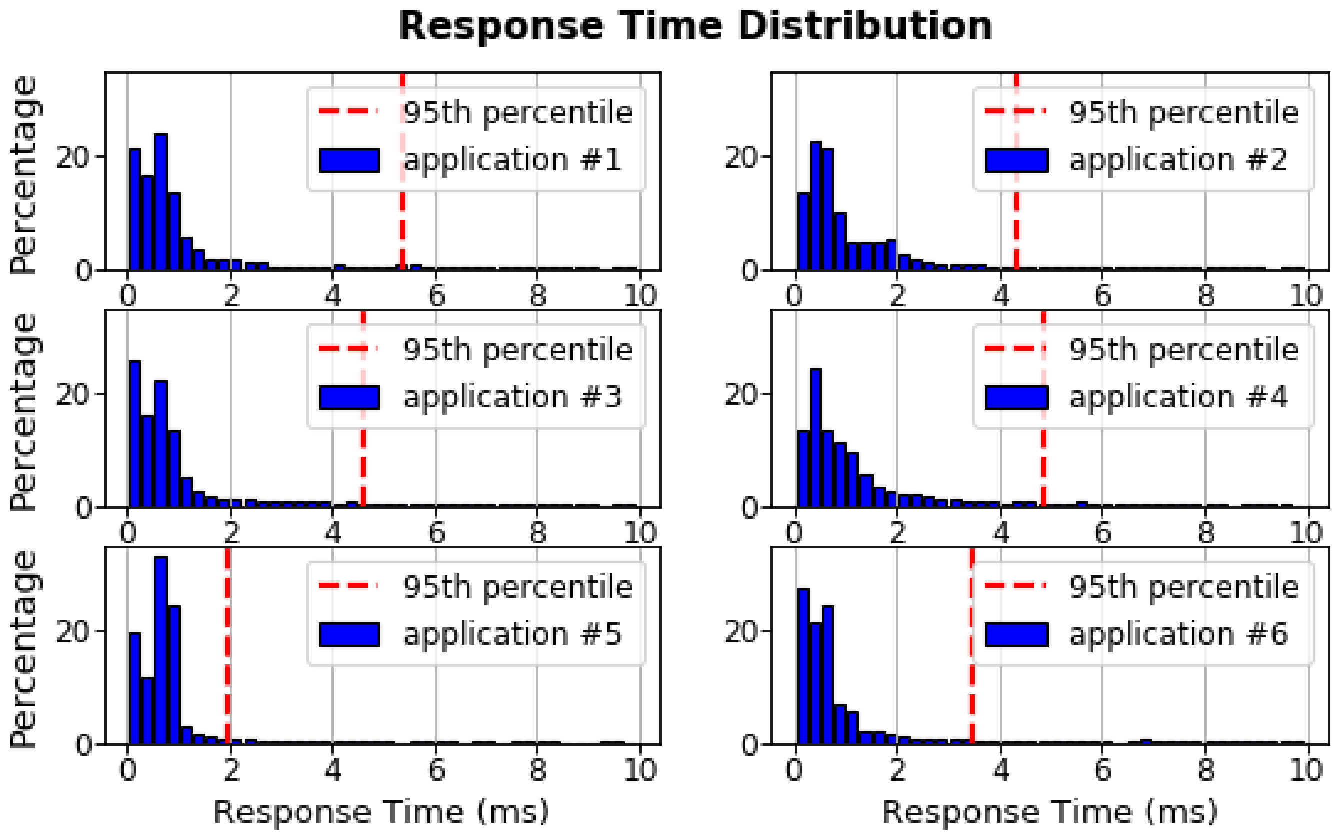
Pairwise DNN: Predict the respond time (RT) based on the tolerable miss ratio

- Training
  - Input: two instances' metrics + right instance's RT
  - Output: Left instance's RT
- Predicting
  - input: A instance's metrics X 2 except miss ratio
  - Output: RT corresponding to tolerate miss ratio
- The granularity of metric is a day



The predicted RT is compared with the safe SLA (RT)

Safe SLA（RT）



Determine the safe RT for different applications

Group all the instances into different applications, and find the 95% percentile of the response times in each group as the corresponding safe limit for that application.

Outline

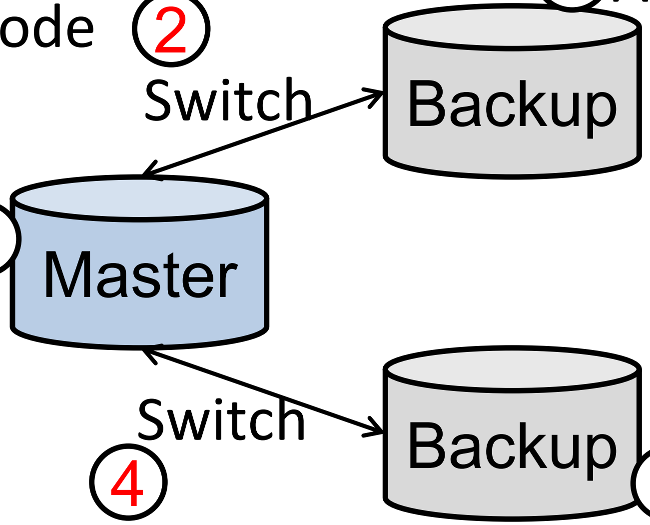
Background and Motivation

Algorithm and System

Deployment and Result

System halt avoidance

Based on X-Paxos: high availability protocol implementation at Alibaba



1 Adjust backup node's BP size

It leads to 10~20 sec system halt to backup node, without affecting system service.

2 Switch master and backup after the backup node recovers

3 Monitor the new master for 24 hours

4 If the new master is abnormal, i.e., the number of slow SQLs increases, rollback will be trigged.

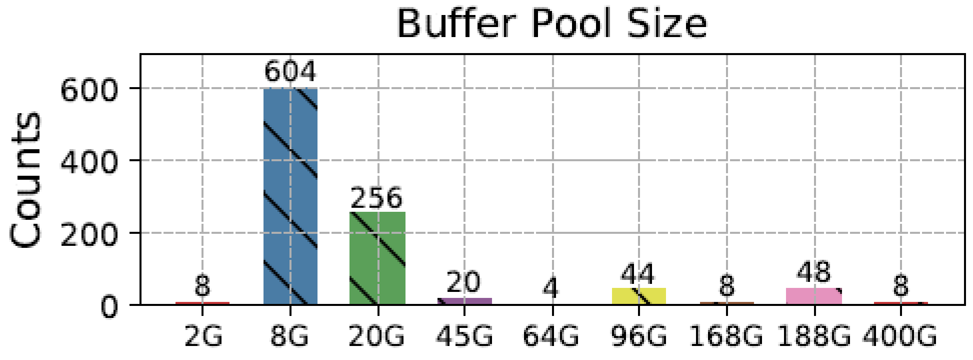
5 If the new master works fine during the following 24 hours, backup nodes' BP will be adjusted.

Evaluation

达摩院
ALIBABA DAMO ACADEMY

- All results are from our product environment
- X-Engine: MySQL compatible database based on LSM-Tree storage engine
- With high performance Paxos implementation
- Pairwise DNN: 100K data samples

	Taobao	Tmall	Youku	Fliggy	Others
Select	5245/s	2815/s	1017/s	22930/s	120/s
Insert	4222/s	0	1/s	1520/s	10/s
Update	0	30/s	315/s	4/s	980/s
Delete	2708/s	0	10/s	0	0



1,000 sample instances scattered across different applications

iBTune has been deployed on 10,000 database instances

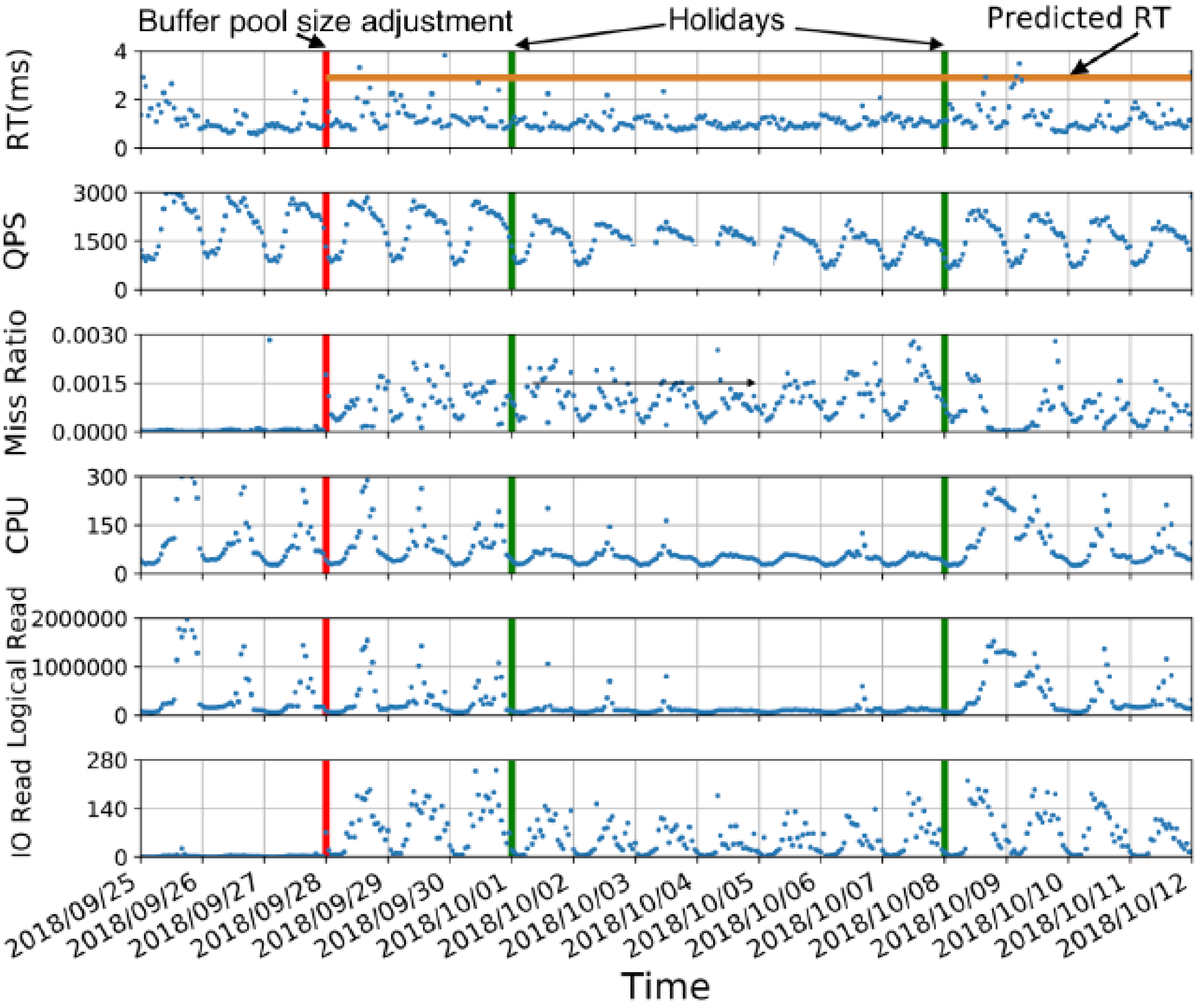
Memory saving : ~17%

Single instance

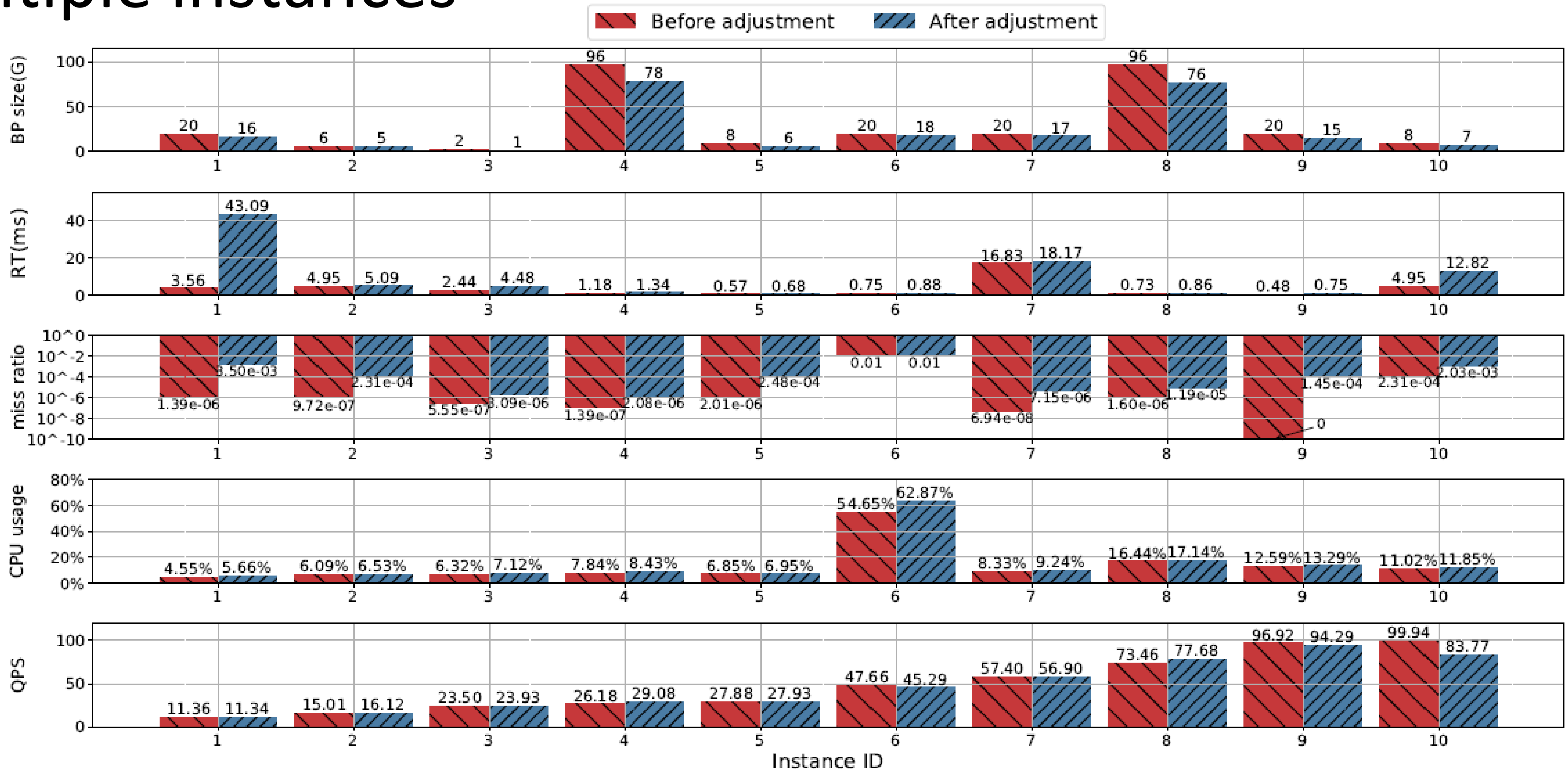
Performance before and after BP adjustment (45G->21G) during holidays and workdays:

- Red line is the time when BP size is adjusted
- Green lines show the holiday which is 7-days
- Predicted RT: only 3 points exceeded which is acceptable

The IO read metric is the real IO, since all our DB instances turn on direct IO



Multiple instances



10 representative instances. The memory saving ranges from 50% to 10%, which strongly supports that a single number does not fit all. Instance 1 has a large increase in RT after the adjustment. We find that there is one query that consumes 99.97% of the total response time. The lookup value in WHERE condition changes for this query.

Conclusion & Future Work

- **iBTune has been deployed on 10,000 database instances with memory saving : ~17%**
- Future work
  - Cache preload
    - Backup node needs run SQLs to load data into cache after BP adjustment
    - Perform switching after preload
  - Buffer increase
    - Currently reply on DBA
    - Automatic increase buffer
  - Multiple parameters tuning
    - DBMS configure file

達摩院
ALIBABA DAMO ACADEMY

Thanks