Reducing Web Latency through Dynamically Setting TCP Initial Window with Reinforcement Learning

Xiaohui Nie [†], Youjian Zhao[†], Dan Pei [†], Guo Chen[*], Kaixin Sui[¶] , Jiyang Zhang[‡]

清華大學 Tsinghua University [†]
湖南大學 HUNAN UNIVERSITY [*]
Microsoft Research 微软亚洲研究院 [¶]
Baidu 百度 [‡]

Motivation and Background

- **Web latency matters!**

Google latency increases 100 … number decrease 0.2%~0.6%[1]

bing … se 1.2% [2]

amazon … sales [3]

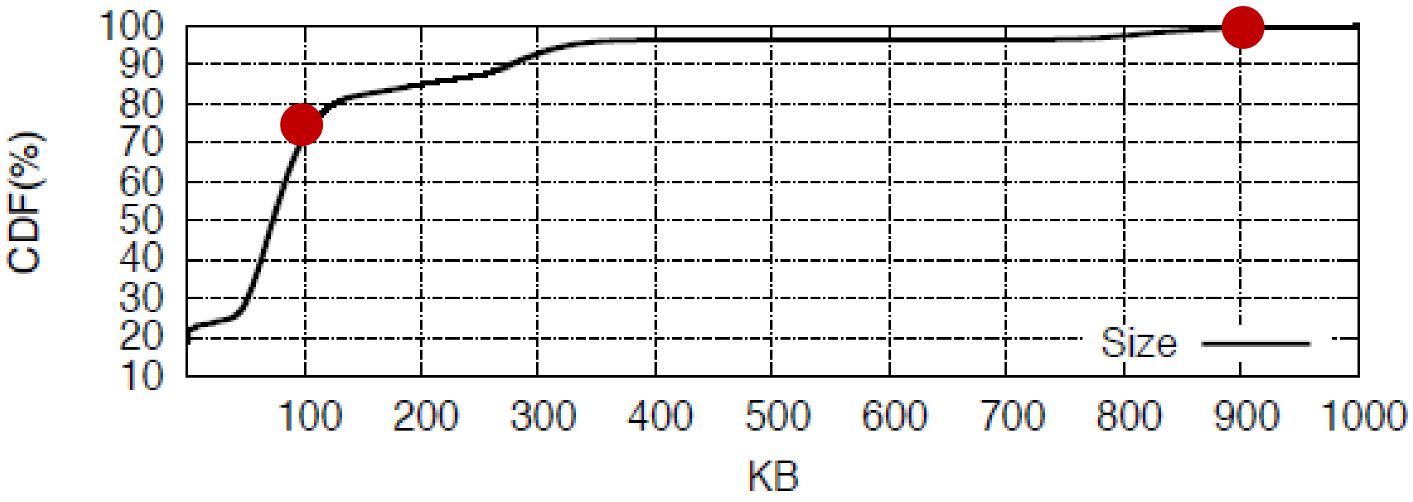
[1] J.Brutlag. (June, 2009). Speed matters for Google web search.

[2] E.Schurman,J.Brutlag.(June,2009).The User and Business Impact of Server Delays, Additional Bytes and Http Chunking in Web Search.

[3] Latency Is Everywhere And It Costs You Sales. https://goo.gl/bRi5Xs

Motivation and Background

- **Currently, data transmission of most web services are based on TCP.**
- **Most flows of web service are short.**
  - 99% flows are smaller than 100KB [Greenberg SIGCOMM'09]
  - 70% flows of Baidu mobile search service are smaller than 100KB.

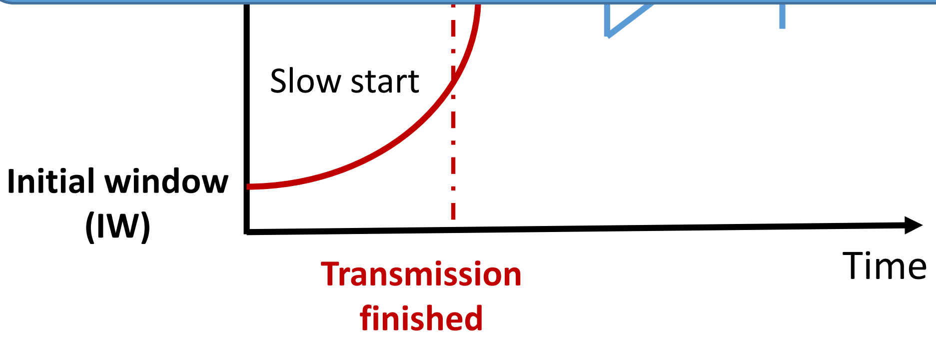


Motivation and Background

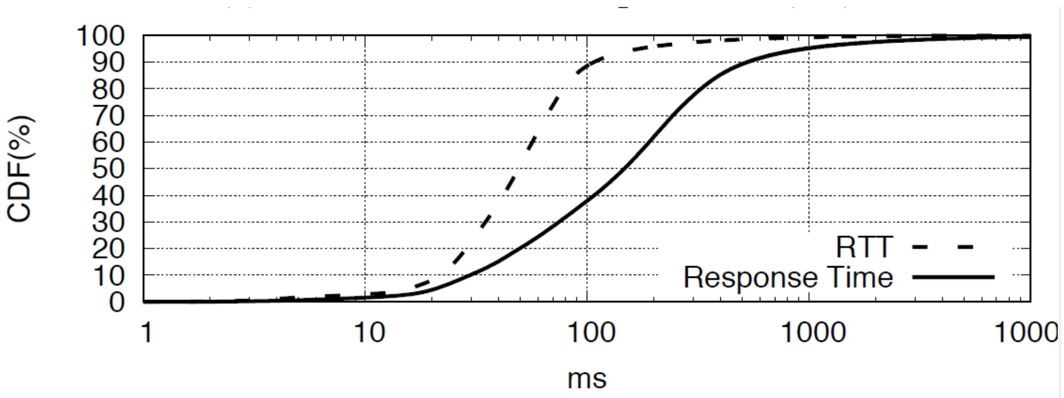
- **Short flow's transmission is slow because of TCP's *flow startup problem* [RFC6077]**
  - Slow-start mechanism with a conservative initial window (IW) to probe the bandwidth during the transmission.

TCP State	% of flows

The basic problem is end-systems don't know how to set the IW.



- **Inefficient bandwidth utilization**
- **Multiple RTTs for short flow**



(b) The distribution of RTT and response time (ms). The x-axis logscale.

Motivation and Background

- **Why IW is hard to choose?**
  - Large IW -> network congestion; Small IW -> long latency
  - No knowledge to learn.
    - When connection established, the sender does not know current network condition.
    - Only one kind of IW has been used in history.

- **One static IW is suboptimal.**
  - Different users have different network conditions.
  - For one user, its network condition could changes over time.

Network	2G	3G	4G	Wi-Fi(2.4GHZ)
RTT	300~1000ms	100~500ms	10~100ms	10ms ~100ms
Bandwidth	100–400 Kbit/s	0.5–5 Mbit/s	1–50 Mbit/s	25 Mbit/s
Ideal Cwnd	3~16	5~223	1~446	2~223

Ideal Cwnd = Bandwidth * RTT

Related Works

- **Many prior works have been done to improve TCP performance.**
  1. **New congestion control algorithm (e.g. TCP Tahoe, Reno, Bic, Cubic, BBR, PCC, Remy)**
     - Pros: Quickly converge to the right available bandwidth after transmission begins.
     - Cons: Flow startup problem exists.

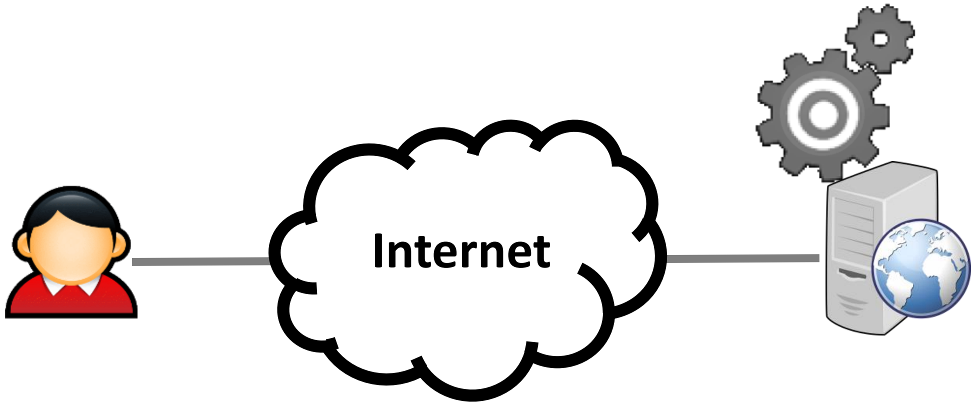
The flow startup problem is only mitigated but not directly solved

     - Cons: one standard value is suboptimal.

  3. **Aggressive startup (e.g. Jump start** [FLDnet07]**):**
     - Pros: fast transmission.
     - Cons: hardly seen deployed; may cause damage to the other co-existing flows.

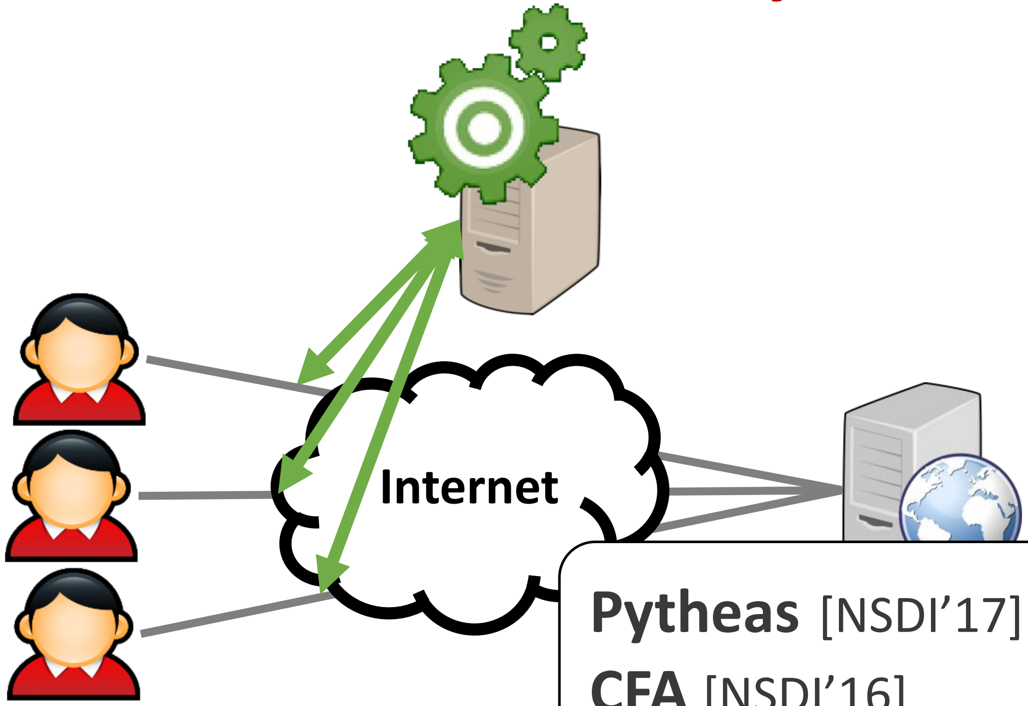
Data-driven approach is promising

Local data of single flow



Classic approaches

Global data of many flows

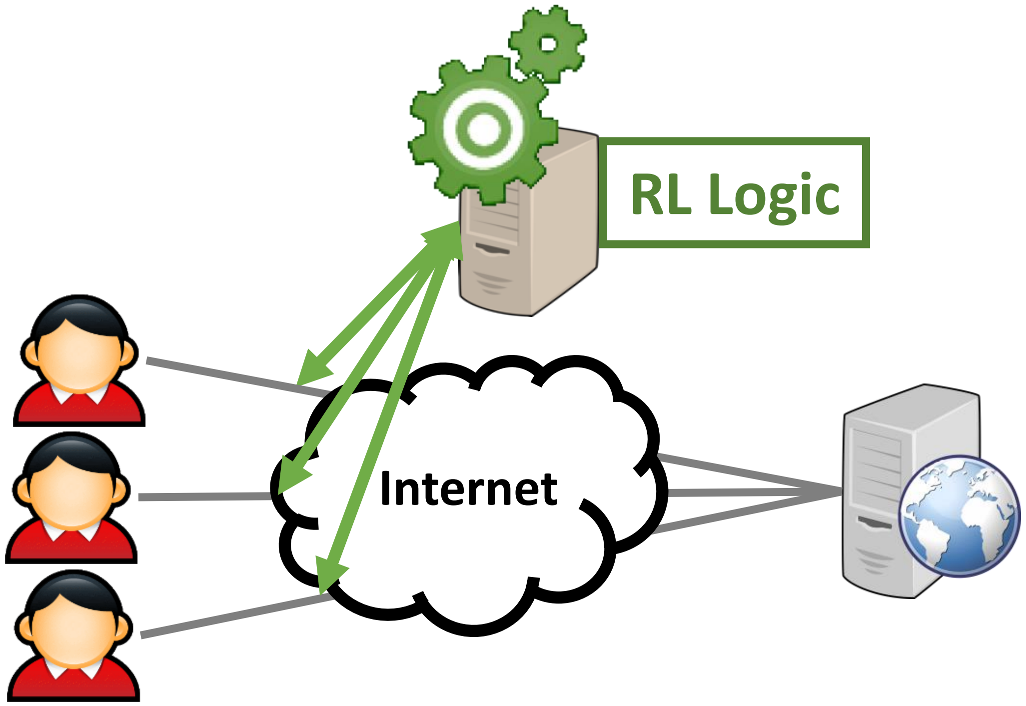


Pytheas [NSDI'17]
CFA [NSDI'16]
VIA [SIGCOMM'16]
CS2P [SIGCOMM'16]

Data-driven approach

Our Idea

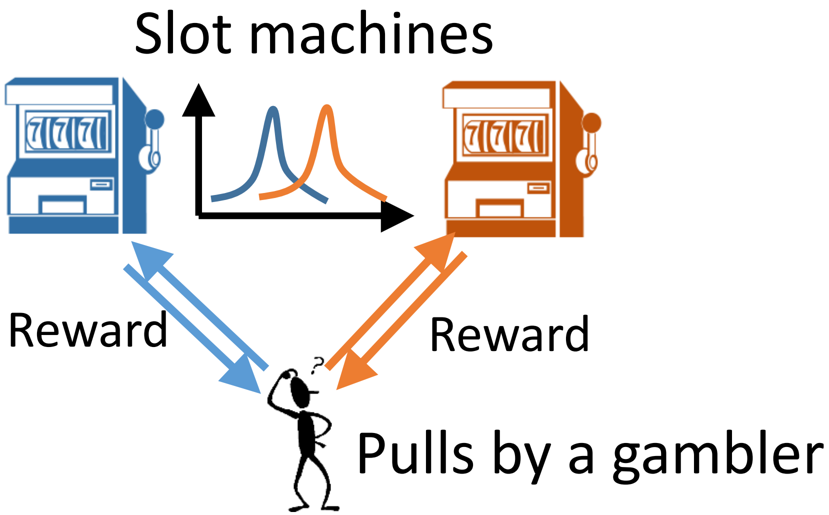
Model IW setting as a Reinforcement Learning problem (Real-time Exploration and Exploitation).



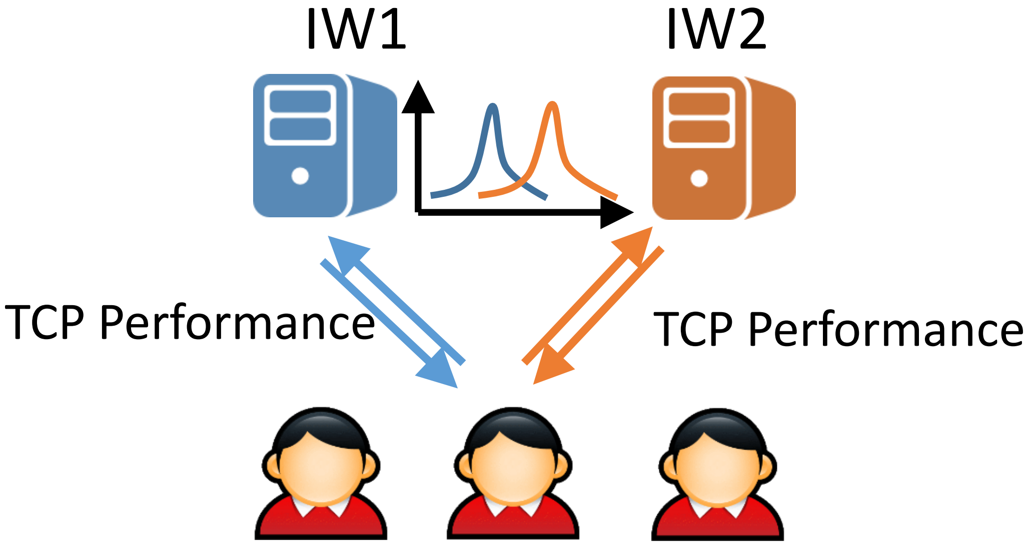
A classic problem in RL

Multi-armed bandit problem

Goal: Maximize mean rewards given a limited amount of pulls



Goal: Optimize mean TCP performance for a limited amount of flows



The challenges in practice

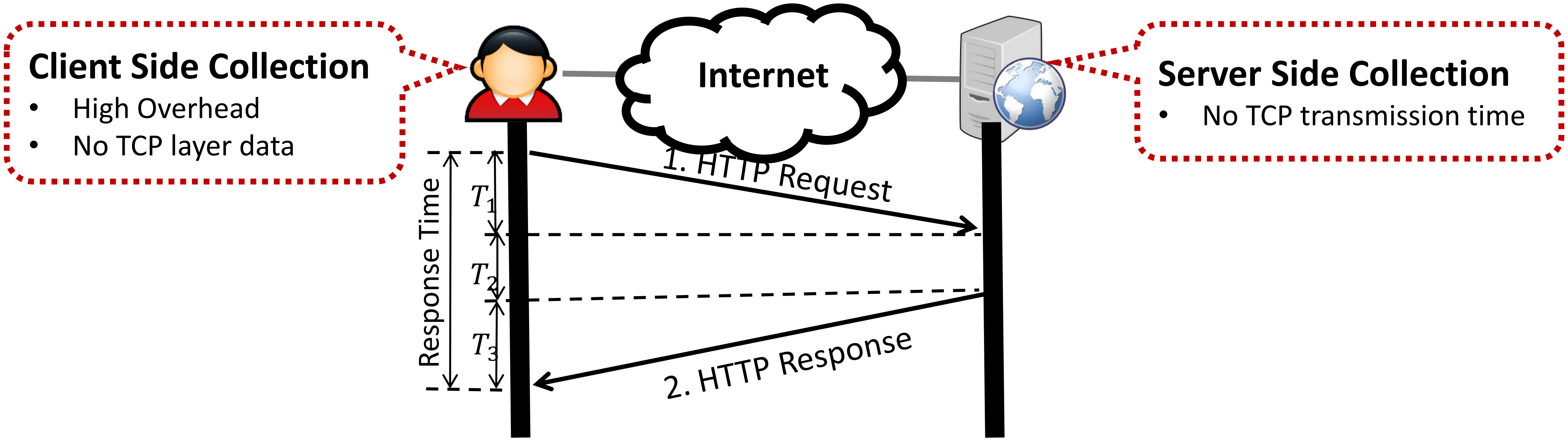
- **Challenge #1:** How to measure TCP performance data on the server side?

- **Challenge #2:** How to apply RL methods on highly variable and non-continuous network conditions of the Internet?

- **Challenge #3:** How to search for the optimal IW from a large window space quickly?

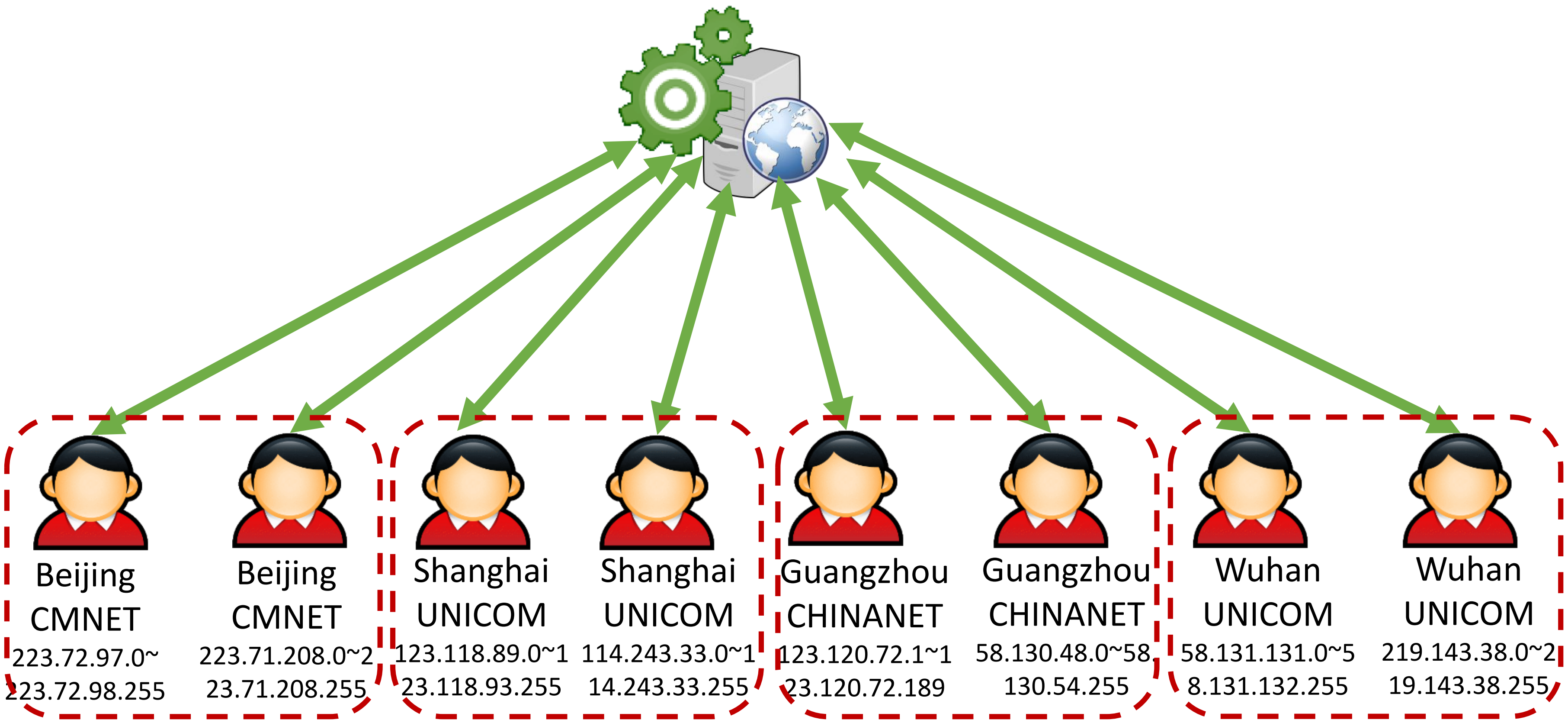
Challenge #1: How to measure TCP performance data on the server side

- **RL needs global fresh data**



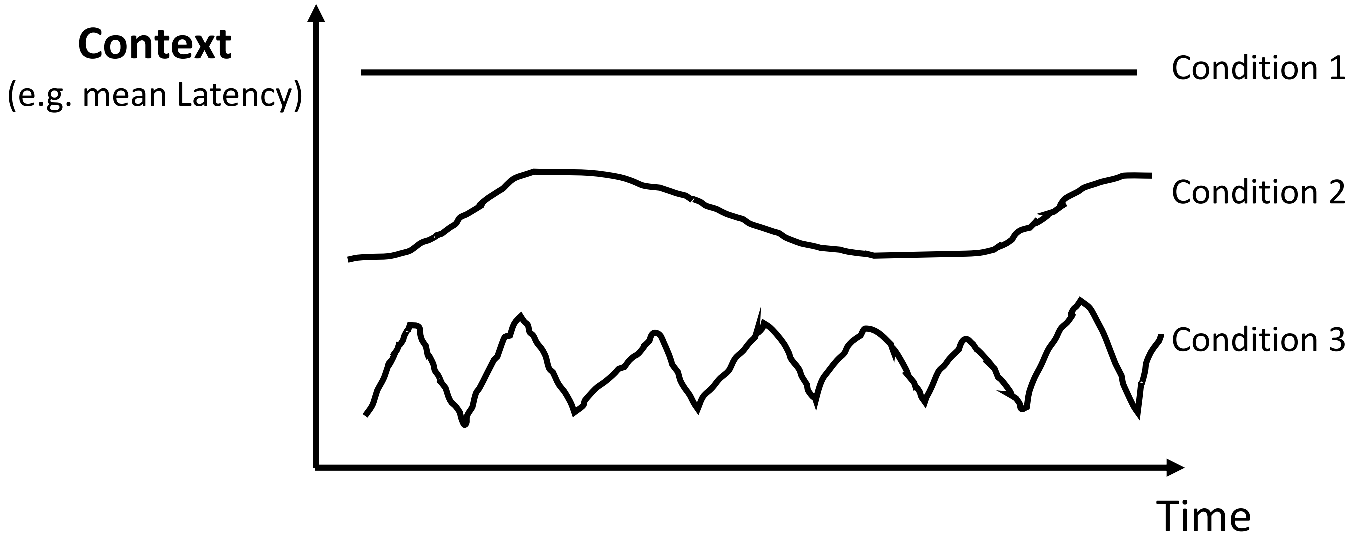
Challenge #2: How to apply RL methods on highly variable and non-continuous network conditions of the Internet?

- **Users network condition are variable**



Challenge #2: How to apply RL methods on highly variable and non-continuous network conditions of the Internet?

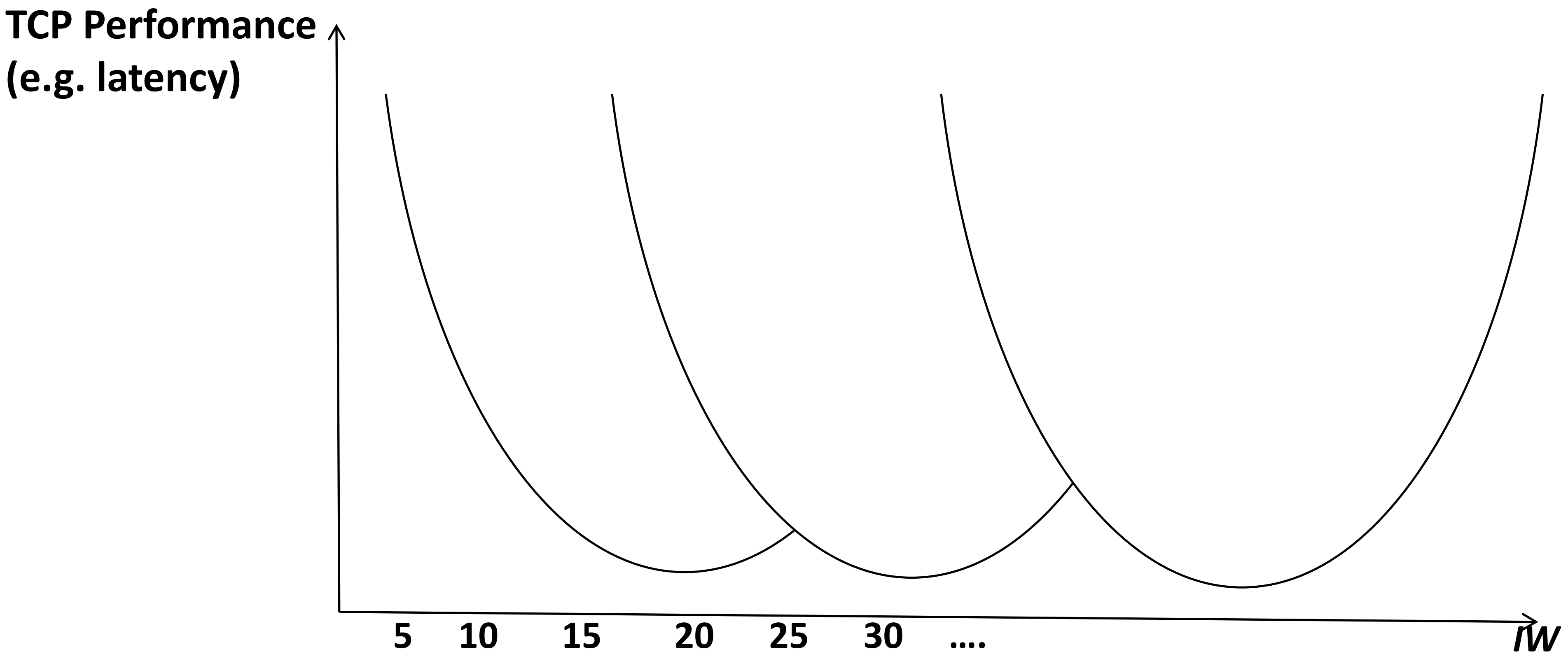
- **RL requires continuity of the context that affects the reward of the decision**



- **Problem: not every user group can satisfy the RL's requirement**
  - Fine-grained -> not enough data samples
  - Coarse-grained -> suboptimal performance

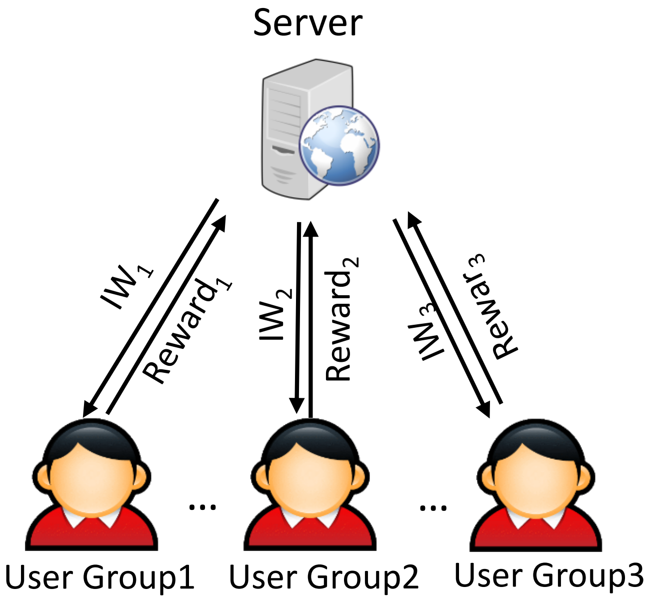
Challenge #3: How to search for the optimal IW from a large window space quickly?

- **RL is a process of searching optimal value from the decision space.**

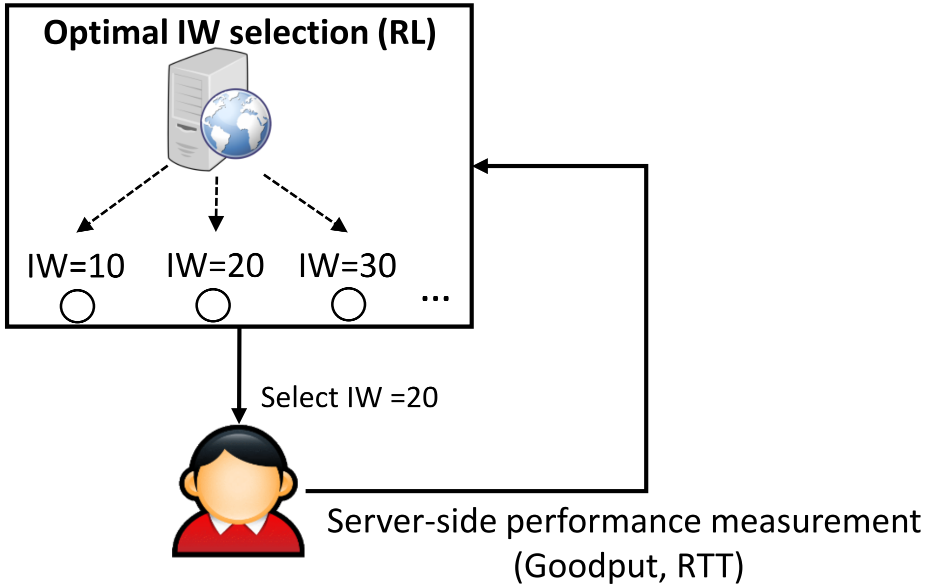


SmartIW: Group-based RL

- **The key ideas:**
  1. Using **different IWs** for **different user groups**
  2. For one user group, **wisely learning** the optimal IW by **RL**



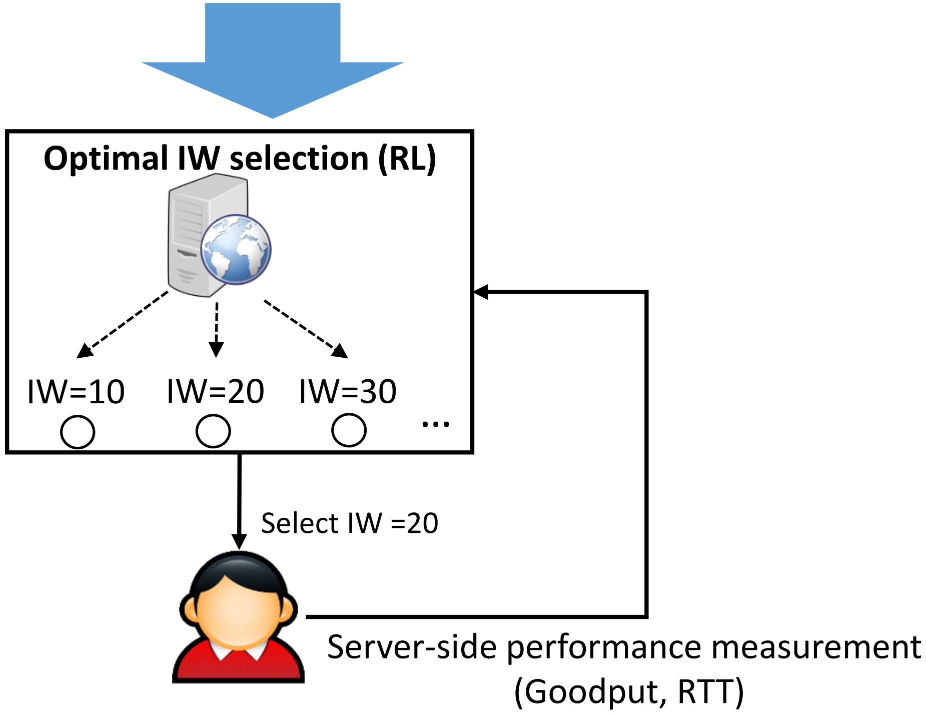
(a) Using different IWs for different user groups



(b) The logic of IW selection for one user group

RL Algorithm

Discounted Upper Confidence Bound



- **Reward:**
  - Goodput & RTT

$$X_s(i) = \alpha * \frac{Goodput_s(i)}{Goodput_{max}} + (1-\alpha) * \frac{RTT_{min}}{RTT_s(i)}$$

- **Decision Space:**

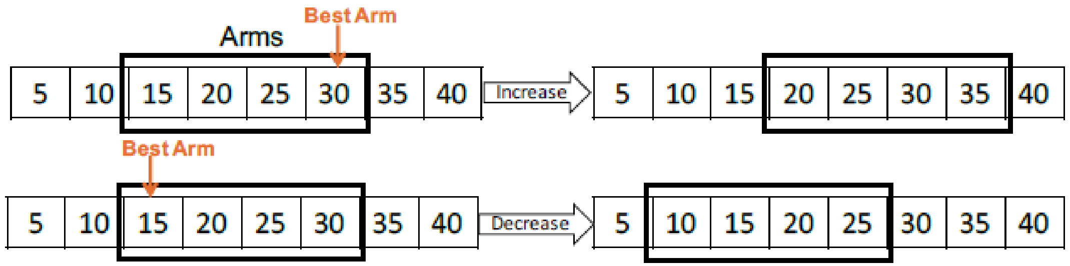


Fig. 4: The procedure of the sliding-decision-space method.

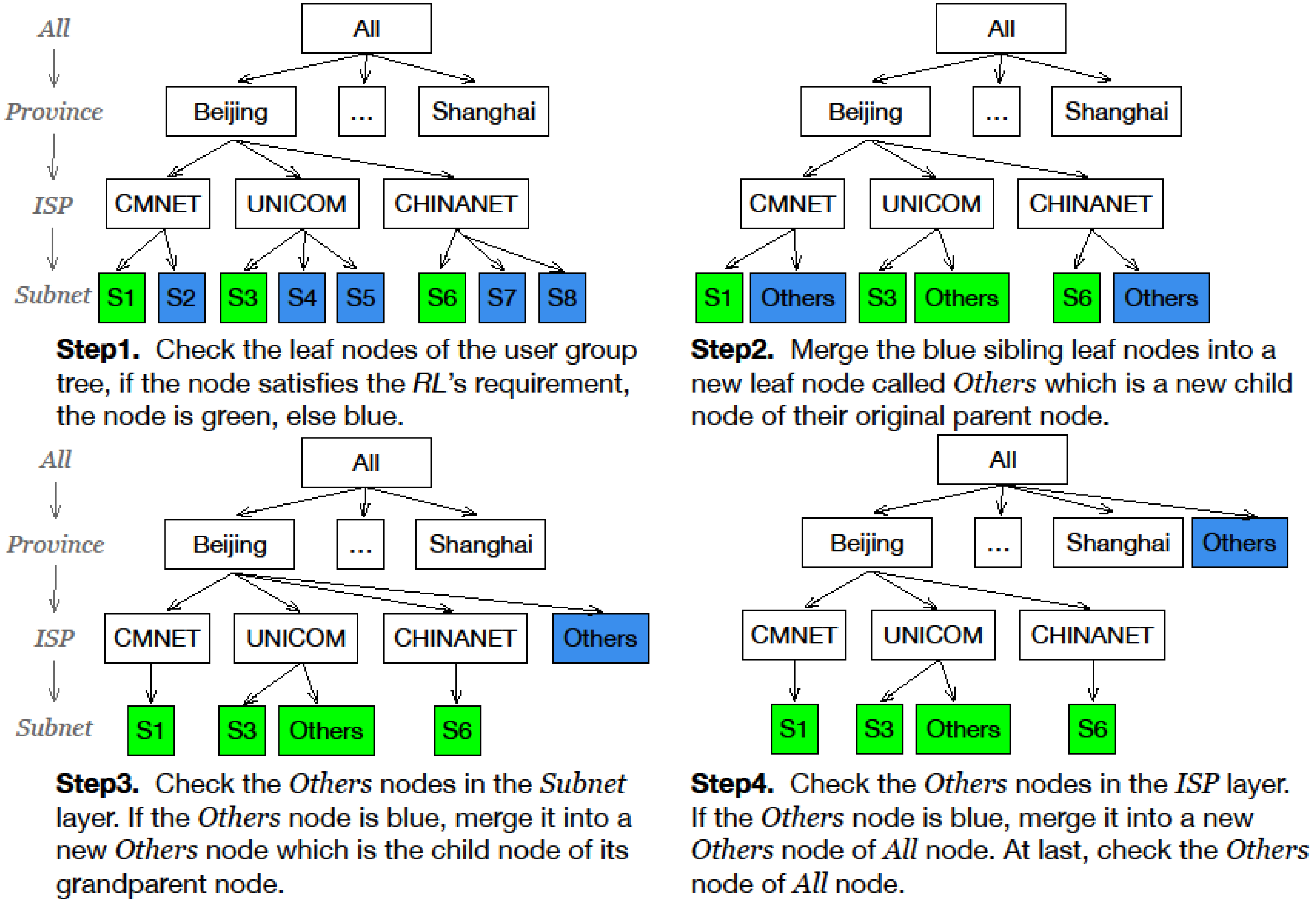
User grouping

- **RL's requirement:**
  - Continuity of context (netwc
- **Network Jitter:**

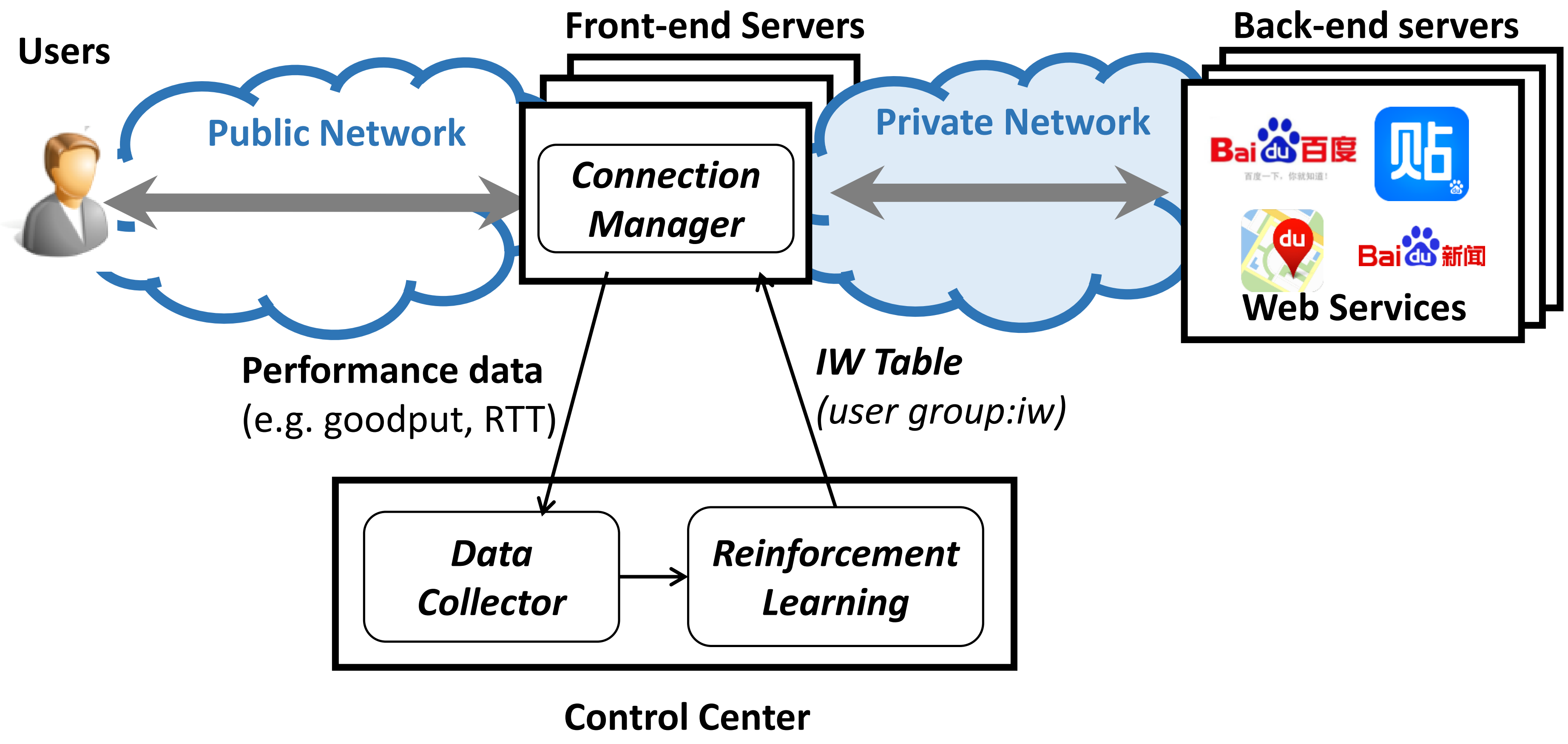
$$J = \frac{\sum_{s=2}^{n} |X_s - X_{s-1}|}{n-1}$$

- **User grouping:**
  - Bottle-up searching
  - Find the finest user groups w RL's requirement:

$$J \leq T$$

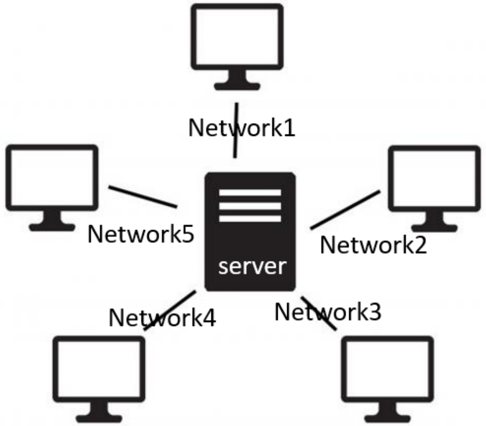


Implementation



Evaluation

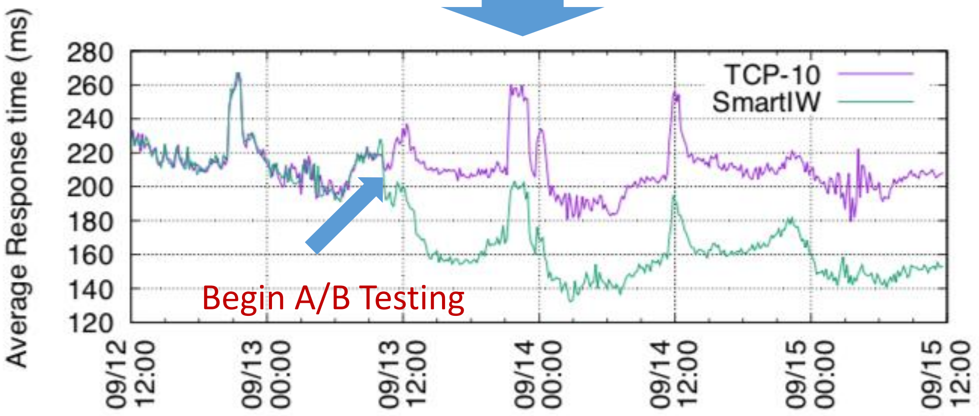
- **Online experiment:**
  - SmartIW can continuously bring about **23% improvement** of the average response time.



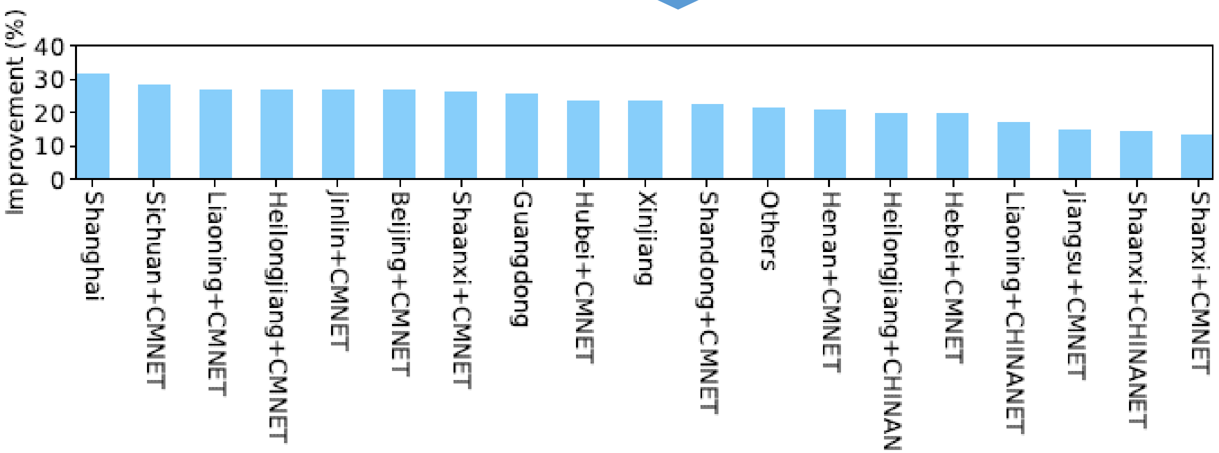
- **Testbed experiment:**
  - Both user grouping and reinforcement learning can help improve the network performance by **29%**.
  - Direactly using a aggressive IW is a bad choice.

Online experiment

Continuously improve overall performance by 23%



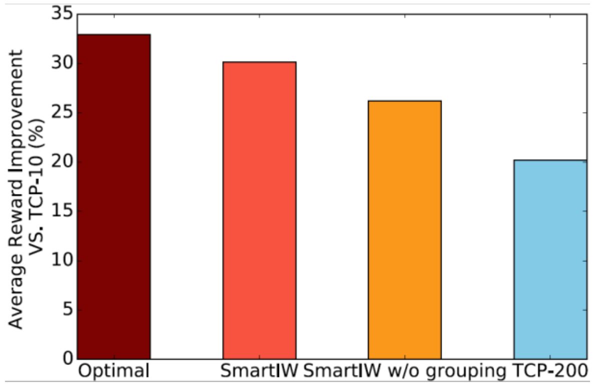
Each group has 15%~32% performance improvement.



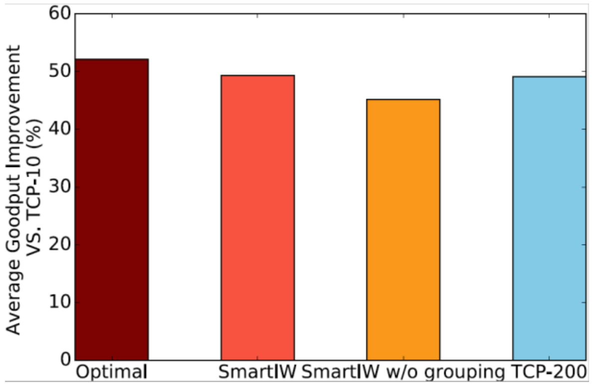
19 user groups (3 Provinces, 15 Province+ISP, 1 Others)

Testbed Experiment

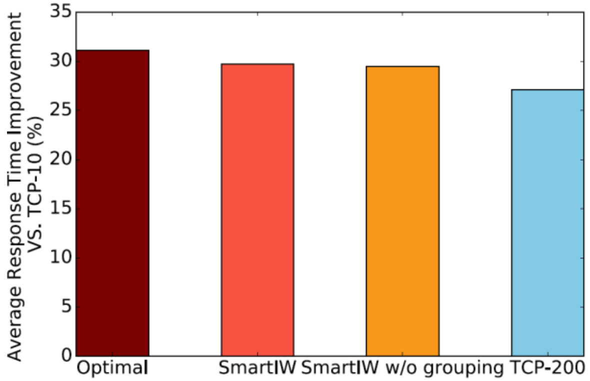
- **Replay user groups' 24*hour network condition.**
- **5 Schemes:**
  - TCP-10
  - TCP-200
  - *SmartIW*
  - *SmartIW w/o grouping*
  - Optimal



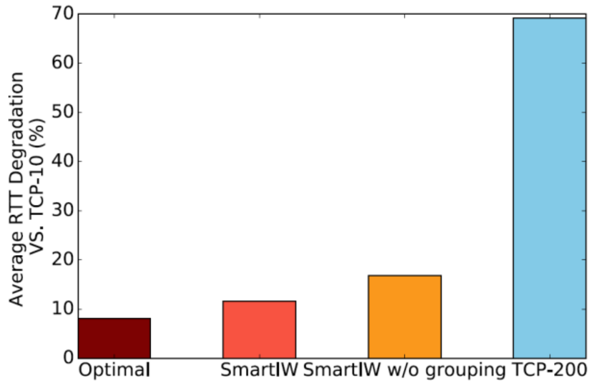
Reward = 0.8*Goodput + 02*RTT



Goodput



Response Time

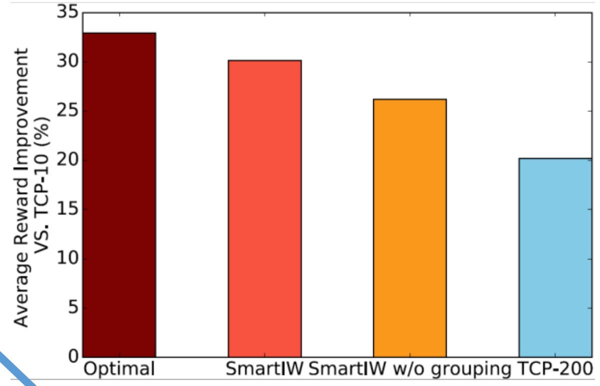


RTT

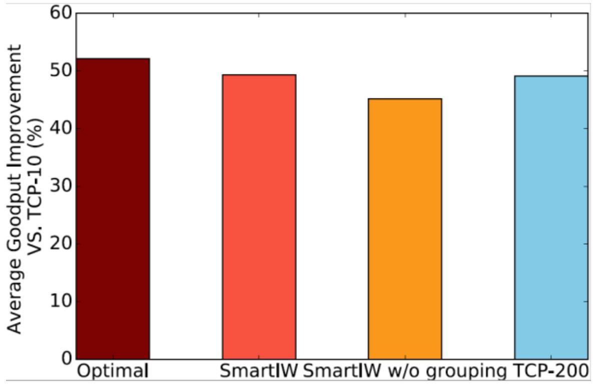
Testbed Experiment

- **Replay user groups' 24*hour network condition.**

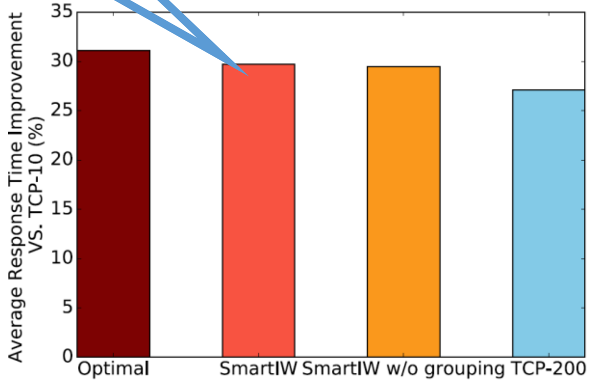
1. SmartIW improves average response time by 29%



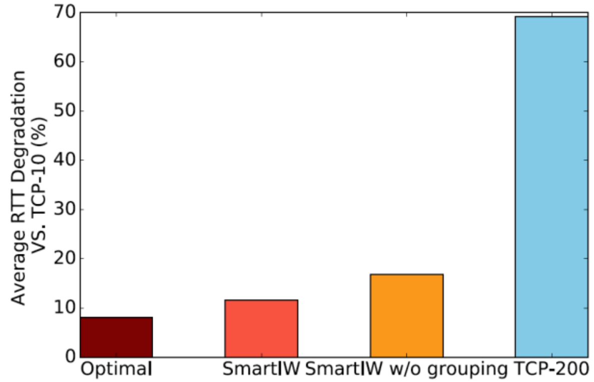
Reward = 0.8*Goodput + 02*RTT



Goodput



Response Time



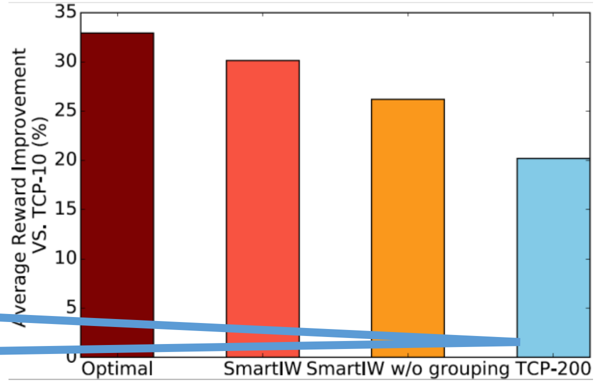
RTT

Testbed Experiment

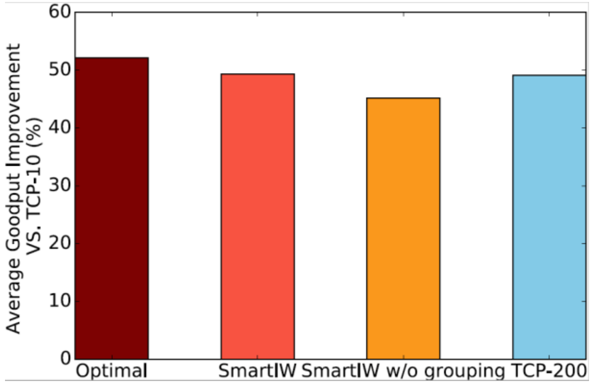
- **Replay user groups' 24*hour network condition.**

1. SmartIW improves average response time by 29%

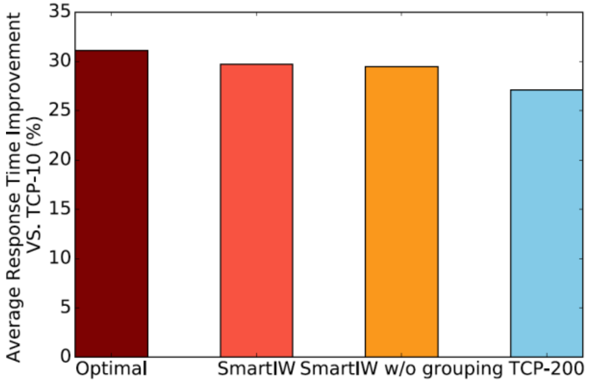
2. Directly using an aggressive IW is suboptimal, causing critical congestion.



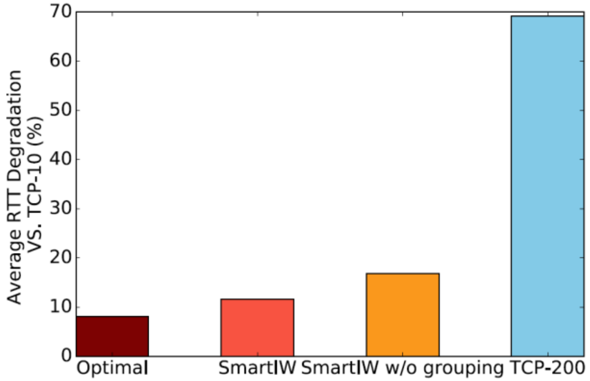
Reward = 0.8*Goodput + 02*RTT



Goodput



Response Time



RTT

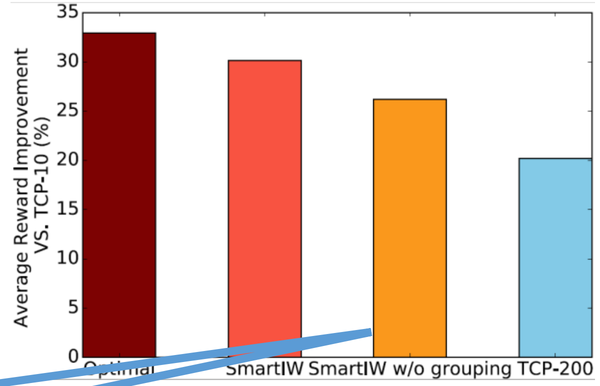
Testbed Experiment

- **Replay user groups’ 24*hour network condition.**

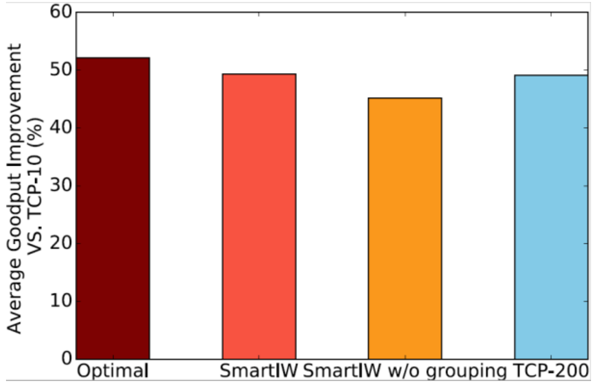
1. SmartIW improves average response time by 29%

2. Directly using an aggressive IW is suboptimal, causing critical congestion.

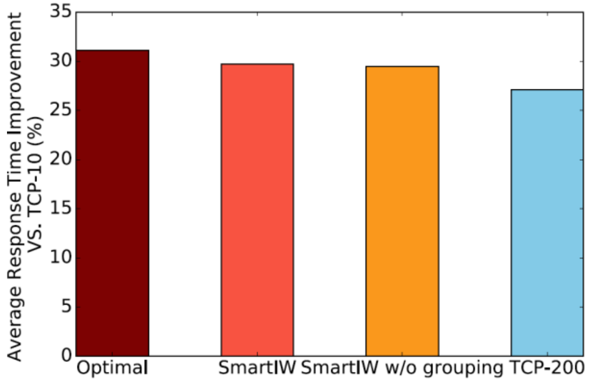
3. Both grouping and RL can improve the performance.



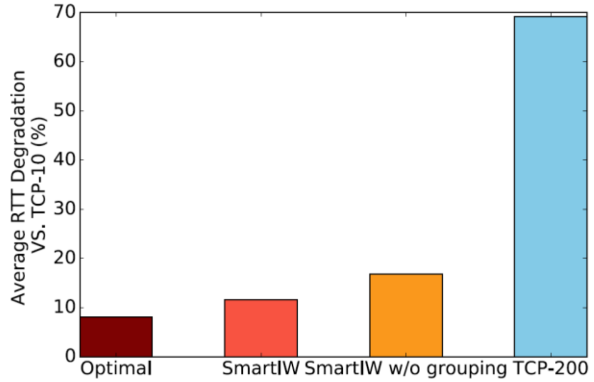
Reward = 0.8*Goodput + 02*RTT



Goodput



Response Time



RTT

Conclusion

- **TCP flow startup problem is very obvious in the realworld.**

- **We propose a system called *SmartIW* that can set TCP *IW* at server side smartly using group-based reinforcement learning.**

- **Testbed and Online experiment prove our system works well.**
  - Online: 23% performance improvement
  - Testbed: 29% performance improvement

Thanks

Q&A?