# TCP WISE: One Initial Congestion Window Is Not Enough

Xiaohui Nie[$], Youjian Zhao[$], Guo Chen[†], Kaixin Sui[†], Yazheng Chen[$], Dan Pei[$], MiaoZhang[‡], Jiyang Zhang[‡]

清華大學 Tsinghua University [$]

Microsoft Research 微软亚洲研究院 [†]

Baidu 百度 [‡]

# Motivation

- **Web latency matters!**

Google — latency increases 100ms ~400ms, query number decrease 0.2%~0.6%[1]

bing — latency increases 50ms, revenue decrease 1.2% [2]

amazon — every 100ms of latency cost them 1% in sales [3]


Users are more likely to perform clicks on the fast page [SIGIR 2014]

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

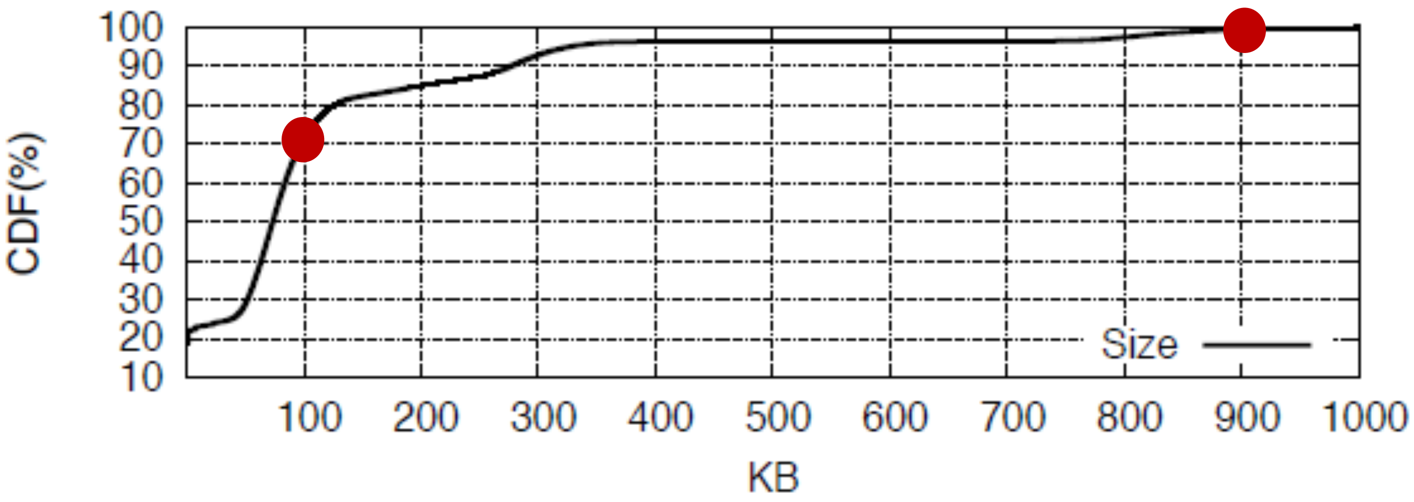
[2] E.Schurman,J.Brutlag.(June,2009).The User and Business Impact of Server Delays, Additional Bytes and Http Chunking in Web Search.

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

# Motivation

- **Currently, data transmission of most web services (*e.g.*, Web search and social websites) are based on TCP.**

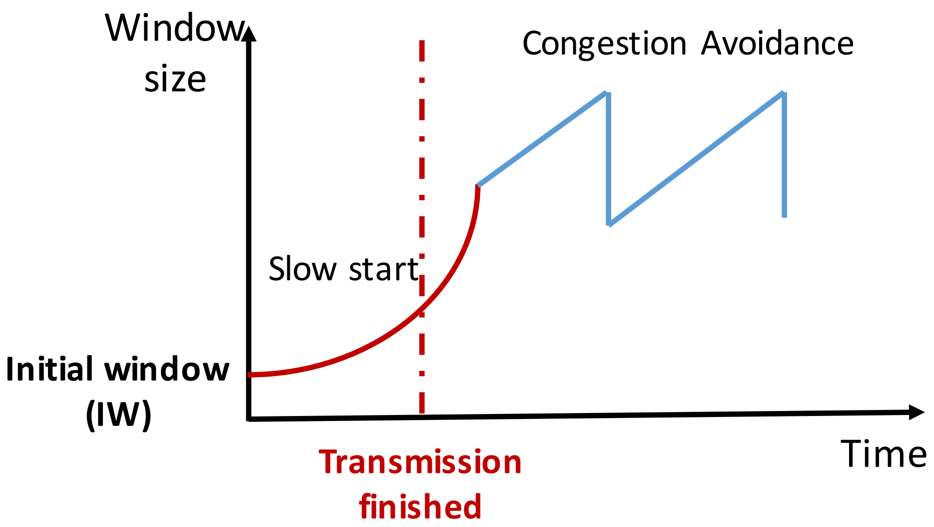
- **Most flows of web service are short.**
  - 99% flows are smaller than 100KB [Greenberg SIGCOMM 09]
  - 70% flows of  Baidu mobile search service are smaller than 100KB.



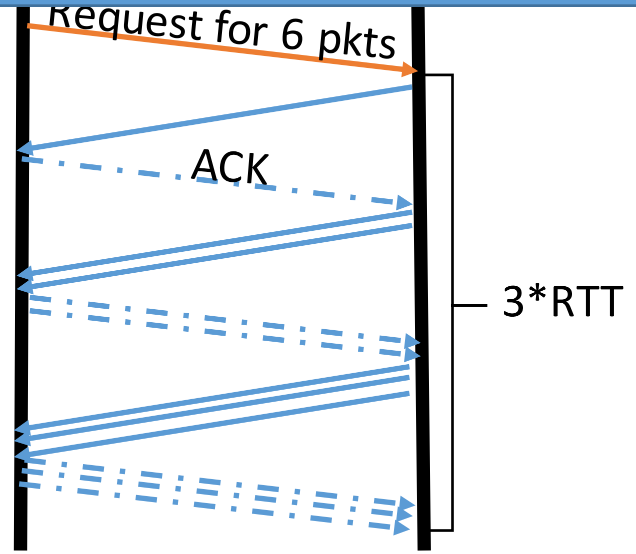
# Motivation

- **Short flows are slow because of TCP's *flow startup problem* [RFC6077]**
  - Slow-start mechanism with conservative IW to probe the bandwidth during the transmission.

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




**Inefficient bandwidth utilization**




**Multiple RTTs for short flow**

# 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)**
     - Pros: Quickly converge to the right available bandwidth after transmission begins.
     - Cons: Slow startup problem exists.

  2. **Fast loss recovery (e.g. Reactive, Proactive** [SIGCOMM13]**, SRTO**[CONEXT15],

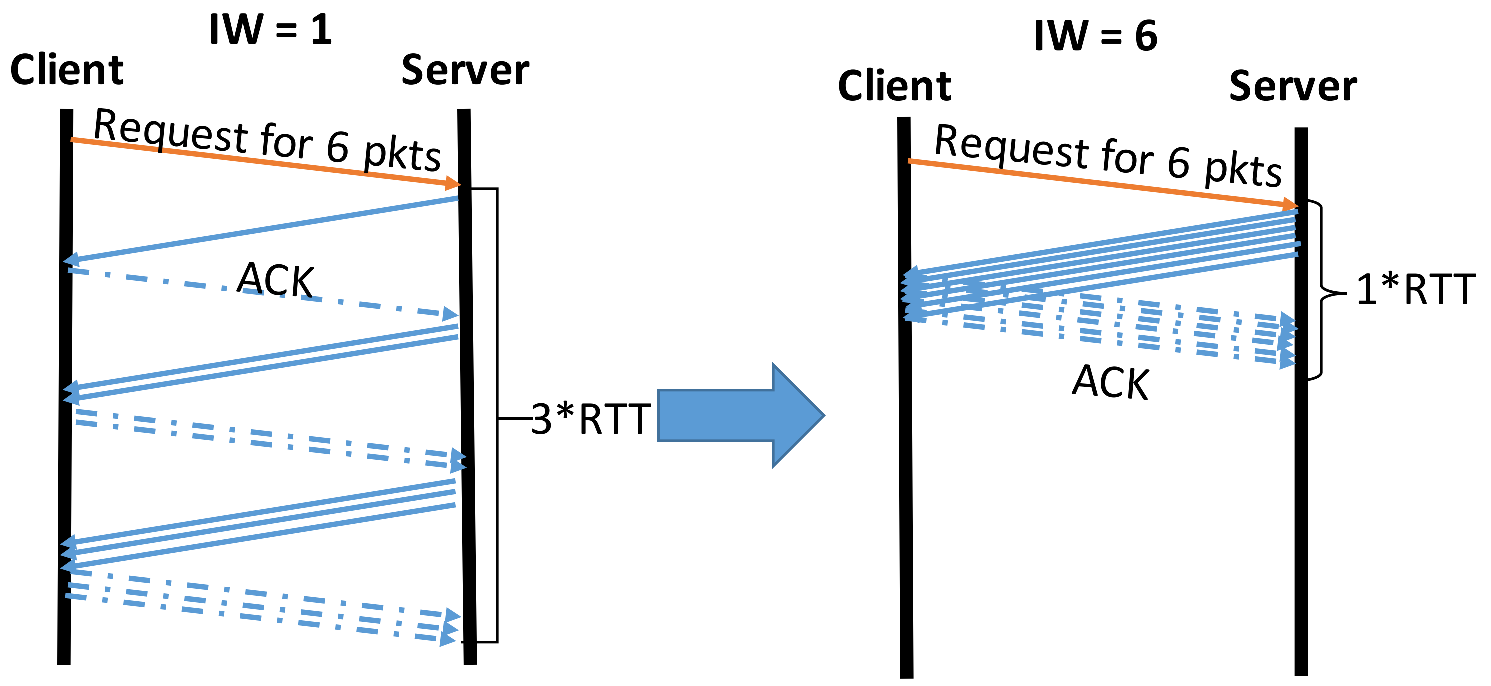
**The flow startup problem is only mitigated but not directly solved**

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

  4. **Increasing IW (IW = 2~4 in 2002**[RFC3390]**, IW = 10 in 2013**[RFC6928])
     - Pros: simple and easily deployed.
     - Cons: one standard value is suboptimal.

# Our goal

- **Solve the flow startup problem by only setting the appropriate Initial congestion window (IW).**
  - Fast bandwidth convergence, *Easy deployment at server side*




**Toy example: client request for 6 packets data, the link limitation IW > 6.**

# Challenges of setting IW

## 1. How to choose IW?

- **Large IW -> network congestion; Small IW -> long latency, which one is best?**
- **No current knowledge to predict the best IW at the flow startup phase.**
  - The TCP sender has very little information on the current network condition.
- **No historical knowledge to learn.**
  - Only one kind of IW has been used.

## 2. Different users' network conditions are different. One IW is not enough.

| 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*

# TCP WISE design

- **TCP WISE key ideas:**
  1. Using **different IWs** for **different user clusters**.
  2. For one user cluster, **wisely exploring** the best IW by continuously performing **A/B testing**.

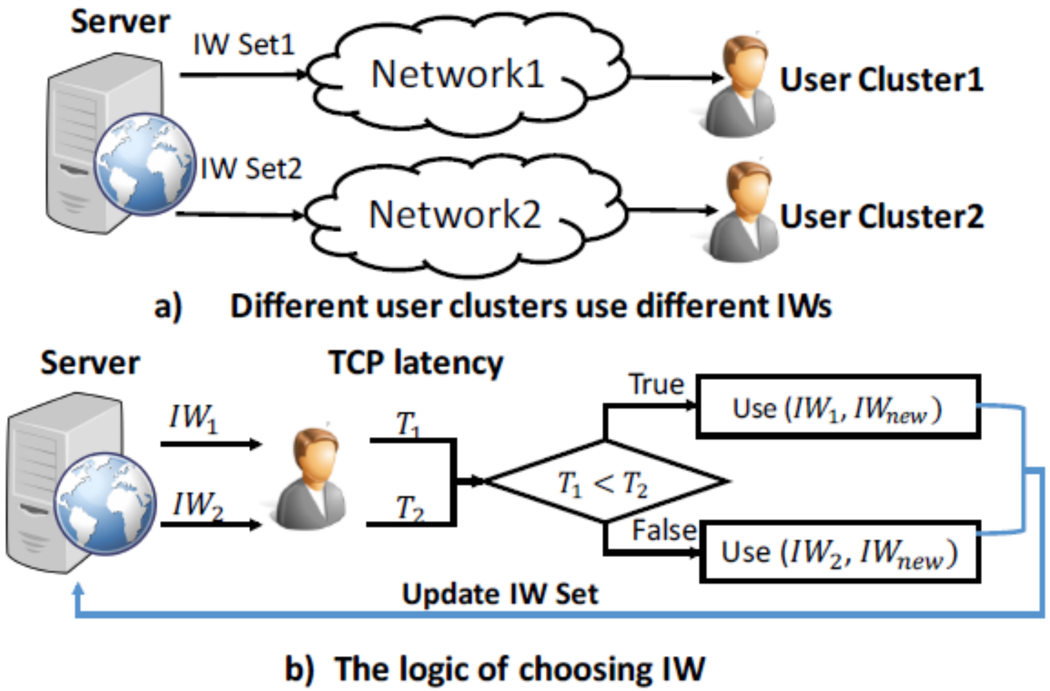
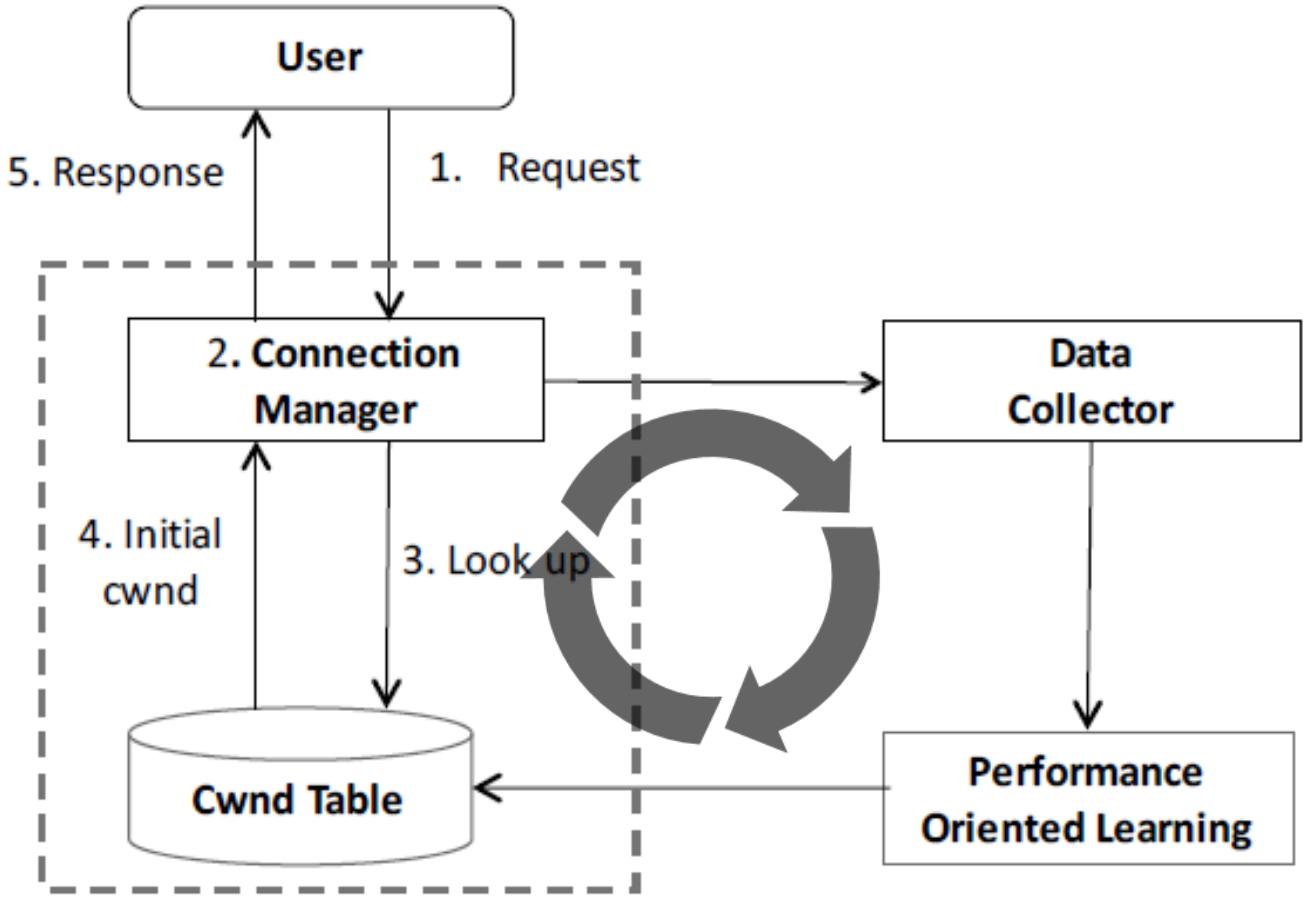


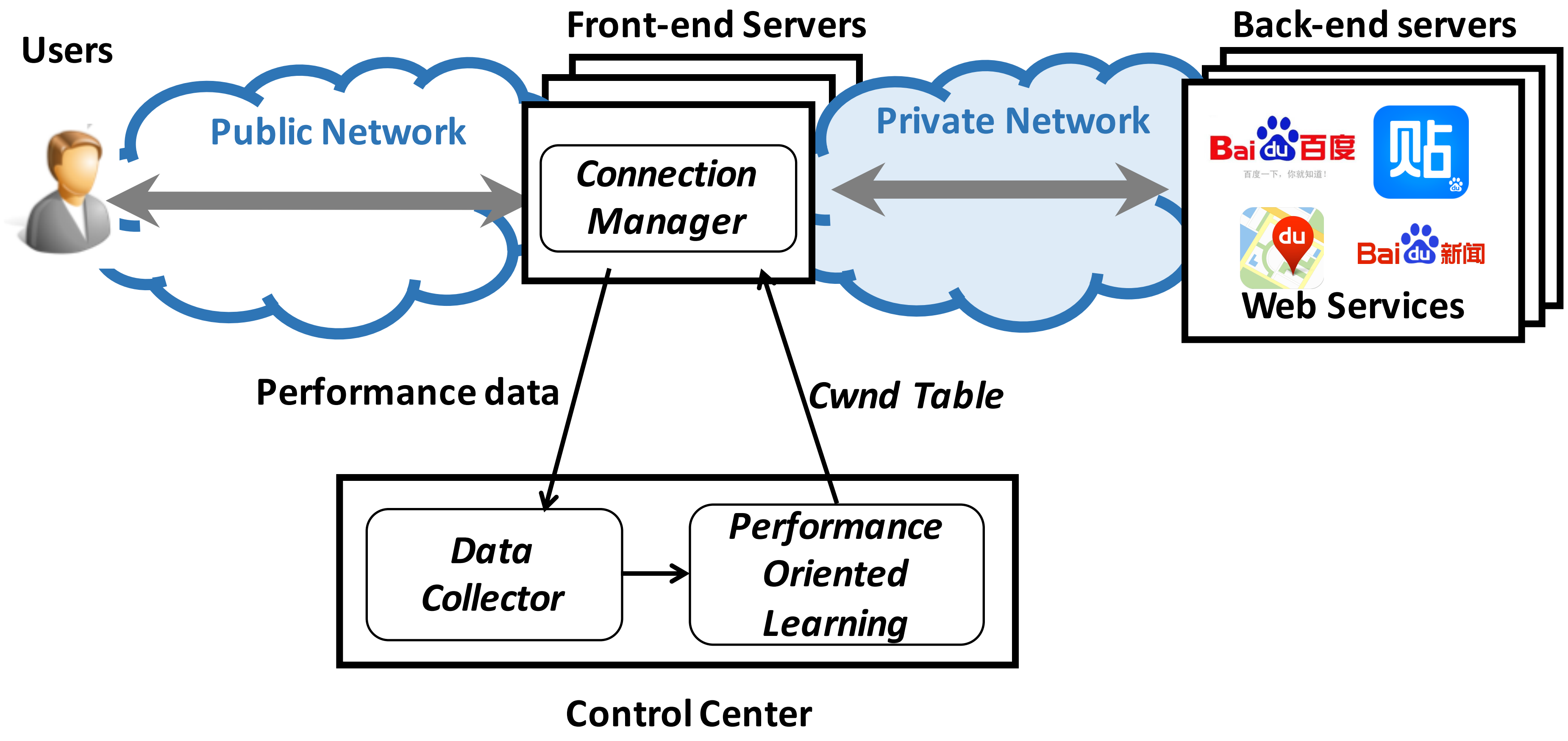

Fig. 3. The key idea of TCP WISE

# System Overview



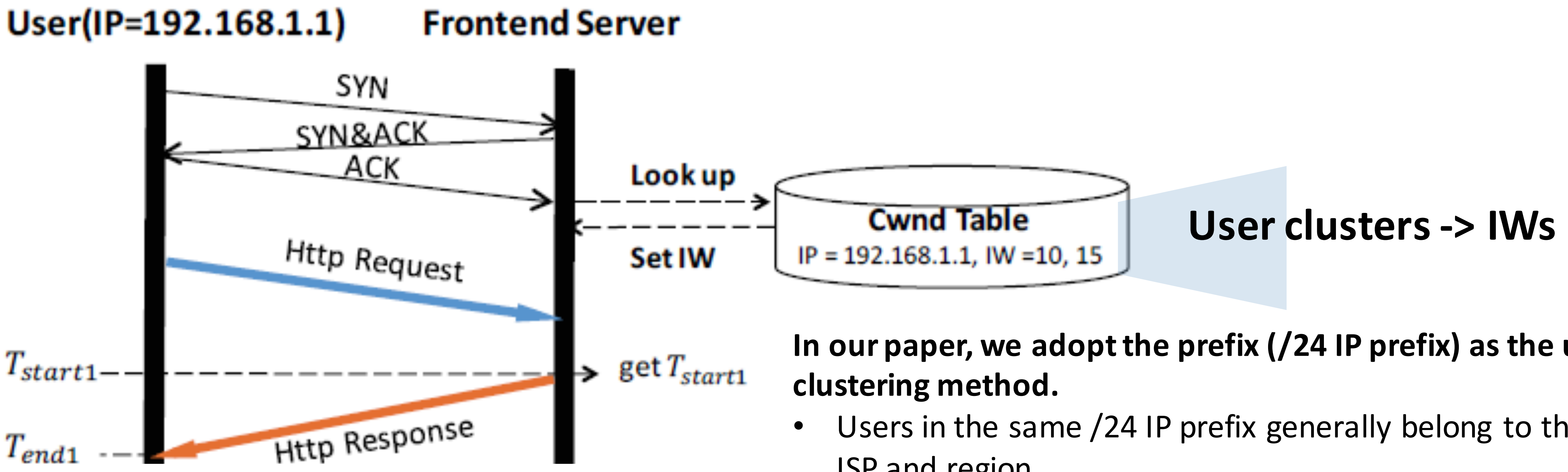

A close-loop learning scheme

# System Overview



# System detail

- **Connection Manager:**
  - Use different IWs for different user clusters.




**In our paper, we adopt the prefix (/24 IP prefix) as the user clustering method.**

- Users in the same /24 IP prefix generally belong to the same ISP and region.
- Users from same /24 IP prefix will have similar network performance [Hongqiang NSDI 16]

# System detail

- **Data collector:**
  - Collect data from frontend servers.

| ID | Metrics |
|---|---|
| 1 | Timestamp |
| 2 | Client IP |
| 3 | Initial Cwnd |
| 4 | Client Rwnd |
| 5 | MSS |
| 6 | Size |
| 7 | TCP Latency |
| 8 | RTT (no accurate ) |
| 9 | Retransmission rate |
| 10 | Timeout |

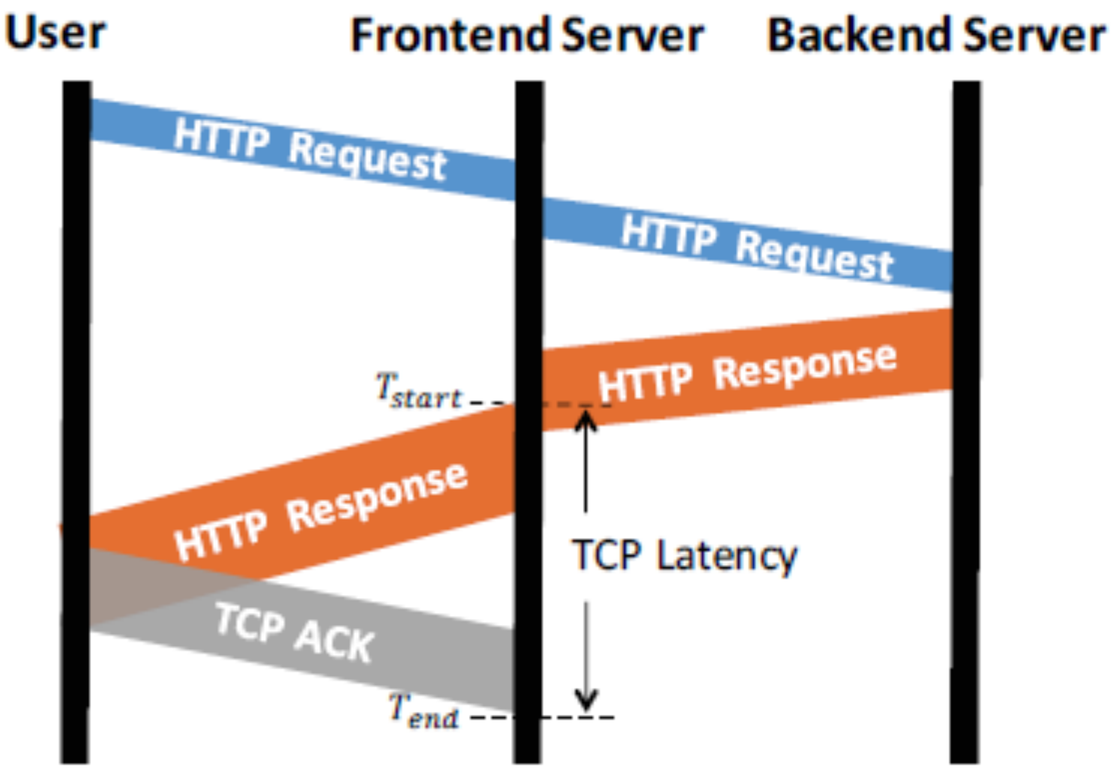
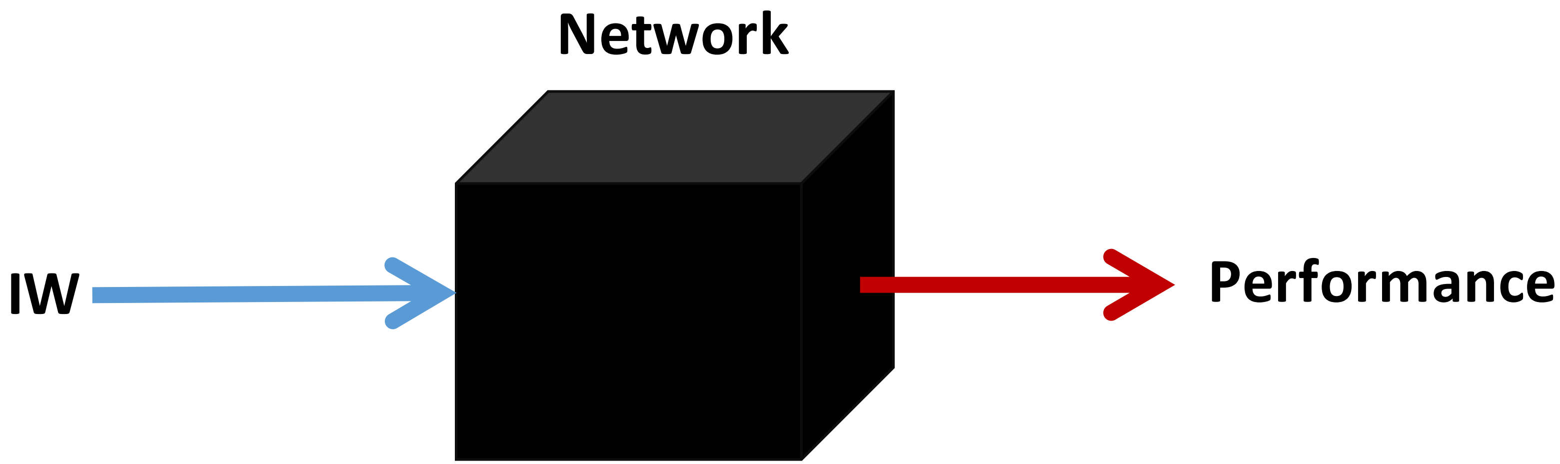



Fig. 1. The detail timeline of the HTTP request/response.

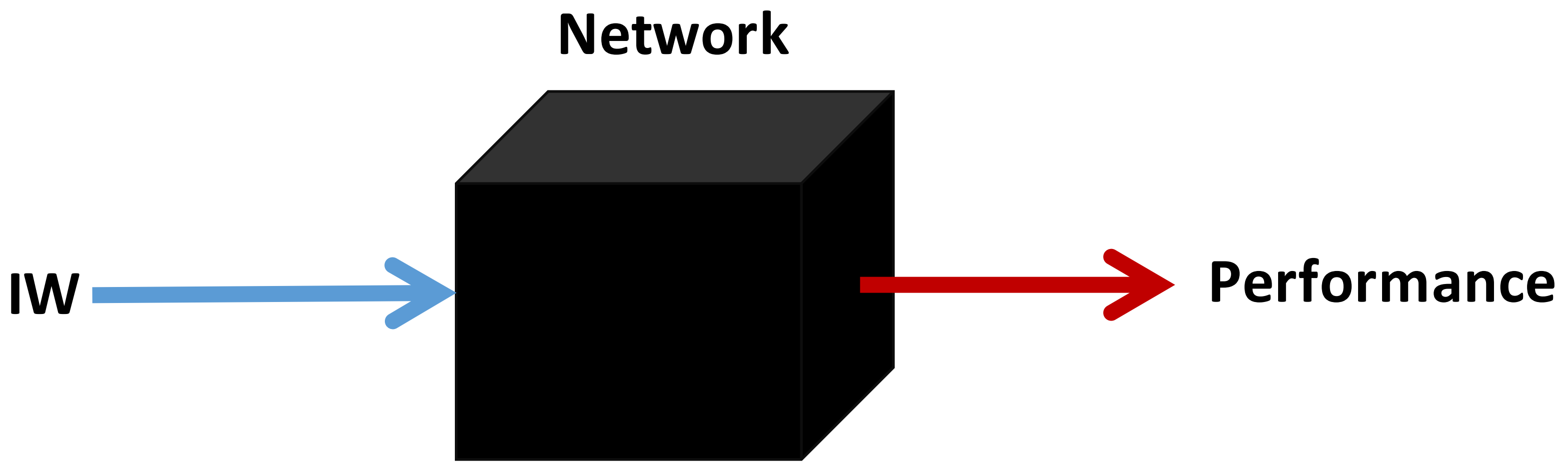
# System detail

- **Performance Oriented Learning**
  - Learning the best IW



# System detail

- **Performance Oriented Learning**
  - Learning the best IW

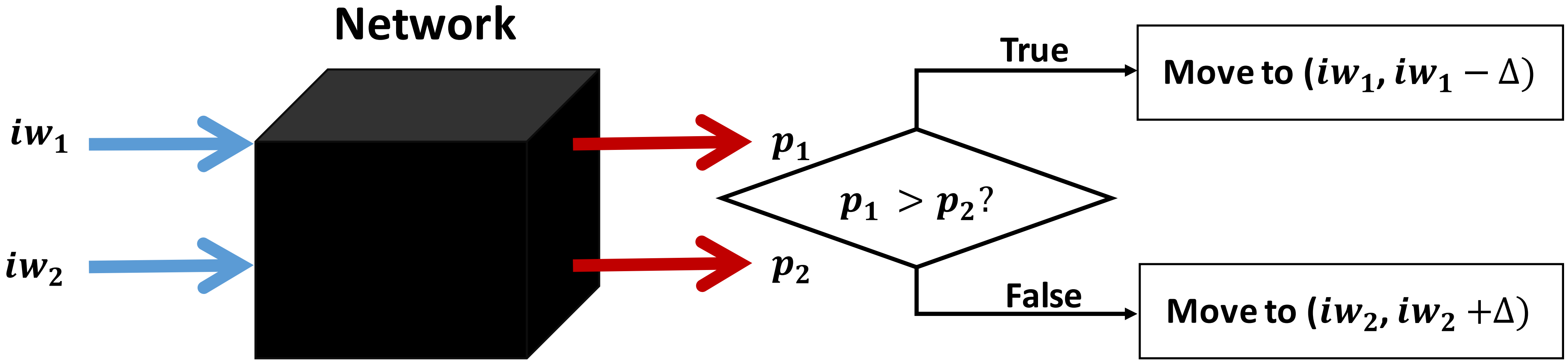



**Performance objective:**

**e.g. average, 80th, 90th TCP latency, average loss rate.**

# System detail

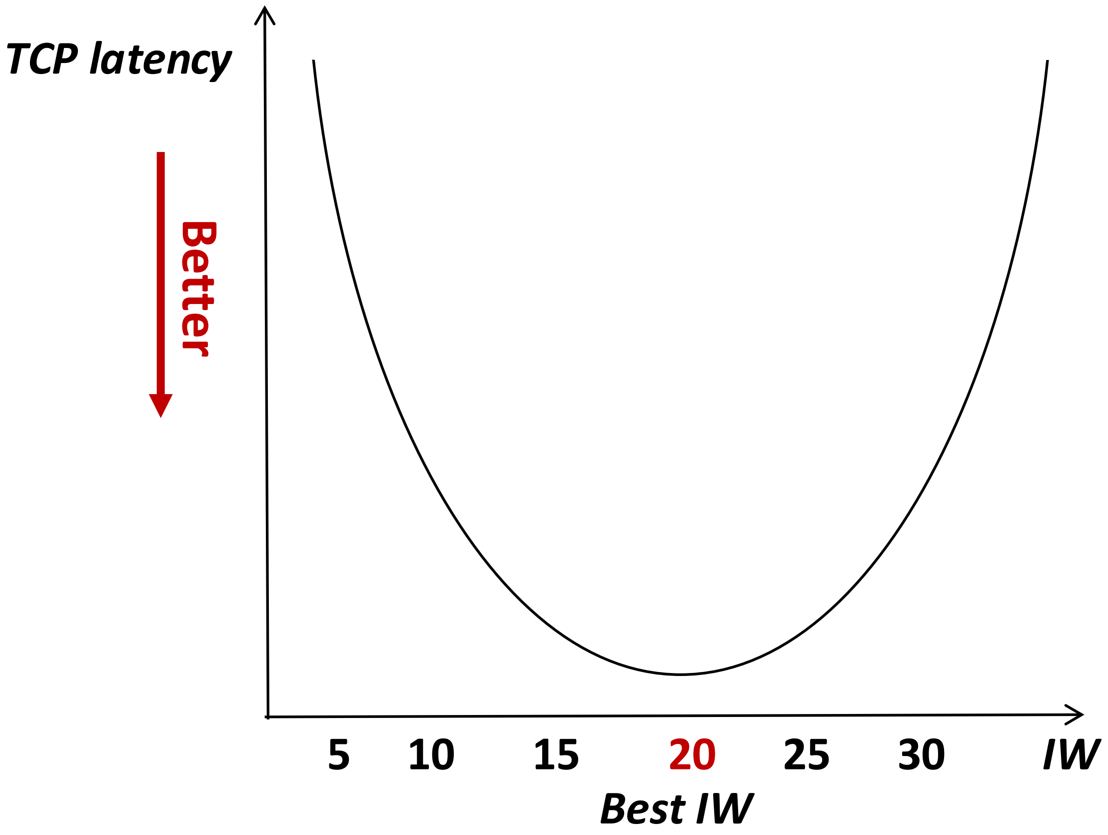
- **Performance Oriented Learning**
  - Learning the best IW




**$\Delta$ is a constant value, $iw_2 = iw_1 + \Delta$**

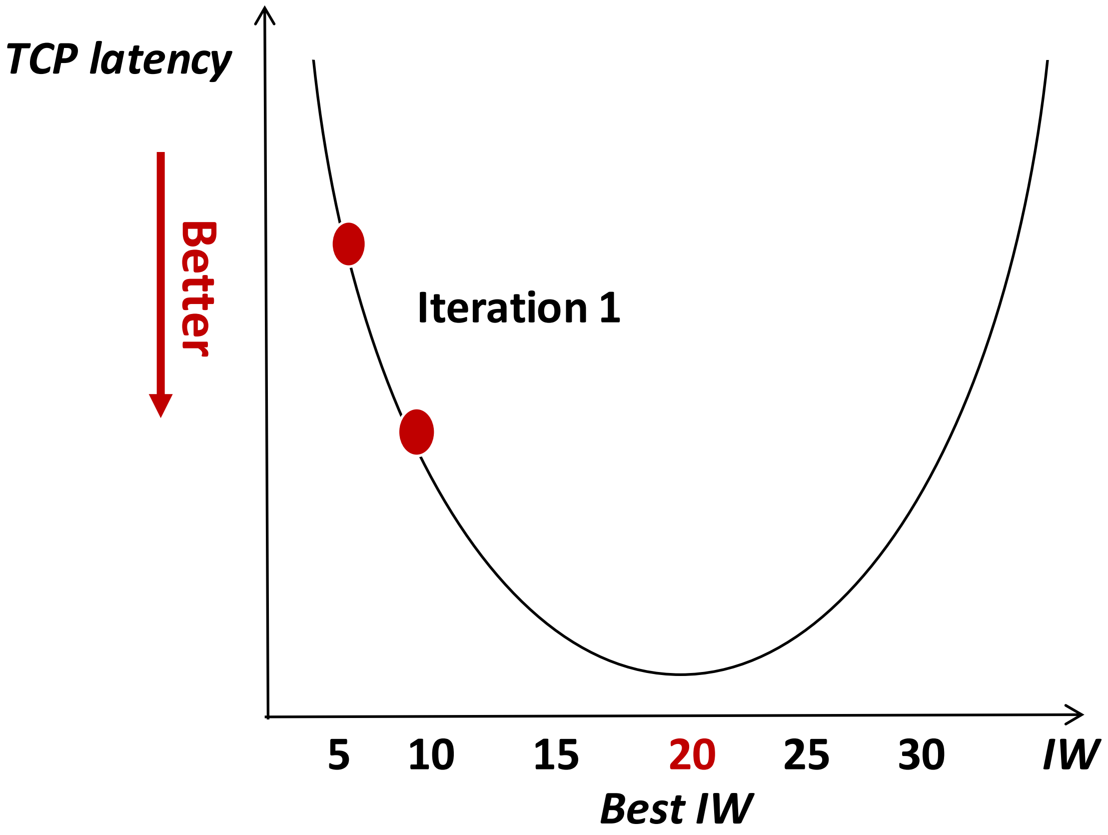
# System detail

- **Performance Oriented Learning**
  - Learning the best IW



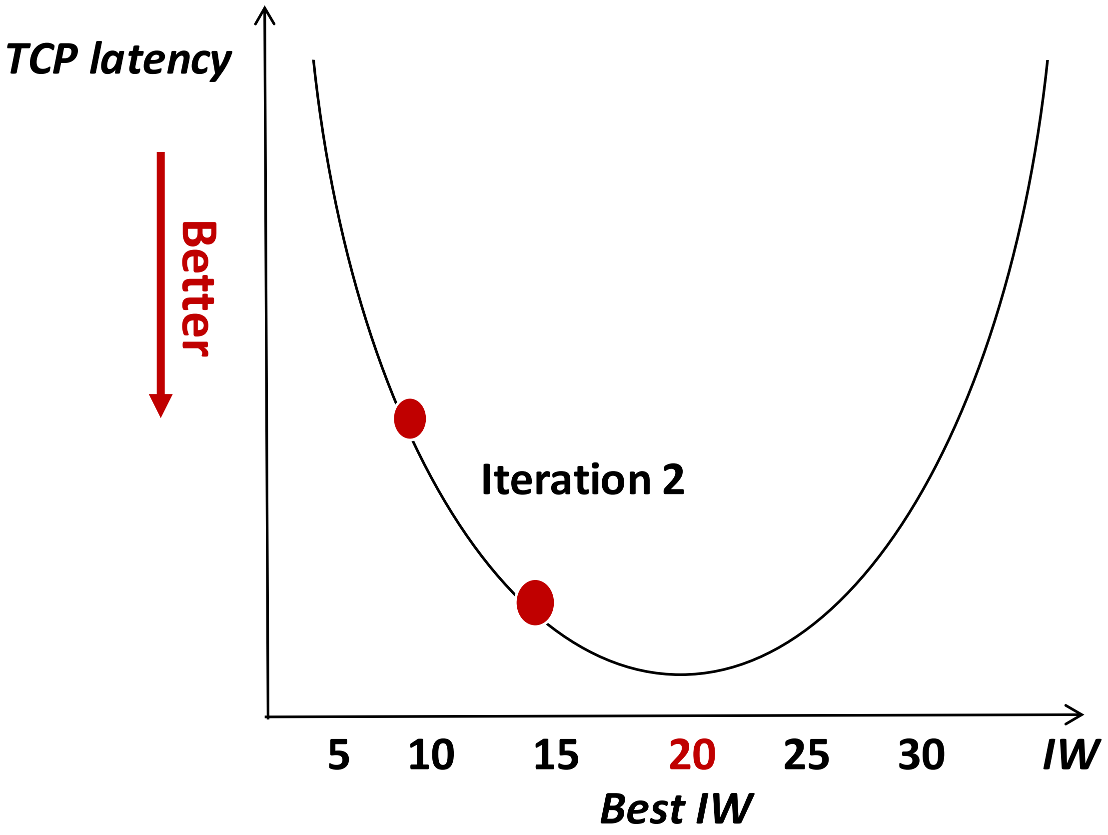
# System detail

- **Performance Oriented Learning**
  - Learning the best IW



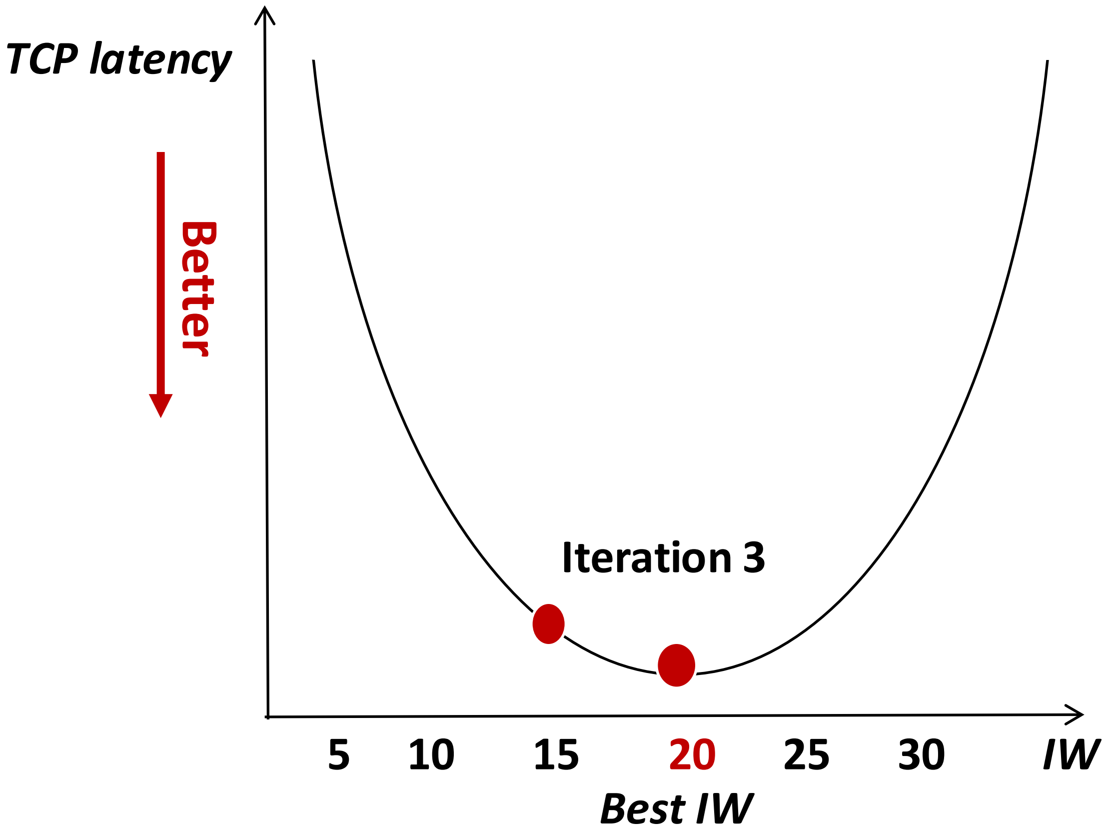
# System detail

- **Performance Oriented Learning**
  - Learning the best IW



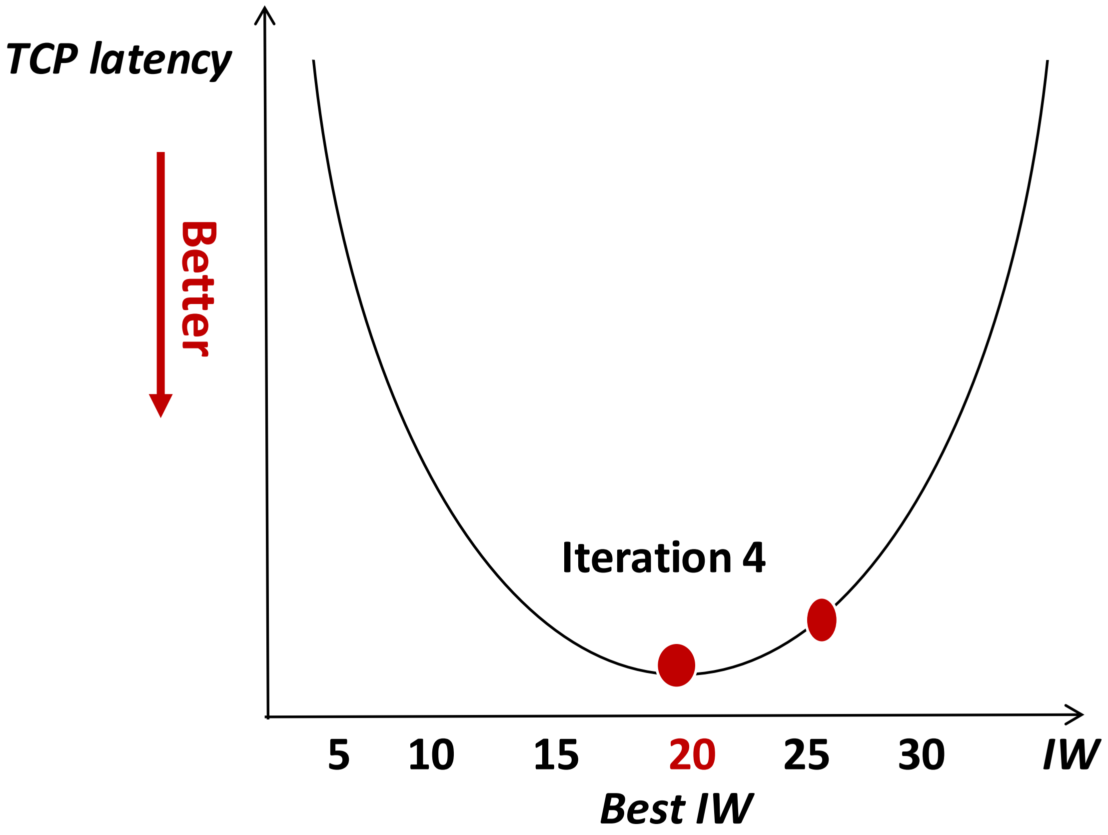
# System detail

- **Performance Oriented Learning**
  - Learning the best IW



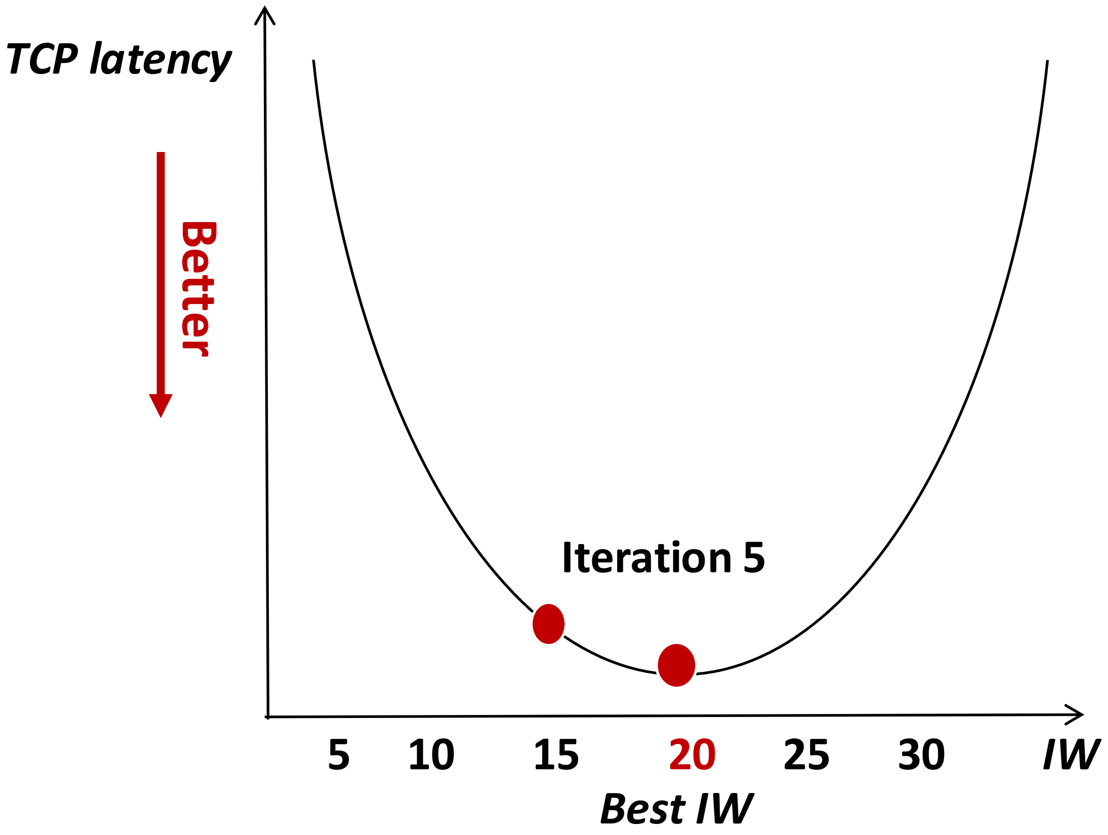
# System detail

- **Performance Oriented Learning**
  - Learning the best IW



# System detail

- **Performance Oriented Learning**
  - Learning the best IW



# Evaluation

- **1. Testbed Experiment**
  - converge to best IW over time
  - handle the network changes.

- **2. Online experiment**
  - reduce the 80th percentile latency of mobile search service by about 10% with little negative impact on loss.

# Evaluation

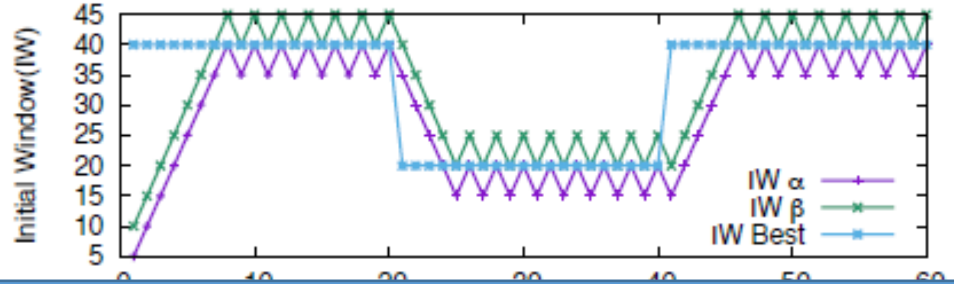
- **1. Testbed experiment**
  - **Testbed setup:**


- **Control the size of HTTP response**
  - 100KB
  - 100 requests in every minutes
  - Learning iteration = 1min

- **Control network condition**
  - Bandwidth, RTT , loss
- **Run TCP WISE**

# Evaluation

- **1. Testbed experiment**




**TCP WISE can converge and handle the network changes**

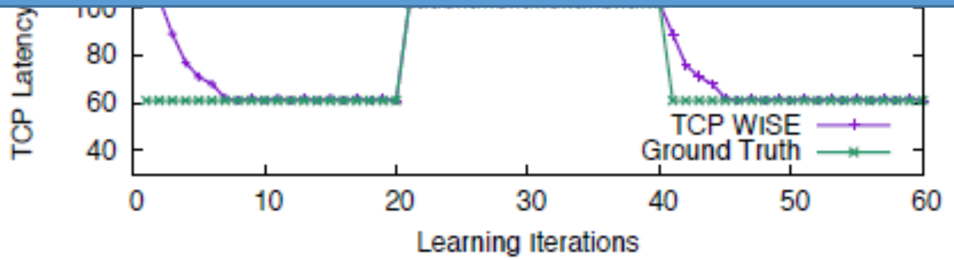
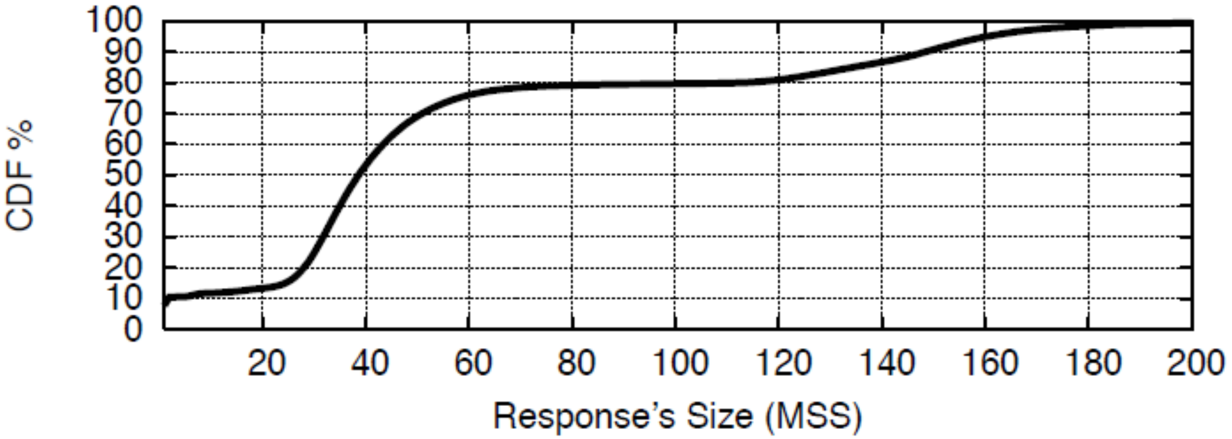
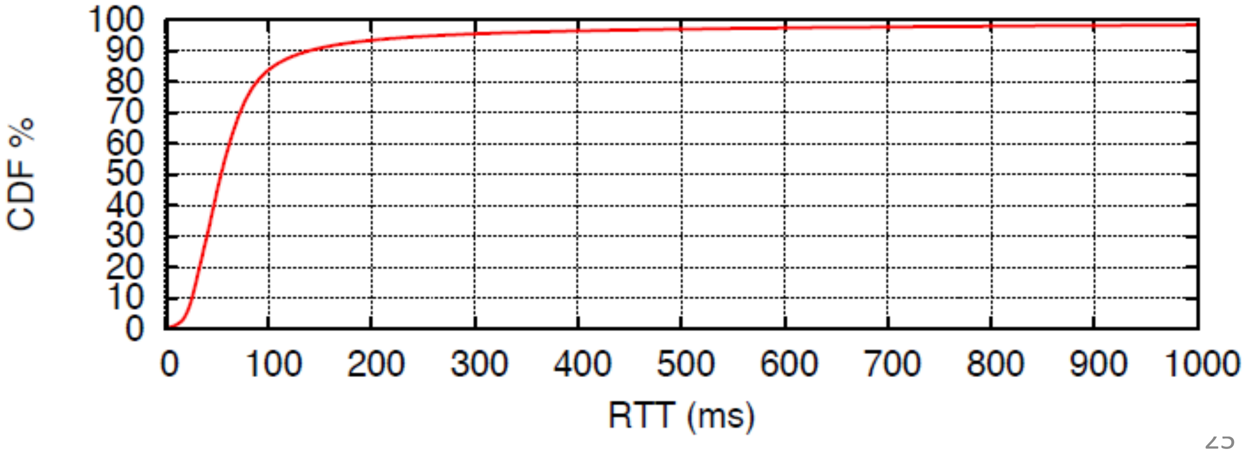



Fig. 7. Bandwidth changes. During 1∼20 and 41∼60 learning iterations, the network condition is (bandwidth = 20Mbps, RTT=20ms, loss = 0). During 21∼40 the network condition changes to (bandwidth = 10Mbps, RTT=20ms, loss = 0).

# Evaluation

- ***2. Online experiment***
  - **Experiment setup:**
    - **Web service: Baidu mobile search**
    - A/B testing: TCP-10 vs TCP WISE
    - Initial IW set = (10, 15, 20, 25 ,30)
    - Δ = 5

# Evaluation

- ***2. Online experiment***
  - ***TCP latency result***

Latency reduction: 30ms~70ms

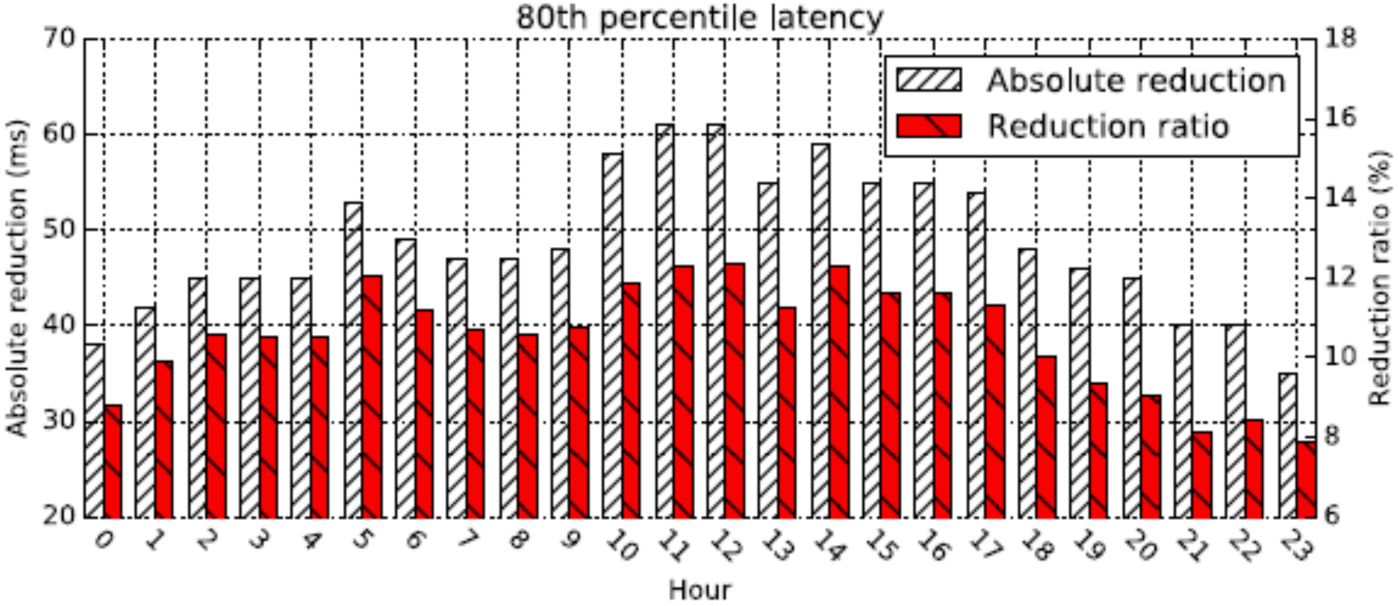



Fig. 12. The $80^{th}$ percentile latency of TCP WISE compared with TCP 10. The x-axis presents the hour, and the left y-axis presents the absolute reduction of latency and the right y-axis presents the reduction ratio of latency.

Reduction ratio: About 10%

# Evaluation

- ***2. Online experiment***
  - **IW distribution**
    - About 4000 user clusters
    - Different user clusters use different IWs. 30 is the popular IW.

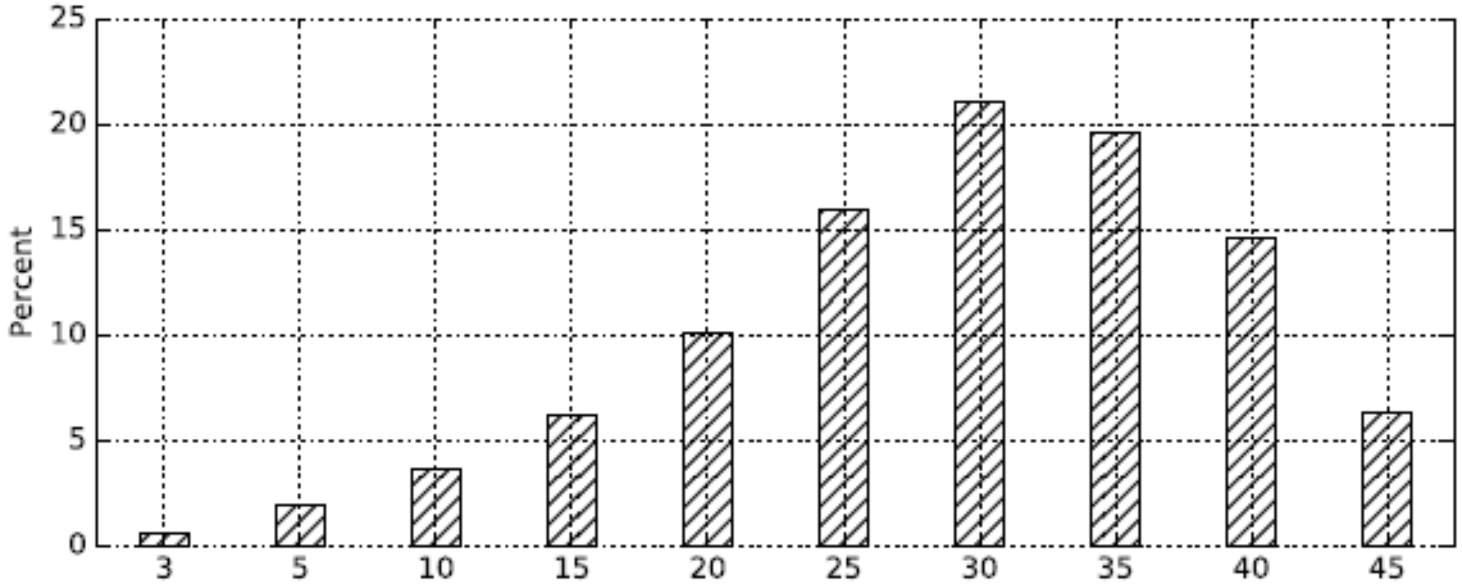



Fig. 11. The distribution of each cluster's IW. X-axis is the IW and y-axis presents the percentage of its user clusters.

# Evaluation

- ***2. Online experiment***
  - **Negative impact**
    - retransmission rate = #retrains packet/# trans packet
    - Timeout ratio = #responses whose transmission occurred timeout/#responses

| Metrics | Retransmission Rate (%) | Timeout Ratio (%) |
|---|---|---|
| TCP WISE | 2.53 | 5.3 |
| TCP-10 | 1.93 | 5.0 |
| Diff | 0.6 | 0.3 |

**Little negative impact**

# Summary

- **Slow startup problem**
  - One initial congestion window is not enough
  - Best IW is unknown

- **We proposed TCP WISE.**
  - Exploring the appropriate IW with A/B testing
  - Using different IWs for different user clusters.

- **Testbed and Online experiment prove TCP WISE works well.**
  - Algorithm can converge and can handle network changes.
  - Reduce the 80*th* latency of the HTTP responses by about 10% online.

# Thanks

Q&A?

# Evaluation

- **1. Testbed experiment**
  - Algorithm convergence and network changes

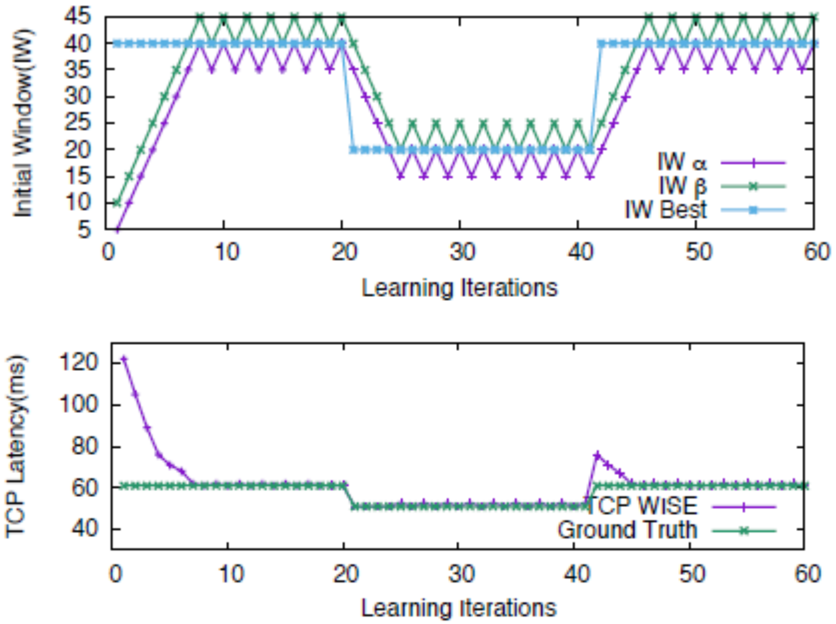



Fig. 8. RTT changes. During 1∼20 and 41∼60 learning iterations, the network condition is (bandwidth = 20Mbps, RTT=20ms, loss = 0). During 21∼40 the network condition changes to (bandwidth = 20Mbps, RTT=10ms, loss = 0).

# Evaluation

- **1. Testbed experiment**
  - Algorithm convergence and network changes

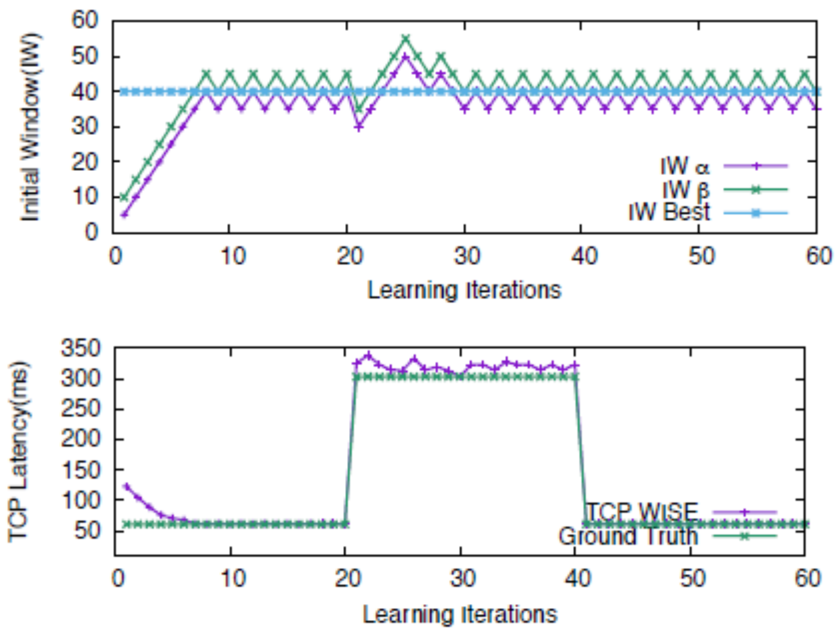



Fig. 9. Loss rate changes. During 1∼20 and 41∼60 learning iterations, the network condition is (bandwidth = 20Mbps, RTT=20ms, loss = 0). During 21∼40 the network condition changes to (bandwidth = 20Mbps, RTT=20ms, loss = 10%)

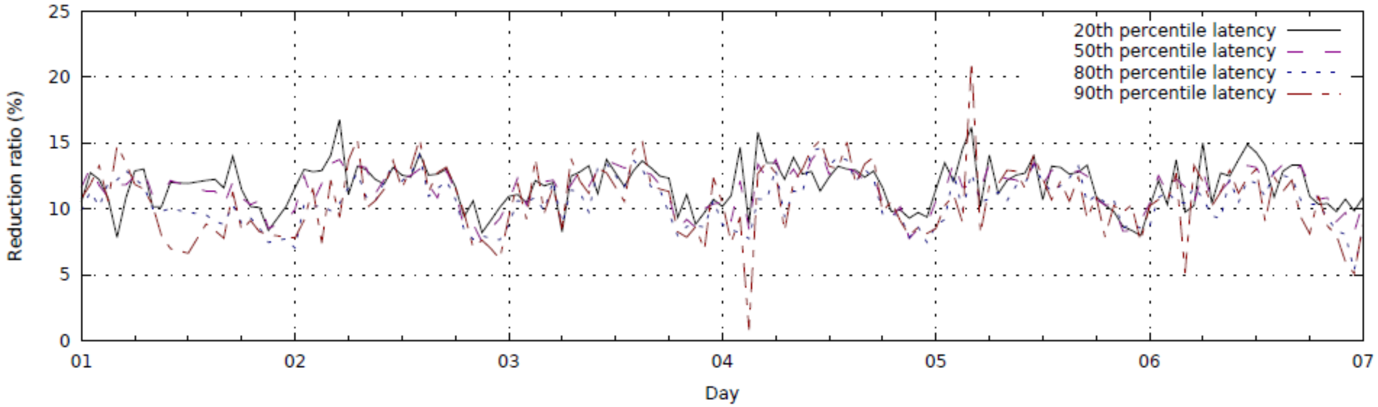


Fig. 13. TCP latency reduction ratio compared with TCP-10 in *Mobile Search* service.

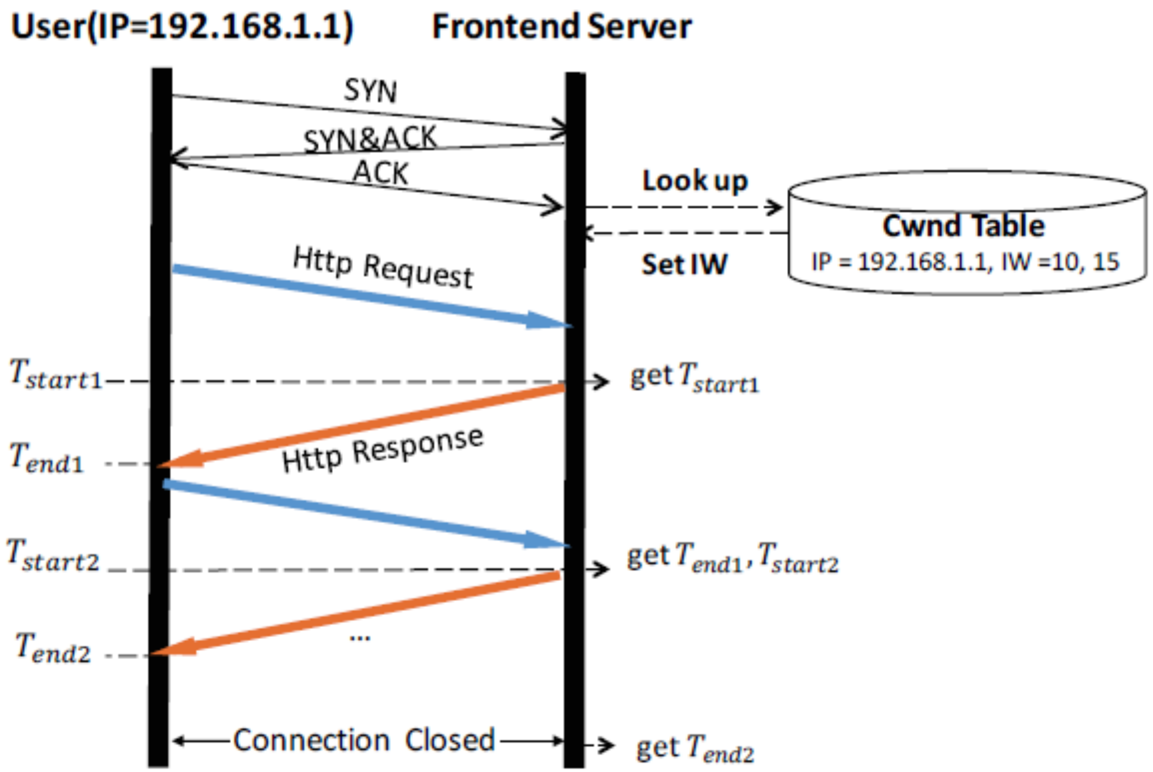
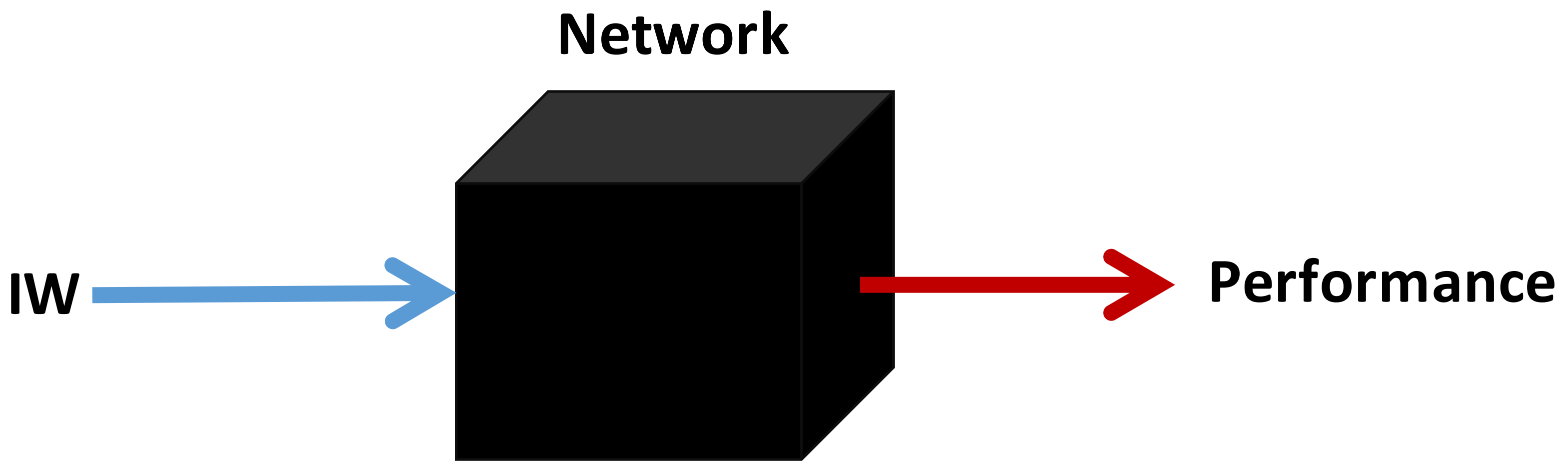



Fig. 5. A simple example of TCP WISE's online workflow, including setting IW and collecting data procedure.

# System detail

- **Performance Oriented Learning**
  - What is the best IW?




**Performance objective:**

**e.g. average, 80th, 90th TCP latency, average loss rate.**