Swarm-based Incast Congestion Control in a Datacenter Serving Web Applications

Haoyu Wang*, *Haiying Shen* and Guoxin Liu^*

*University of Virginia, ^Clemson University

Outline

- Introduction
- Approach description
- Evaluation
- Conclusion

Outline

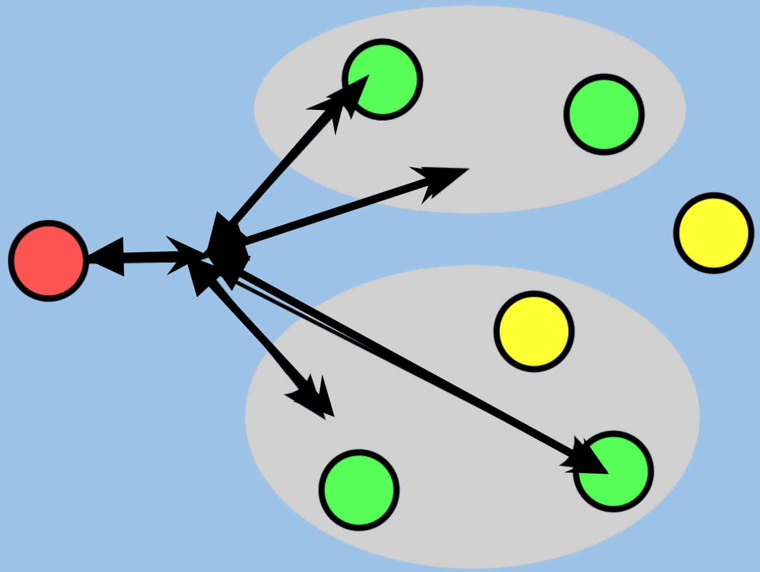
- Introduction
- Approach description
- Evaluation
- Conclusion

Incast congestion is a common problem in modern datacenters

1. TCP timeout and retransmission
2. Throughput loss
3. Increased latency
4. Application failure

Glenn from *Morgan Stanley*, *NSDI 2015*

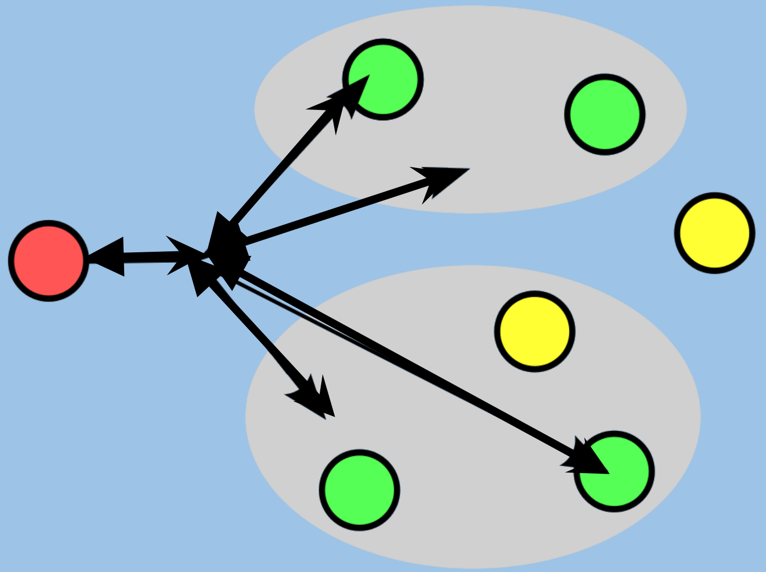
Incast congestion



Incast is a many-to-one communication pattern commonly found in cloud data centers. It begins when a singular parent server places a request for data objects to a large number of servers simultaneously.

The Nodes respond to the singular Parent. The result is a micro burst of many machines simultaneously sending TCP data streams to one machine

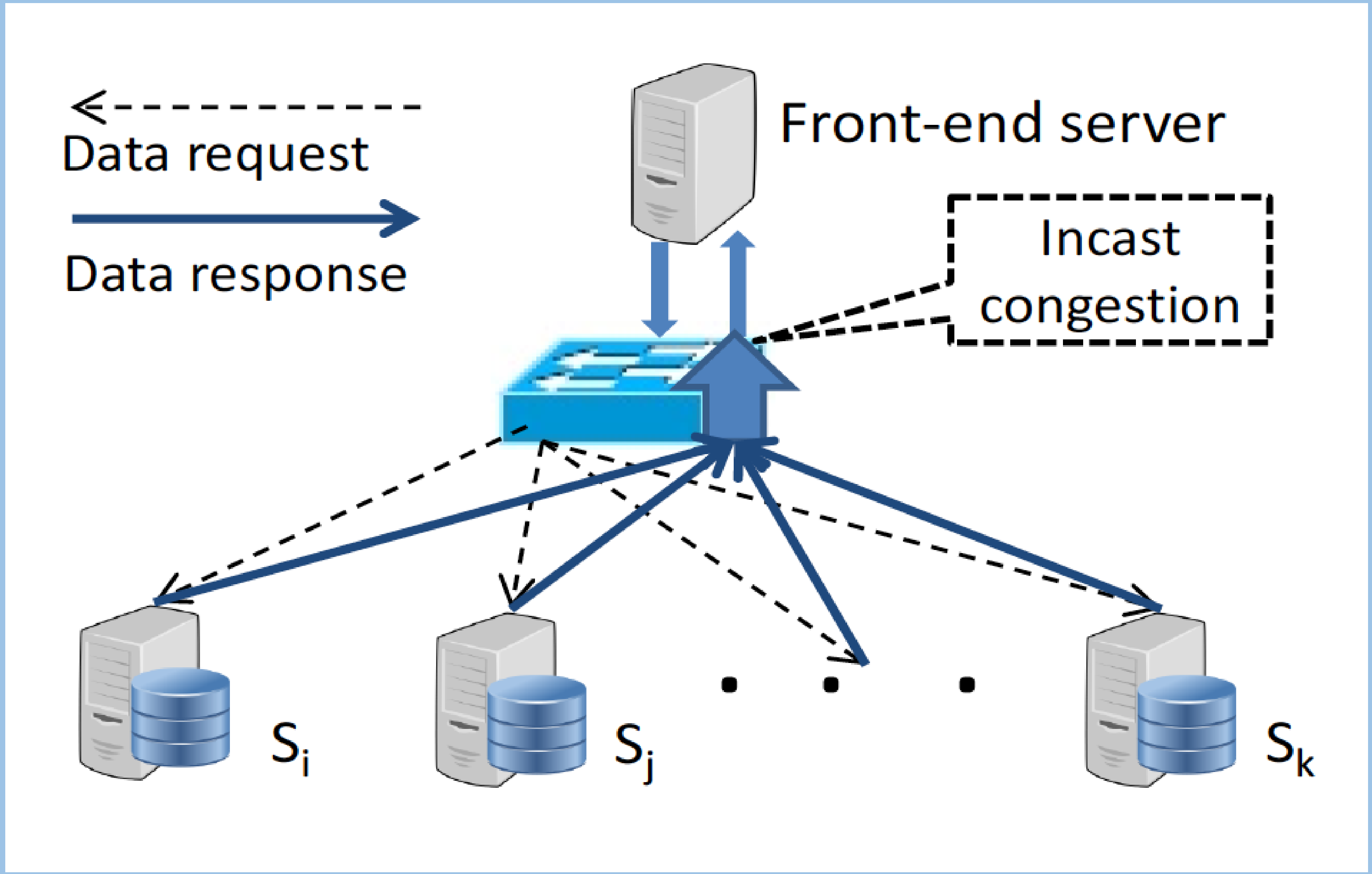
Incast congestion



Incast is a many-to-one communication pattern commonly found in cloud data centers. It begins when a singular parent server places a request for data objects to a large number of servers simultaneously.

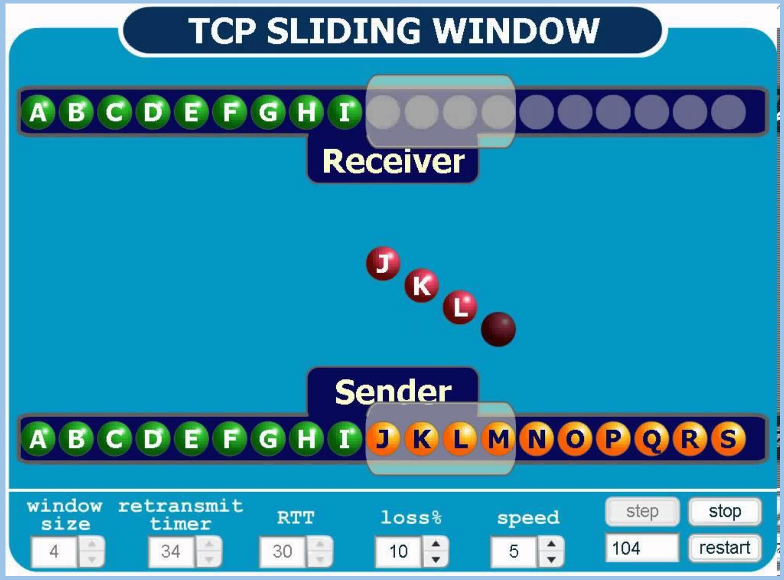
The servers respond to the singular parent, resulting a micro burst of many machines simultaneously sending TCP data streams to one machine

Introduction



Previous work

Sliding Window *MCN'95*

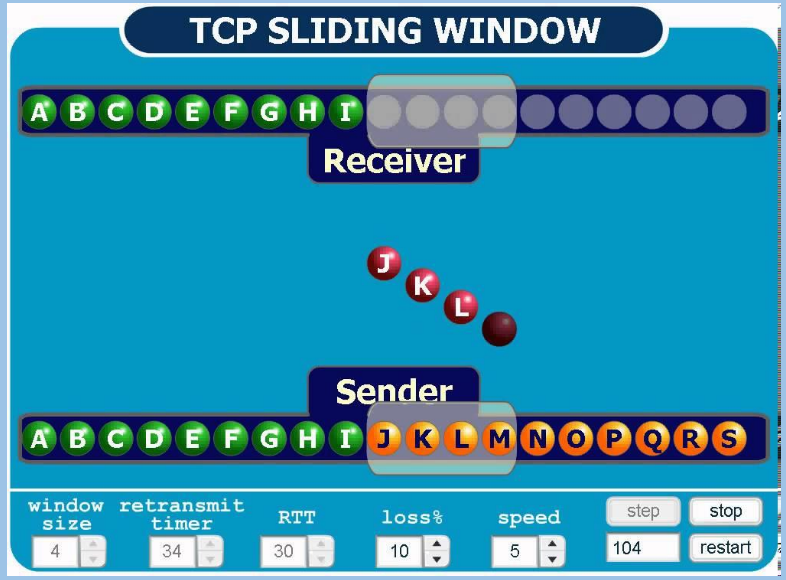


The window size changes after the congestion is detected

ICTCP (Improved sliding window protocol)

Staggered flow

Previous work



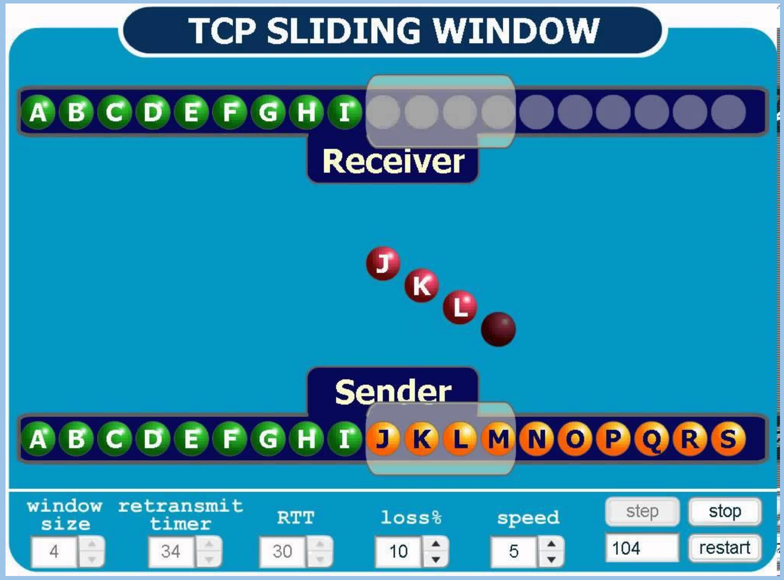
Sliding Window

The window size changes after the congestion is detected

ICTCP (Improved sliding window protocol) *Conext'10*

Staggered flow

Previous work



Sliding Window

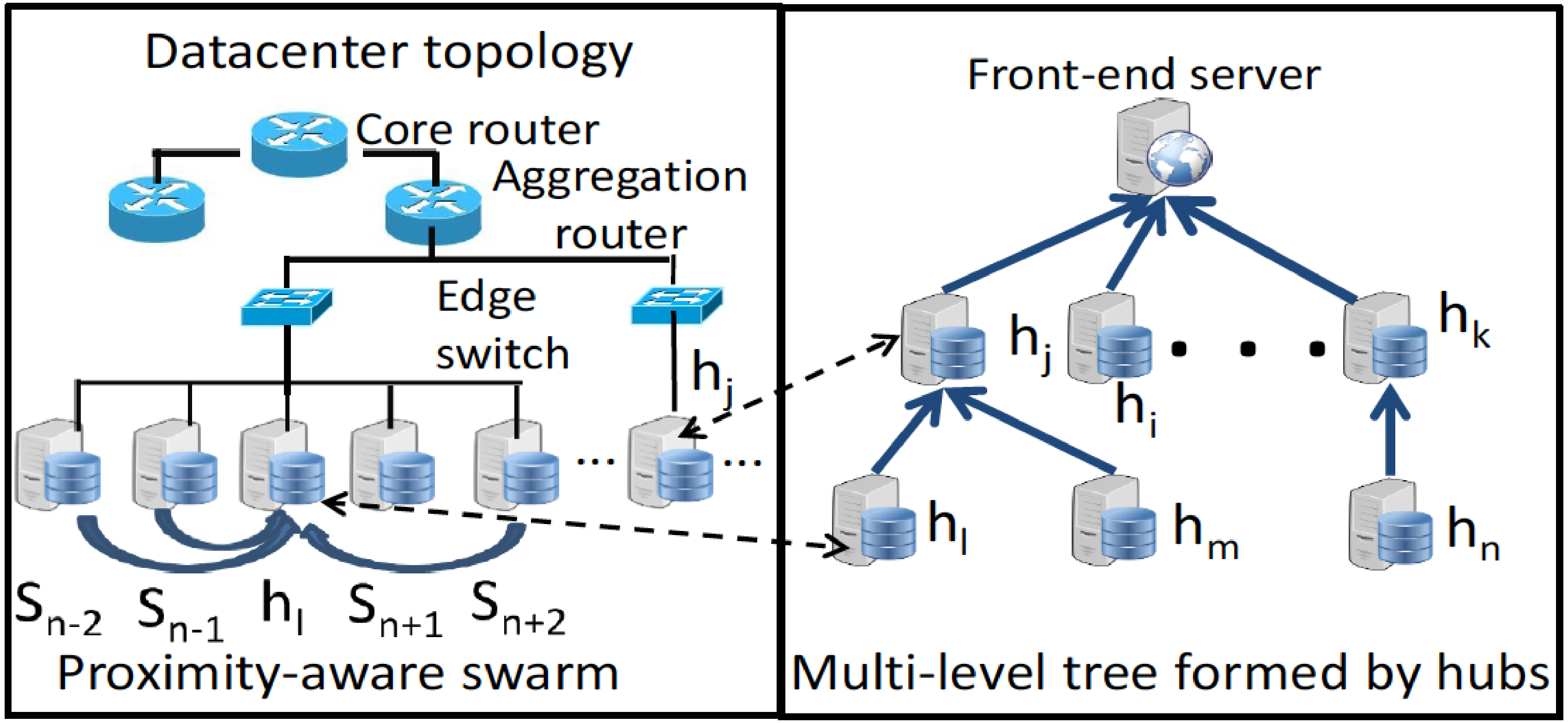
The window size changes after the congestion is detected

ICTCP (Improved sliding window protocol) *Conext'10*

Staggered flow *MASCOTS'12,* COMPSACW'13

Outline

- Introduction
- Approach description
- Evaluation
- Conclusion



A multilevel tree with proximity-aware swarm

Hub: The server connecting with the font-end server and has the largest spare capacity to handle I/O among each rack

Approach Description

A swarm structure is formed only for one data request

1. The transient structure does not need to be maintained
2. Transmitting data through a much smaller structure greatly reduces the latency
3. Data servers without requested data objects do not need to participate in the structure

Determine a suitable number of hubs:

$$N = \frac{\frac{S_e}{B_d} * B_u}{\bar{s} * m}$$

Building multi-level tree of hubs:

1. The hubs under the same aggregation router are linked together in the tree
2. A hub's child always has a smaller number of requested data objects than its parent

Pseudocode of multi-level tree generation

1. Cluster target data servers in each rack into a swarm
2. /* Select a hub from each swarm*/
3. For each swarm do
4. Select the data server with the largest number of requested data objects as the hub; Enqueue the hub into queue Q_h
5. Sort the hubs in Q_h in ascending order of number of requested data objects

UNIVERSITY of VIRGINIA

Pseudocode of multi-level tree generation

1. /*Create multi-level tree from hubs*/
2. While Q_h>N do
3. Dequeue a hub $h_i$from Q_h
4. Select a hub h_j with the smallest number of data objects and under the same aggregation router as h_i; Link h_i as child to h_j
5. While h_j has less than children and h_i has children do
6. Transmit the last child from h_i to be a child of h_j

Two-level data transmission speed control

In order to avoid overloading the front-end server:

1. At the front-end server

The front-end server periodically adjusts the assigned bandwidth to each hub after each short time period

2. At the aggregation router

For multi front-end servers under the same router, we adjust the request transmission speed of each front-end server

Outline

- Introduction
- Approach description
- Evaluation
- Conclusion

Evaluation

Simulation setup:

3000 data servers with fat tree structure

TCP retransmission timeout: 10ms

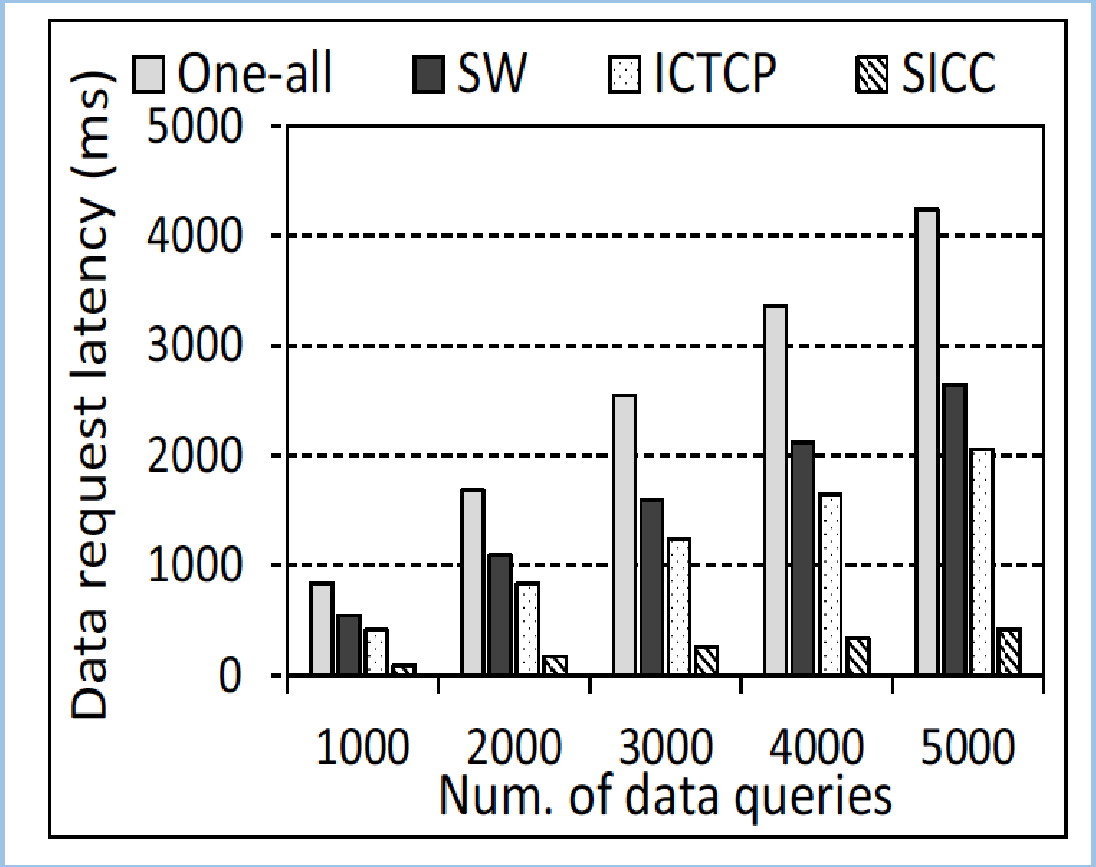
Comparison methods:

1. One-all

2. Sliding window protocol (SW) *MCN'95*

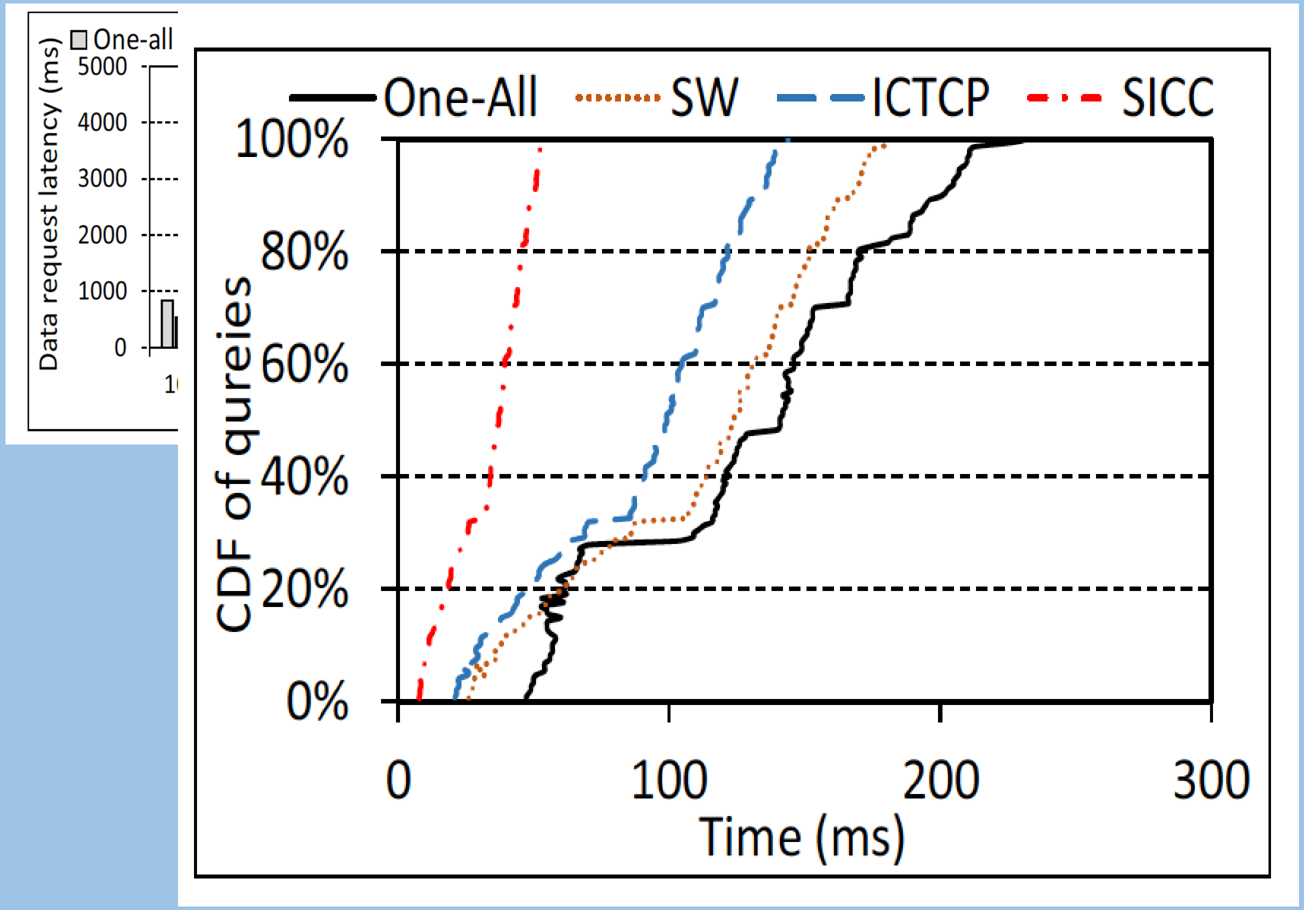
3. ICTCP *Conext'10*

Evaluation



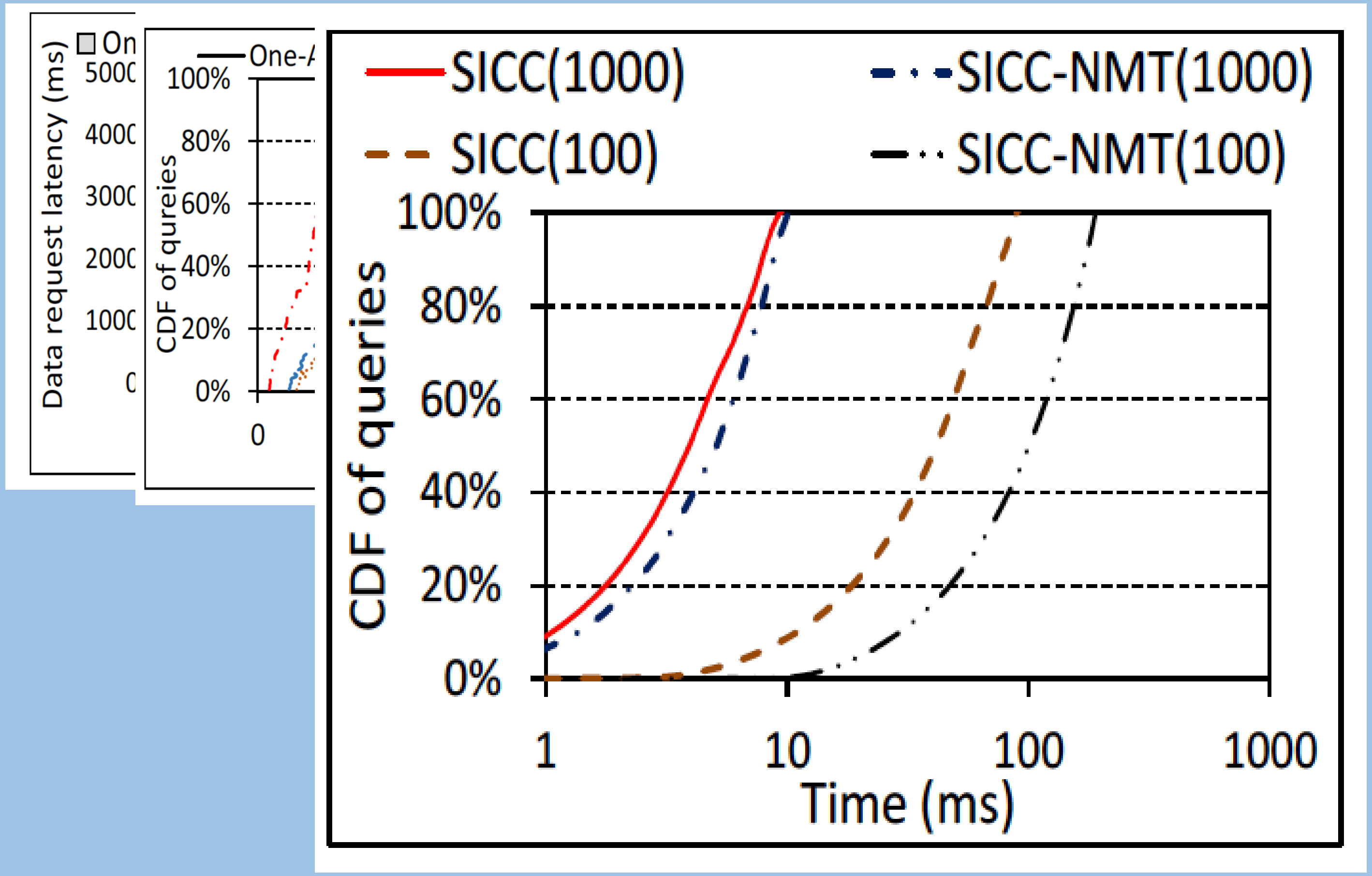
Performance of SICC

Evaluation



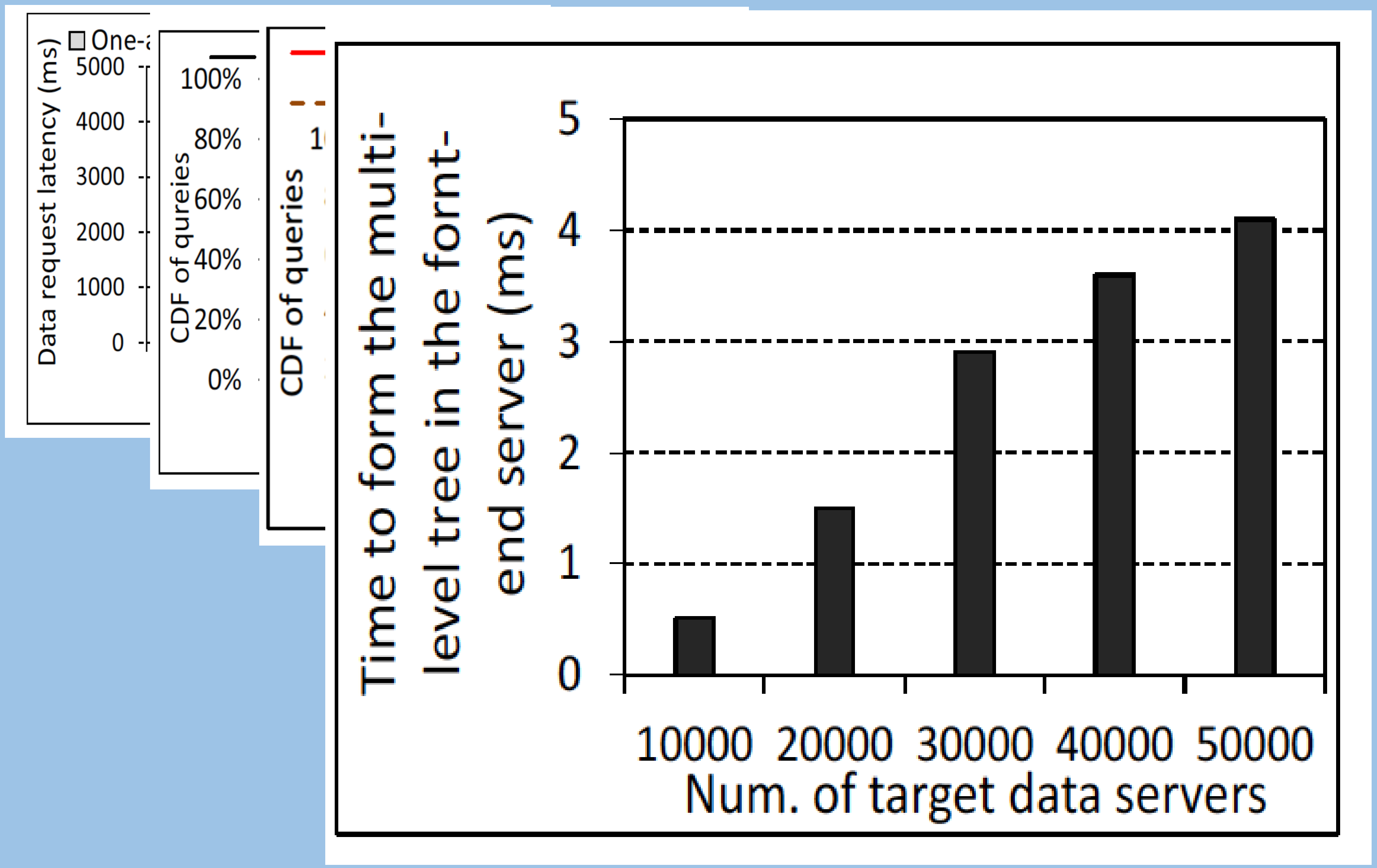
Performance of SICC

Evaluation



Performance of multi-level tree of hubs

Evaluation



Computing time of multi-level tree generation

Outline

- Introduction
- Approach description
- Evaluation
- Conclusion

Conclusion

1. Incast congestion is a common problem in modern datacenters

2. We proposed Swarm-based Incast Congestion Control method (SICC)
    1. Proximity-aware swarm based data transmission
    2. Two-level data transmission speed control
    3. other enhancements

3. Experiments show that SICC achieves higher throughput and lower latency

Thank you!

Question