NETWORK-BASED COMPUTING LABORATORY

Advanced RDMA-based Admission Control for Modern Data-Centers

Ping Lai Sundeep Narravula Karthikeyan Vaidyanathan Dhabaleswar. K. Panda

Computer Science & Engineering Department
Ohio State University

OHIO STATE

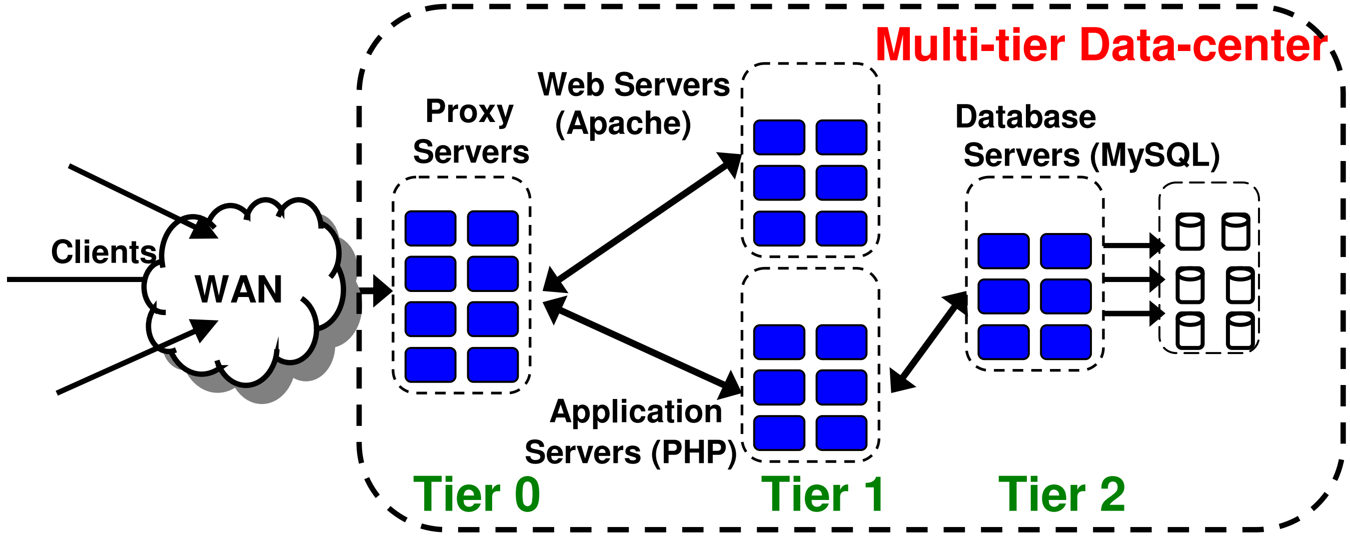
Outline

- Introduction & Motivation
- Proposed Design
- Experimental Results
- Conclusions & Future Work

Introduction

- Internet grows
  - Number of users, various type of services, huge amount of data
  - Typical apps: e-commerce, bio-informatics, online banking etc.
- Web-based multi-tier data-centers
  - Huge bursts of requests →server overloaded
  - Clients pay for service →guaranteed QoS

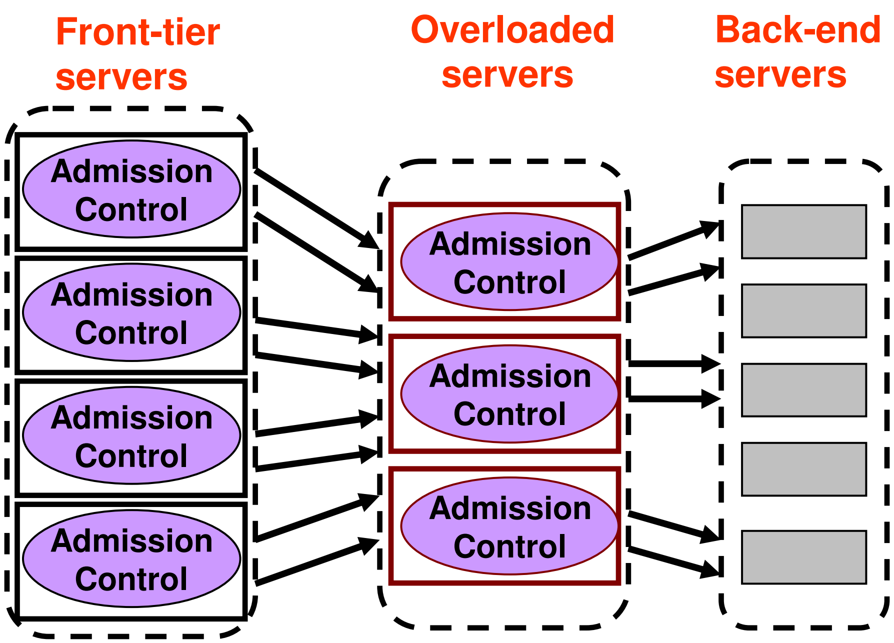
} Efficient admission control needed!



General Admission Control

- What is admission control?
  - determine whether to accept/drop the incoming requests while guaranteeing the performance (or QoS requirements) of some already existing connections in the overloaded situation

- Typical approaches
  - Internal approach: on the overloaded servers
  - External approach: on the front-tier nodes. Main advantages are:
    - Make global decisions
    - More transparent to the overloaded servers
    - Easily applicable to any tier



Motivation

- External approach
  - Front-tier proxy servers need to get load information from back-end servers
- Problems with the existing designs
  - Use TCP/IP – coarse-grained and high overhead; responsiveness depends on load
  - Workload is divergent and unpredictable – require fine-grained and low overhead

Opportunity & Objective

- Opportunity: modern high-speed interconnects
  - iWARP/10-Gigabit Ethernet, InfiniBand, Quadrics etc.
  - High performance: low latency & high bandwidth
  - Novel features: atomic operation, protocol offloading, RDMA operations etc.
  - RDMA: low latency & no communication overhead on the remote node

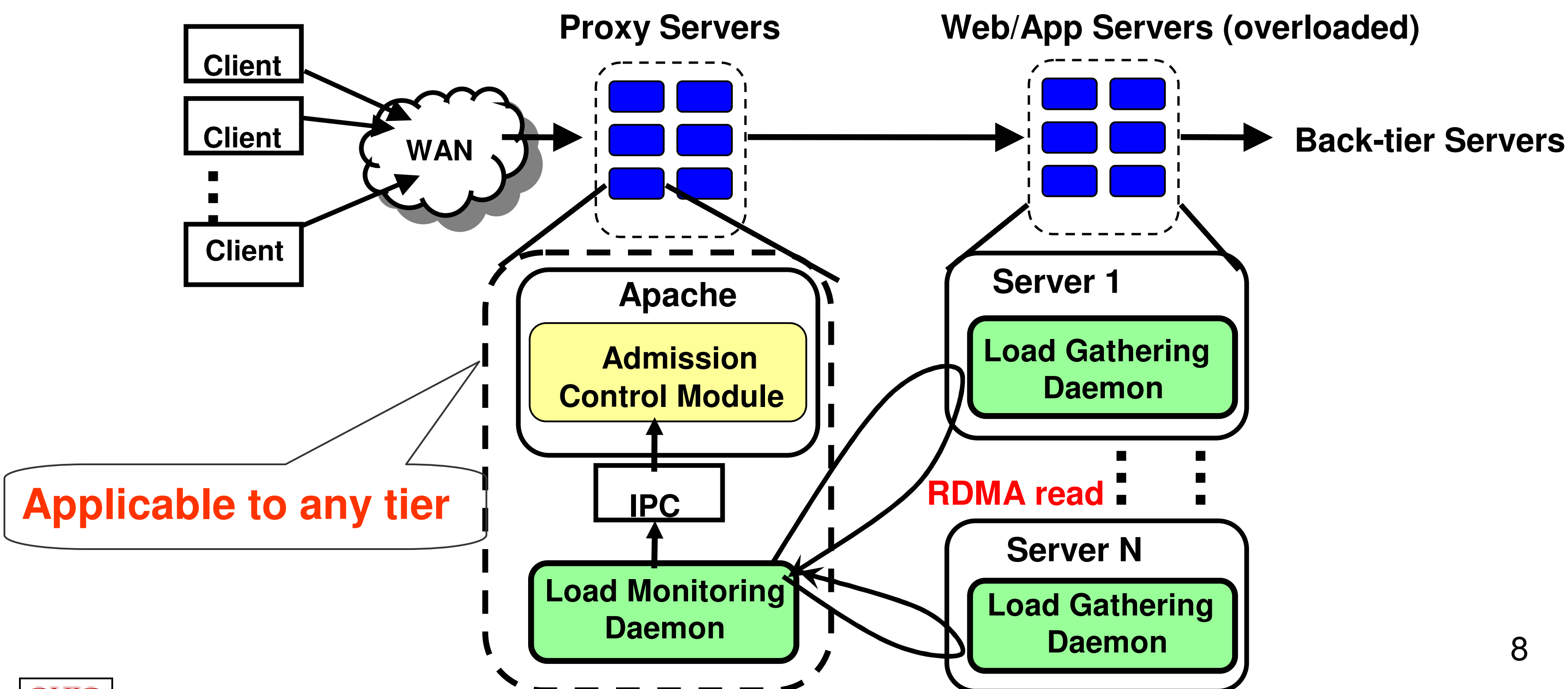
- **Objective**
  - Leverage the advanced features (RDMA operation) to design more efficient, lower overhead and better QoS guaranteed admission control

Outline

- Introduction & Motivation
- Proposed Design
- Experimental Results
- Conclusions & Future Work

System Architecture

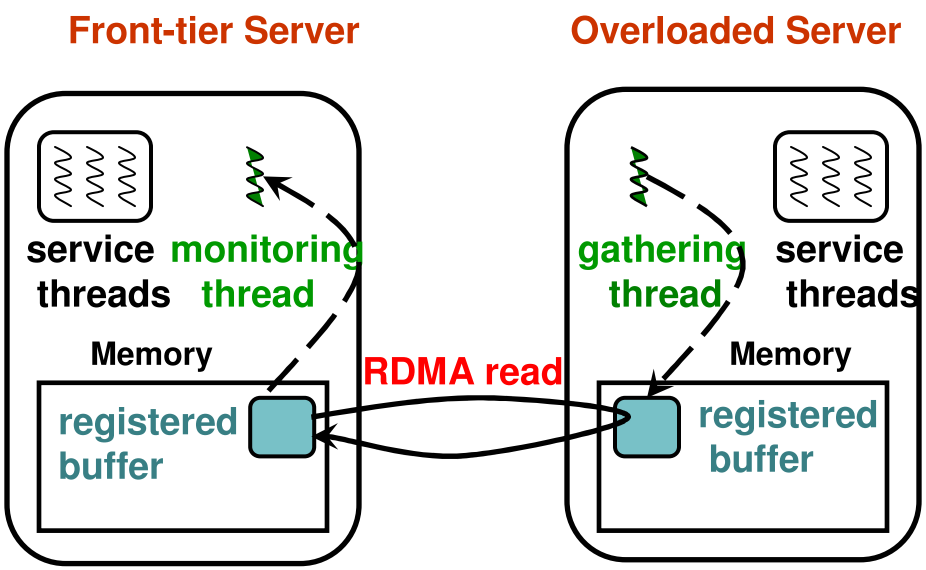
- Load gathering daemon running on overloaded web servers
- Load monitoring daemon running on front-tier proxy servers
- Admission control module running on front-tier proxy servers



Load Gathering and Monitoring Daemon

- Load gathering daemon
  - Running on each of the overloaded servers in background – low overhead
  - Gather instantaneous load information
- Load monitoring daemon
  - Running on each of the front-tier proxy servers
  - Retrieve load information from all the load gathering daemons
- Communication is important!
  - TCP/IP is not good, so?

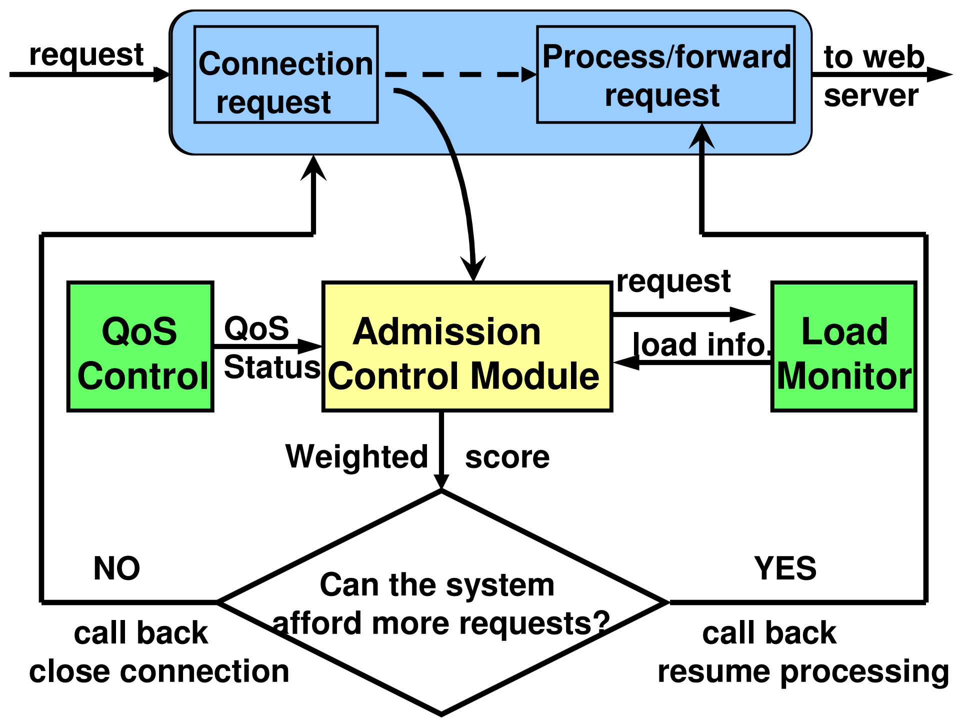
Gathering and Monitoring Daemon Cont.



- Use RDMA read
  - Monitoring daemon issues RDMA read to gathering daemon
    - Buffer must be registered and pinned down before the operation
    - Monitoring daemon has to know the memory address of the remote buffer
  - Retrieve load information at high granularity under overload – better decisions
  - No CPU involvement on the loaded servers – low overhead

Admission Control Module

- Use *shared memory* to communicate with load monitoring daemon
- Attach to Apache: dynamically loadable; trap into Apache request processing
- New processing procedure
  - Apache main thread call the admission control module after TCP connection is established
  - Admission control module uses weighted score to make decisions
  - If all of the back-end servers are overloaded, call back to Apache thread to close the new connections; otherwise, call back to resume the processing



Outline

- Introduction & Motivation
- Proposed Design
- Experimental Results
- Conclusions & Future Work

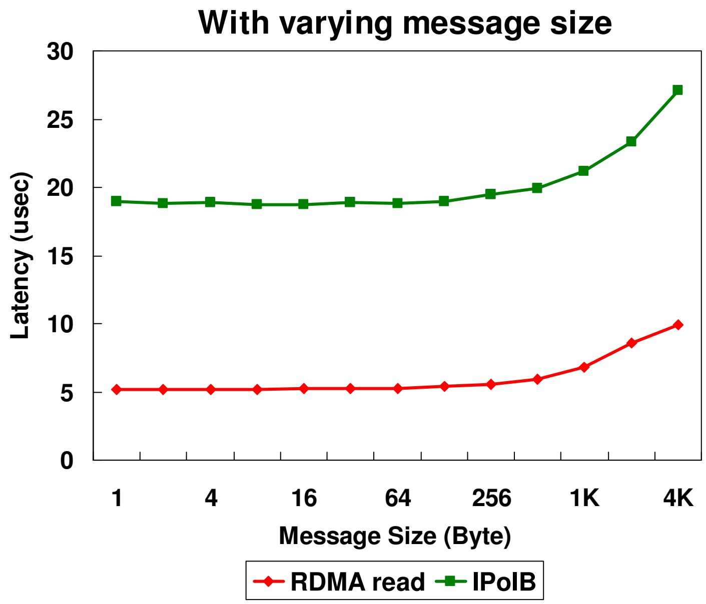
Experimental Platforms

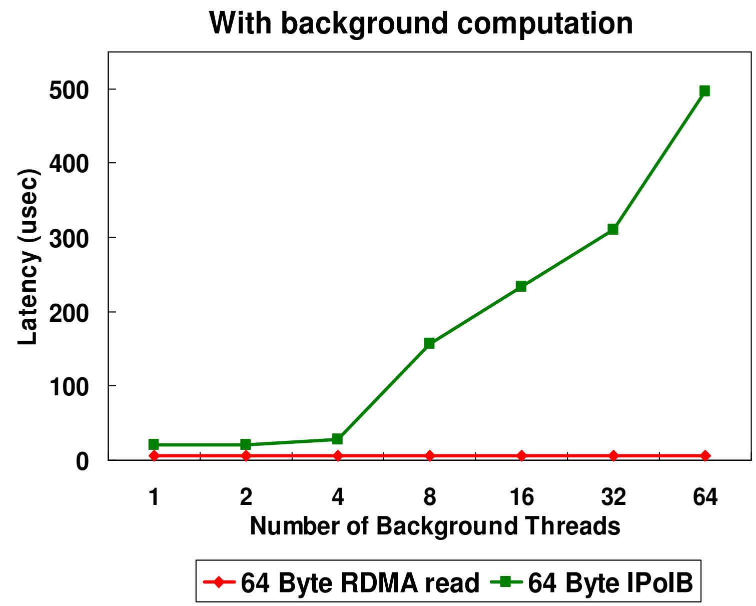
- 32 Compute nodes
  - Dual Intel 64-bit Xeon 3.6 GHz CPU, 2 GB memory
  - Mellanox MT25208 InfiniBand HCA, OFED 1.2 driver
  - Linux 2.6
- Two-tier data-center including proxy servers and web servers; web servers are potentially overloaded
- Apache 2.2.4 for proxy servers and web servers

Experiment Results Outline

- Micro-benchmarks: basic IBA performance
- Data-center level evaluation
  - Single file trace
    - Average response time and aggregate TPS
    - Instant performance analysis
    - QoS analysis
  - Worldcup trace and Zipf trace
    - Worldcup trace: real data from world cup 1998
    - Zipf trace: workloads follow Zipf-like distribution (probability of i'th most popular file $\propto 1/i^{\alpha}$)

Performance of RDMA read and IPoIB (TCP/IP over IBA)





- 1 Byte message
  - RDMA read: 5.2 us
  - IPoIB: 18.9 us
- Improvement using RDMA increases when message size increases

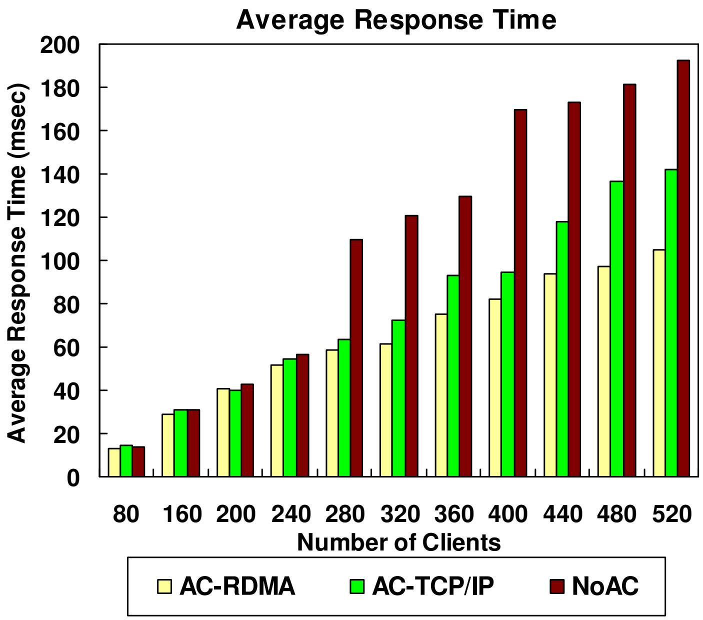
- IPoIB significantly degrades
- RDMA read keeps constant latency

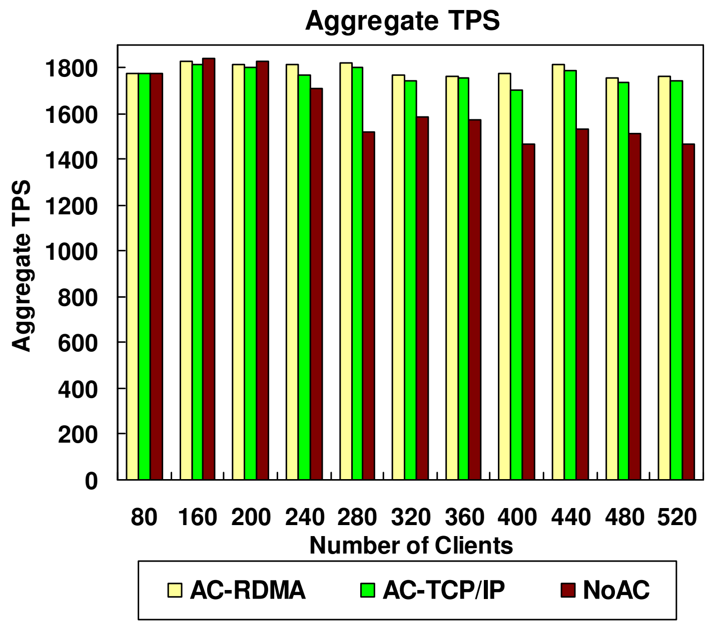
Performance of IPoIB depends on load, while RDMA NOT!

Data-Center level Evaluation

- Configuration
  - 4 nodes used as proxy servers
  - 1 node used as web server
  - Remaining nodes are clients
- Load information updated every 1 ms
- Measured average client-perceived response time (for successful request) and aggregate system TPS
- Comparing performance of three systems
  - **AC-RDMA**: system with admission control based on RDMA read (the proposed approach)
  - **AC-TCP/IP**: system with admission control based on TCP/IP
  - **NoAC**: system without any admission control

Performance with Single File Trace (16 KB)



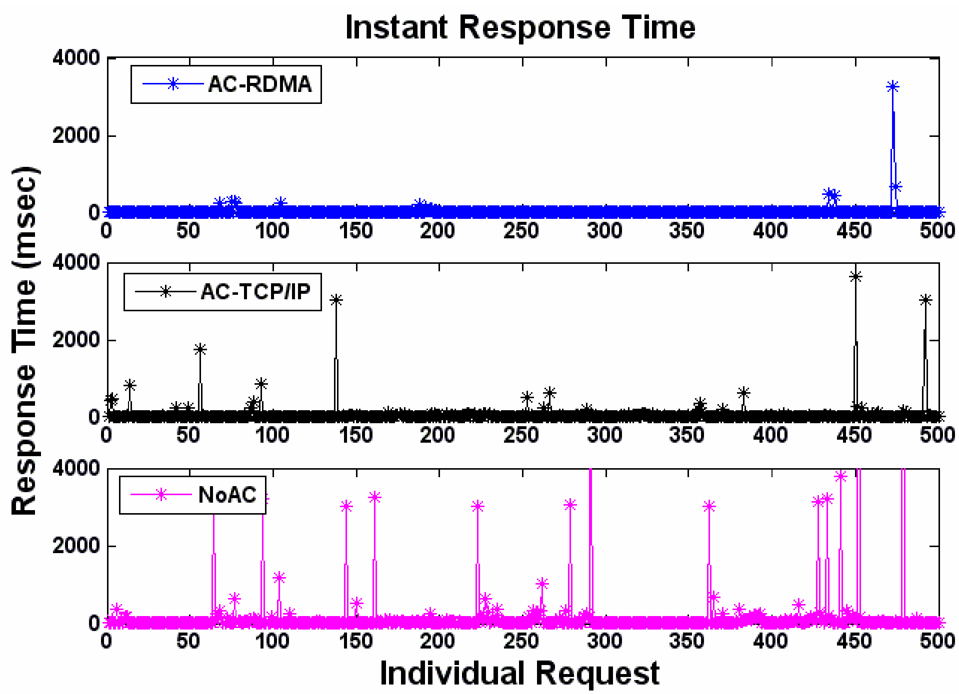


- With 520 clients
  - NoAC: 192.31ms
  - AC-TCP/IP: 142.29ms -26% improvement
  - AC-RDMA: 105.03ms - 26% improvement over AC-TCP/IP (45% improvement over NoAC)

- AC-RDMA and AC-TCP/IP are comparable
- System with admission control has higher TPS than the original system

Instant Performance

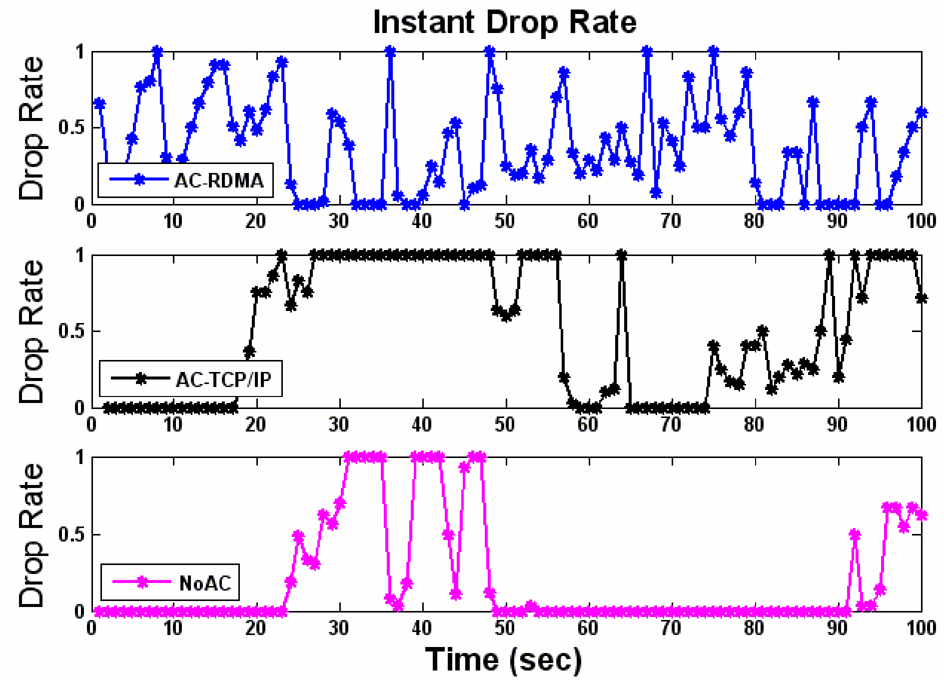
- Workload: 400 clients
- Instant response time
  - NoAC: many requests served with very long time
  - AC-RDMA: almost no such requests
  - AC-TCP/IP: some requests with long response time



Instant performance is consistent with the trend of average response time

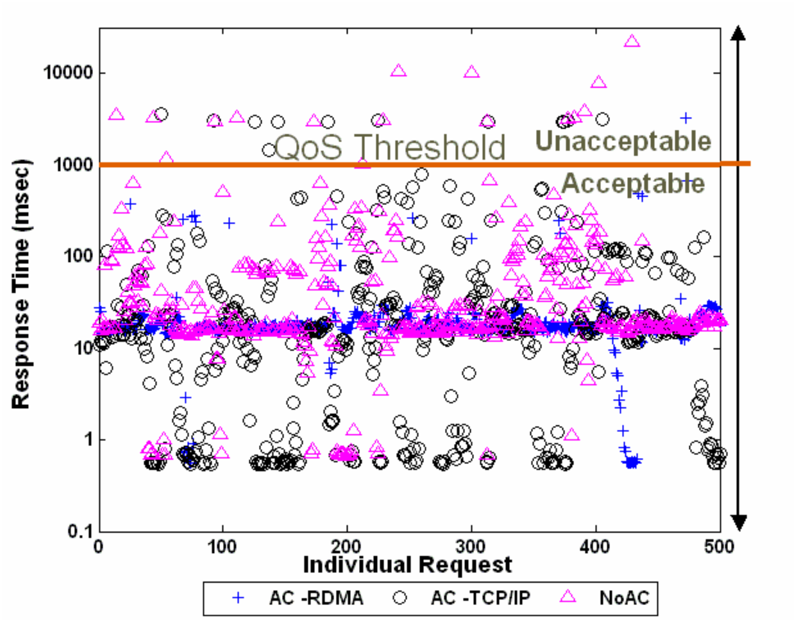
Instant Performance Cont.

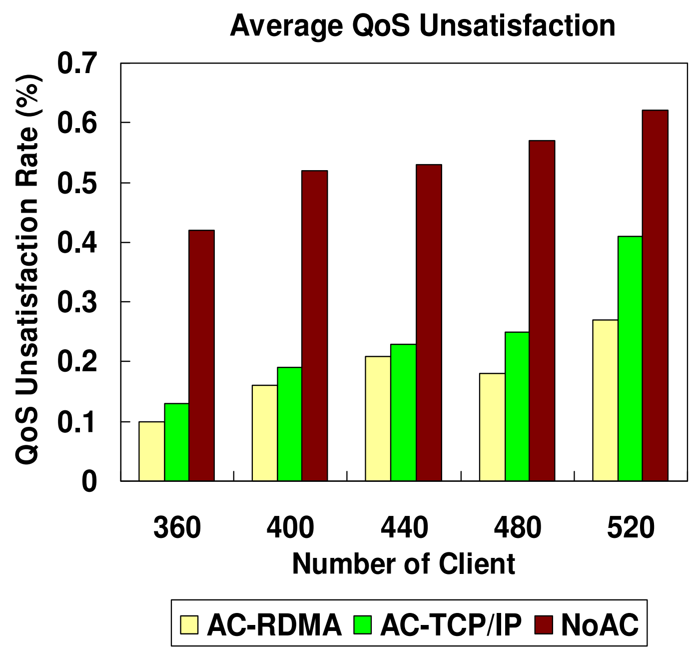
- Instant drop rate
  - AC-RDMA: closely reflects the instantaneous changing load on web servers
  - AC-TCP/IP: longer streak of continuous drops or acceptance
  - NoAC: a lot of acceptance; some drops because of timeout



AC-RDMA gets the load information timely, while AC-TCP/IP sometimes reads the stale information due to the slow response from overloaded servers in TCP/IP communication

QoS Analysis

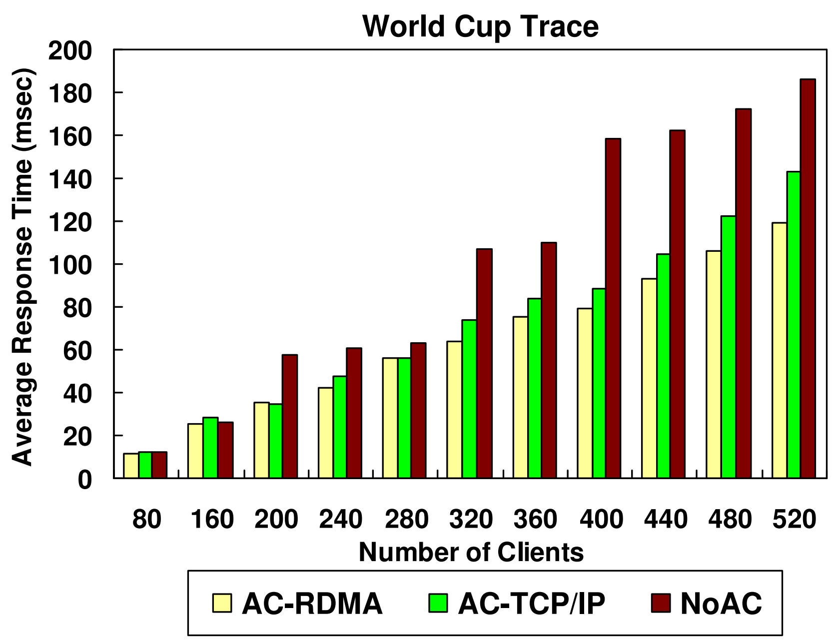


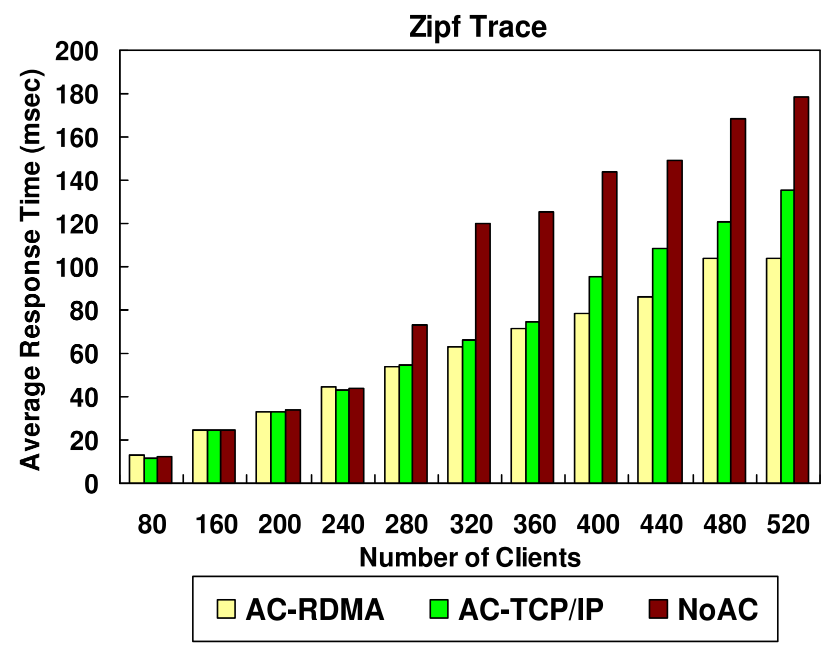


- Instant QoS status
  - AC-RDMA has much better capability of satisfying the QoS requirement
- Average QoS unsatisfaction
  - AC-RDMA is the best

With the same requirement of QoS (e.g., response time), AC-RDMA can serve more clients than the other two systems

Performance with Worldcup and Zipf Trace





- AC-RDMA is better
  - Compared to AC-TCP/IP: 17%
  - Compared to NoAC: 36%

- AC-RDMA is better
  - Compared to AC-TCP/IP: 23%
  - Compared to NoAC: 42%

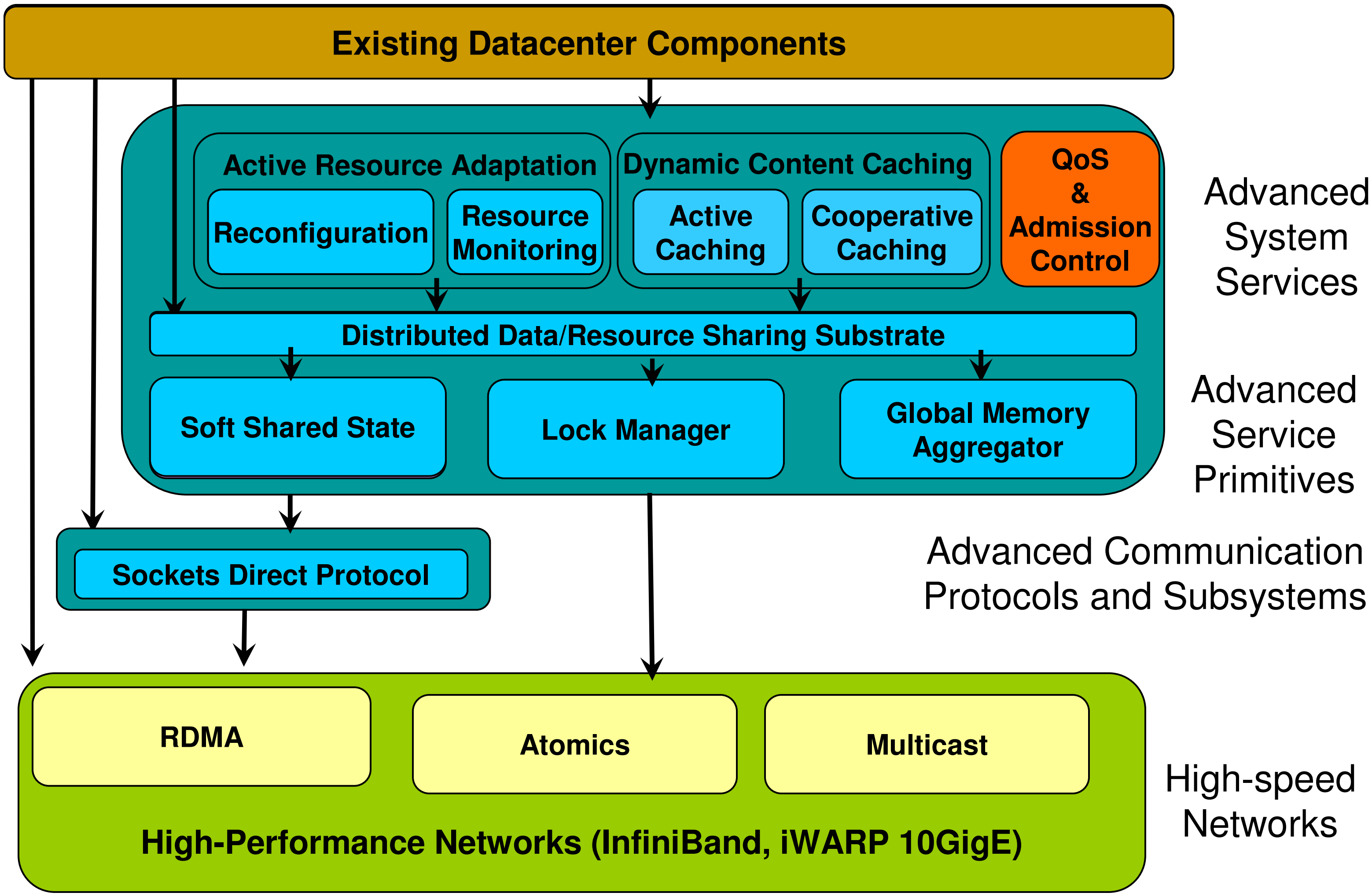
Outline

- Introduction & Motivation
- Proposed Design
- Experimental Results
- Conclusions & Future Work

Conclusions & Future Work

- Leveraged RDMA read in designing efficient admission control mechanism used in multi-tier data-centers
- Implemented the design in a two-tier data-center over InfiniBand
- Evaluated with single file, worldcup trace and Zipf trace
  - AC-RDMA outperforms AC-TCP/IP up to 28%, outperforms NoAC up to 51%
  - AC-RDMA can provide better QoS satisfaction
  - Main reasons
    - Update load information timely
    - No extra overhead on the already overloaded servers
- Future work: study the scalability performance, incorporate other earlier work for integrated resource management service etc.

Overall Datacenter Framework



Thank you

{laipi, narravul, vaidyana, panda}@cse.ohio-state.edu

NBC-LAB

Network-Based Computing Laboratory

http://nowlab.cse.ohio-state.edu/

Data-Center Web Page

http://nowlab.cse.ohio-state.edu/projects/data-centers/index.html