Lightning Talks
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

Problem: Disparity in cloud scheduling designs

- Centralized schedulers $\rightarrow$ High quality, low speed
- Sampling-based schedulers $\rightarrow$ High speed, low quality

Tarcil: Key scheduling techniques to bridge this gap

- Account for resource preferences $\rightarrow$ High decision quality
- Analytical framework for sampling $\rightarrow$ Predictable performance
- Admission control $\rightarrow$ High quality, fast decisions at high load
- Distributed design $\rightarrow$ High scheduling throughput
- 50msec avg scheduling latency, > 95% of tasks meet QoS
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

High-Performance Service-Oriented Computing

Nic McDonald & Bill Dally
Service Oriented Access Control

To CPU

Processor Interconnect Controller

Network Access Controller

Security Logic

Dynamic Memory Allocator

Hash Map Controller

Memory System

To LAN
Service Oriented Access Control

146 TB vs. 5.33 GB
1.12 GB vs. 20.8 MB
Service Oriented Rate Control

Diagram showing a network of nodes connected by high-speed links (5 Gbps and 15 Gbps). The nodes are labeled as Edit, ADs, User DB, View, Mem Cache, and Post DB.
Service Oriented Rate Control

Service A

Proc

Proc

Proc

15 Gbps

Service B

Proc

Proc

Proc
Service Oriented Rate Control

Disadvantages:
- high latency
- bandwidth waste
Service Oriented Rate Control

Disadvantages:
- overhead for transient non-uniformity

15 Gbps
Service Oriented Rate Control

Bandwidth:
112.5% overhead

99.9%ile Latency:
325 cycles
310 cycles
Service Oriented Rate Control

Bandwidth:
0.5% overhead

99.9%ile Latency:
55 cycles
32 cycles
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

Infrastructure and Modules for Building Scalable Control Planes

Collin Lee and John Ousterhout
Exploring Easier Distributed Systems Development

● Distributed systems are necessary but hard to build
  ○ Most of the difficult concentrated in “control planes”
  ○ Needed for scalability, availability, fault-tolerance, or fundamentally distributed applications

● Difficulty makes distributed systems “cost prohibitive” for some applications
  ○ “best” solution is to use a distributed system but it is too hard or time consuming to build

● Can building distributed systems be as easy as single node applications?

● Are there interfaces, abstractions, modules, infrastructures to hide to complexity of developing distributed systems?
Collecting Ideas and Industry Feedback

Possible Applications?
- Storage System Coordinators?
- Cluster/Resource Managers?
- Software Defined Networks?
- Drone Air Traffic Control?
- Self-Driving Car Fleet Coordination?
- <your_application_here>?

Know Challenges?
- Code duplication
  - normal operation
  - durability
  - recovery
- Control node scalability
- <your_frustrations_here>

Have ideas? Please come chat!
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

Nimbus: Running Fast, Distributed Computations with Execution Templates

Omid Mashayekhi
(omidm@stanford.edu)

Hang Qu
Chinmayee Shah
Philip Levis

February 2016
Nimbus: Running Fast, Distributed Computations with Execution Templates
Omid Mashayekhi, Hang Qu, Chinmayee Shah, Philip Levis

- In-memory data analytics has become CPU-bound.
Nimbus: Running Fast, Distributed Computations with Execution Templates
Omid Mashayekhi, Hang Qu, Chinmayee Shah, Philip Levis

- In-memory data analytics has become CPU-bound.

Runtime Overhead ~ 19-32%
In-memory data analytics has become CPU-bound.

- Optimizing applications in a lower level language speeds tasks up.

Runtime Overhead ~ 19-32%
In-memory data analytics has become CPU-bound.

- Optimizing applications in a lower level language speeds tasks up.
- Shorter task means higher task rate which results in excessive runtime overhead.

Runtime Overhead ~ 19-32%

Almost entirely Runtime Overhead
In-memory data analytics has become CPU-bound.
  - Optimizing applications in a lower level language speeds tasks up.
  - Shorter task means higher task rate which results in excessive runtime overhead.

Current scheduling architectures have limited task rate.

<table>
<thead>
<tr>
<th>Spark Limit</th>
<th>Naiad Limit</th>
<th>C++ Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 KTask/sec</td>
<td>2.3 KTask/s</td>
<td>250 KTask/s</td>
</tr>
</tbody>
</table>

Task Rate Limit For Logistic Regression
In-memory data analytics has become CPU-bound.
  - Optimizing applications in a lower level language speeds tasks up.
  - Shorter task means higher task rate which results in excessive runtime overhead.

Current scheduling architectures have limited task rate.

Key insight behind Nimbus is that long running CPU-bound applications are iterative in nature (e.g. ML algorithms, scientific computing, etc.).

Scheduler can memoize and reuse computations as patterns recur.

Execution Templates provide an abstraction for memoizing and reusing the computations and suppressing the command exchange by the scheduler.
Nimbus: Running Fast, Distributed Computations with Execution Templates
Omid Mashayekhi, Hang Qu, Chinmayee Shah, Philip Levis

- Nimbus achieves tasks rates as high as **half a million** tasks per second!

20X speedup for ML benchmarks
HPC applications within the cloud frameworks with negligible overhead (3-11%)
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

Gaining Visibility and Insight into Distributed Systems

Stephen Yang and John Ousterhout
Gaining Visibility and Insight into Distributed Systems

- Problem: Developing/Debugging Distributed Systems is hard due to Limited Visibility
  - Logging can be inefficient

- Proposal: Build an interactive visualization tool to support development of such systems
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

Core-Aware Scheduling: Application Concurrency with Core Availability
Henry Qin & John Ousterhout

- Questing for high throughput in low-latency services
- Can we efficiently multiplex machines between latency-sensitive foreground tasks CPU-intensive background tasks?
- Allocating cores in the kernel, scheduling threads at user level
- More detailed talk in the afternoon.
- Industry representatives please stop by the poster; I have many questions for you!
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”
2. Nic McDonald: “Service-Oriented Rate Control”
4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”
5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”
6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”
7. Sean Choi: “Customizing Open vSwitch using P4”
8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”
Problem Statement

- Can we expedite the process of implementing new network protocols or features?
- Can we do this without incurring a huge performance cost?

Approach

- We present a P4 to OvS compiler
  - System administrators can easily describe the changes as a P4 program
  - P4-OvS switch is aimed to provide similar performance with significantly reduced complexity
Customizing Open vSwitch using P4
Muhammad Shahbaz, Sean Choi, Ben Pfaff, Changhoon Kim, Nick Feamster, Jennifer Rexford, and Nick McKeown

P4-OvS Compiler Specification

P4 Program → P4 Compiler → C Code

Parse, Match, Action → OVS Source Code

Runtime Flow Rules

Flow Rule Type Checker

Match-Action Rules → Slow Path Configuration → OVS Executable
Customizing Open vSwitch using P4
Muhammad Shahbaz, Sean Choi, Ben Pfaff, Changhoon Kim, Nick Feamster, Jennifer Rexford, and Nick McKeown

Throughput Results

![Throughput Results Graph]

- PISCES
- PISCES (Optimized)
- OVS

Packet Size (Bytes)
Throughput (Gbps)
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”

High Speed Networks Need Proactive Congestion Control

Lavanya Jose\textsuperscript{1}, Lisa Yan\textsuperscript{1}, Stephen Ibanez\textsuperscript{1}, Issac Keslassy\textsuperscript{2}, George Varghese\textsuperscript{3}, Sachin Katti\textsuperscript{1}, Mohammad Alizadeh\textsuperscript{4}, and Nick McKeown\textsuperscript{1}
\textsuperscript{1}Stanford, \textsuperscript{2}Technion, \textsuperscript{3}Microsoft, \textsuperscript{4}MIT

**Context**

- Network speed: 10 → 100 Gb/s
- 1 MB / 100 Gb/s = 80 μs

<table>
<thead>
<tr>
<th>Speed (Gb/s)</th>
<th>Completion Time (RTTs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>70-80</td>
</tr>
<tr>
<td>40</td>
<td>17-20</td>
</tr>
<tr>
<td>100</td>
<td>7-8</td>
</tr>
</tbody>
</table>

**Reactive Congestion Control**

- Adjust Flow Rate
- Measure Congestion
High Speed Networks Need Proactive Congestion Control

Problem

- Can we use explicit information (Link Capacities, Traffic Matrix) to find the optimal flow rate allocation more quickly?
High Speed Networks Need Proactive Congestion Control

Problem

- Can we use explicit information (Link Capacities, Traffic Matrix) to find optimal flow rate allocation more quickly?

Approach

- Distributed proactive congestion control
- Message passing between flows and links
High Speed Networks Need Proactive Congestion Control

Approach

- Distributed proactive congestion control
- Message passing between flows and links

Convergence Time

<table>
<thead>
<tr>
<th></th>
<th>RCP (Reactive)</th>
<th>PERC (Proactive)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median</td>
<td>14 RTTs</td>
<td>4 RTTs</td>
</tr>
<tr>
<td>Tail (99th %)</td>
<td>71 RTTs</td>
<td>10 RTTs</td>
</tr>
</tbody>
</table>
High Speed Networks Need Proactive Congestion Control

Using Programmable Data Planes

- Useful platform for deployment of distributed proactive congestion control schemes
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”
2. Nic McDonald: “Service-Oriented Rate Control”
4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”
5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”
6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”
7. Sean Choi: “Customizing Open vSwitch using P4”
8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”
Grazelle: Hardware-Optimized In-Memory Graph Processing

Samuel Grossman, Heiner Litz, and Christos Kozyrakis
Stanford University
Hardware Features

Vector Units  Sequential Accesses

Software Prefetching  NUMA Awareness

Caching Overheads  Simultaneous Multithreading
Hardware Features: X-Stream

- ✗ Vector Units
- ✗ Software Prefetching
- ✔ Caching Overheads
- ✔ Sequential Accesses
- ✗ NUMA Awareness
- ✗ Simultaneous Multithreading
Hardware Features: Polymer

✘ Vector Units

✘ Software Prefetching

~ Caching Overheads

✔ Sequential Accesses

✔ NUMA Awareness

✘ Simultaneous Multithreading
Hardware Features: Grazelle

- ✔ Vector Units
- ✔ Software Prefetching
- ✔ Caching Overheads
- ✔ Sequential Accesses
- ✔ NUMA Awareness
- ✔ Simultaneous Multithreading
Lightning Talks

1. Christina Delimitrou: “Tarcil: Reconciling Scheduling and Quality in Large Shared Clusters”

2. Nic McDonald: “Service-Oriented Rate Control”


4. Omid Mashayekhi: “Nimbus: Running Fast, Distributed Computations with Execution Templates”

5. Stephen Yang: “Gaining Visibility and Insight into Distributed Systems”

6. Henry Qin: “Core-Aware Scheduling: Balancing Application Concurrency with Core Availability”

7. Sean Choi: “Customizing Open vSwitch using P4”

8. Stephen Ibanez: “High Speed Networks Need Proactive Congestion Control”