Composing SDN Controller Enhancements with Mozart

Zhenyu Zhou
Theophilus Benson
Global SDN Market is expected to rise from its initial estimated value of USD 8.92 billion in 2018 to an estimated value of USD 67.98 billion by 2026, registering a CAGR of 28.90% in the forecast period of 2019-2026.
CONTENTS

01 Background
SDNApps’ Assumptions

02 Mozart Design
Abstractions and Interfaces

03 Evaluation
Performance and Overhead
What are SDNs?

SDNApps

Controller

Rule 1
Rule 2

Network
Case Study: Hedera

• Improving data center performance by load balancing elephant flows

Gather network state

Configure new paths

Rule 1

Rule 2

Merged Rule

Controller
What are SDNs?

SDNApps

Controller

Functionality

Rule 1
Rule 2

Merged Rule

Optimization

Network

SDNEnhancements
SDNApps’ Assumptions & SDNEnhancements

**Rule 1**
- Consistent Update
- Fault Tolerance Path

**Rule 2**
- Consistent Update
- Fault Tolerance Path

**Merged Rule**
- Consistent Update
- Fault Tolerance Path

**TCAM-Optimizer**

**Infinite Hardware Resources**

More results can be found in the paper

Controller

Instantaneous Updates

Controller

Unmodified Actions
Case Study: Hedera
Related Works

App 1  App 2  App 3  Composer

Pyretic [NSDI’13]  
Ignores SDNEnhancements

Athens [CoNEXT’14] , Chopin [CoNEXT ’18]  
Requires understanding all SDNEnhancements
What abstractions are required to systematically include SDNEnhancements into the SDN ecosystem?
CONTENTS

01 Background
  SDNApps’ Assumptions

02 Mozart Design
  Abstractions and Interfaces

03 Evaluation
  Performance and Overhead
Another Story: Compiler Optimization

Source Code
- C
- C++
- Java
- Verilog

Intermediate Form
- Front End
- Optimizer
- Back End

Object Code
- Alpha
- SPARC
- x86
- IA-64

Options:
- -fgcse-after-reload
- -fipa-cp-clone
- -floop-interchange
- -floop-unroll-and-jam
- -fpeel-loops
- ...

Image referenced from: https://www.cs.cmu.edu/afs/cs/academic/class/15745-s02/www/lectures/lect1.pdf
Mozart

Source Code
- C
- C++
- Java
- Verilog

Intermediate Form
- Optimizer

Object Code
- Alpha
- SPARC
- x86
- IA-64

-vgcse-after-reload
...

Compiler

SDN

App 1
App 2
App 3

Rule 1
Rule 2

Transaction

Mozart

{IO}
{AD}
{LS}
{PF}

SDN Network
SDN-Flags

<table>
<thead>
<tr>
<th>Message 1</th>
<th>Flag 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message 2</td>
<td>Flag 2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Global Flag</td>
<td></td>
</tr>
</tbody>
</table>

App
Transaction

Rule 1
Rule 2

Merged Rule

Checker

Optimization
SDN-Flags

Order

Match 1 | Action 1
---|---
Match 2 | Action 2

Location
CONTENTS

01 Background
  SDNApps' Assumptions

02 Mozart Design
  Abstractions and Interfaces

03 Evaluation
  Performance and Overhead
Evaluation

• Data-Plane
  
• Control-Plane

• Workloads

Mininet

Fat Tree

FloodLight

Ryu

Realistic DCN

Synthetic
Best - Random
Worst - Stride

> sudo mn
Evaluation

Can Mozart improve performance?

What’s the overhead of Mozart?

What’s Mozart’s benefit for backward compatibility?

How much work does Mozart introduce for SDNApp updates?

More results can be found in the paper
Evaluation

Proactive SDNApp (Hedera)

- Saves **24.8%** reduction in aggregate bandwidth introduced by TCAMOptimizer.

More results can be found in the paper
Evaluation

Mozart Overhead

- Sublinear
- 1.58% to latency

More results can be found in the paper
Conclusion

• SDN controllers are **ill-equipped** with poor primitives for supporting SDNApps and abstractions for enabling SDNEnhancements.

• Mozart interface allows for a **systematic and principled** inclusion of SDNEnhancements into the SDN ecosystem.
Thanks for your attentions!