Scaling Open vSwitch
with a
Computational Cache

Alon Rashelbach
Ori Rottenstreich
Mark Silberstein
Technion, Israel

USENIX NSDI 2022
• Virtual Switch
• Open-source
• Data-centers
Packets
1K OpenFlow Rules $\Rightarrow$ 500K OpenFlow Rules
1K OpenFlow Rules $\Rightarrow$ 500K OpenFlow Rules

14x geomean throughput degradation

In some cases: throughput drops from a few Mpps to a few Kpps
1K OpenFlow Rules $\Rightarrow$ 500K OpenFlow Rules

14x geomean throughput degradation
In some cases: throughput drops from a few Mpps to a few Kpps

In this paper: a geomean speedup of 12x
In this paper: a geomean speedup of 12x

Neural-net inference per packet in OVS data-path

14x geomean throughput degradation

In some cases: throughput drops from a few Mpps to a few Kpps
Background
Challenge
This paper
Integration with OVS
Evaluation
NuevoMatch algorithm for packet classification

**NuevoMatch** algorithm for packet classification

*Src IP* | *Dst IP* | *Action*
---|---|---
10.0.10.* | 56.0.*.* | Port 1
10.0.*.* | 56.22.7.1 | Drop

...
NuevoMatch algorithm for packet classification

OpenFlow rules

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Dst IP</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0.10.*</td>
<td>56.0.<em>.</em></td>
<td>Port 1</td>
</tr>
<tr>
<td>10.0.<em>.</em></td>
<td>56.22.7.1</td>
<td>Drop</td>
</tr>
</tbody>
</table>

NuevoMatch

- RQ-RMI Model
- RQ-RMI Model
- RQ-RMI Model

Unique model architecture
40ns NN inference on CPU

NuevoMatch algorithm for packet classification

Classifies packets on the CPU using Neural Network Inference

OVS Data-path
OVS Data-path

Packet

Exact Match Cache

Exact match on packet header fields

HIT

Perform Action
OVS Data-path

Packet

Exact Match Cache

Exact match
on packet header fields

HIT
Perform Action

MISS

Megaflow Cache

Longest-prefix match
on packet header fields

HIT
Perform Action
OVS Data-path

Exact Match Cache

- Exact match on packet header fields
- HIT → Perform Action
- MISS → Megaflow Cache

Megaflow Cache

- Longest-prefix match on packet header fields
- HIT → Perform Action
- MISS → Upcall To Control-Path

Packet
OVS Data-path

Packet

Exact Match Cache

Exact match on packet header fields

HIT

Perform Action

MISS

Exact Match Cache

Longest-prefix match on packet header fields

HIT

Perform Action

MISS

Upcall to Control-Path

Bottleneck!

See paper for details
Can we use the NuevoMatch algorithm within the OVS data-path?
Can we use the NuevoMatch algorithm within the OVS data-path?

NO!
Challenge: Slow Updates
Challenge: Slow Updates

**NuevoMatch** uses neural-nets to learn rules

**Neural-nets** must be trained when rules change

The **training time** depends on the total number of rules
Challenge: Slow Updates

NuevoMatch uses neural-nets to learn rules

Neural-nets must be trained when rules change

The training time depends on the total number of rules

Train once
Challenge: Slow Updates

NuevoMatch uses neural-nets to learn rules

Neural-nets must be trained when rules change

The training time depends on the total number of rules

Train once × Train fast
Must Support Frequent Updates

100K updates per second (on average)

(*) Katsikas et al. What you need to know about (smart) network interface cards. PAM, 2021.
Must Support Frequent Updates

100K updates per second (on average)

Training an RQ-RMI model with 100K+ rules takes 5–20 minutes!
Must Support Frequent Updates

100K updates per second (on average)

Training an RQ-RMI model with 100K+ rules takes 5-20 minutes!

NuevoMatch update rate is 1000x slower!

(*) Katsikas et al. What you need to know about (smart) network interface cards. PAM, 2021.
Background: RQ-RMI models in NuevoMatch

Range: a single rule field

<table>
<thead>
<tr>
<th>Row num</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-100</td>
</tr>
<tr>
<td>2</td>
<td>130-700</td>
</tr>
<tr>
<td>3</td>
<td>800-803</td>
</tr>
<tr>
<td>4</td>
<td>807-890</td>
</tr>
<tr>
<td>5</td>
<td>970-990</td>
</tr>
<tr>
<td>6</td>
<td>991-991</td>
</tr>
</tbody>
</table>
Background: RQ-RMI models in NuevoMatch

Range: a single rule field

Sample the table for generating \([\text{range, #row}]\) tuples on the fly

<table>
<thead>
<tr>
<th>Row num</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-100</td>
</tr>
<tr>
<td>2</td>
<td>130-700</td>
</tr>
<tr>
<td>3</td>
<td>800-803</td>
</tr>
<tr>
<td>4</td>
<td>807-890</td>
</tr>
<tr>
<td>5</td>
<td>970-990</td>
</tr>
<tr>
<td>6</td>
<td>991-991</td>
</tr>
</tbody>
</table>
Background: RQ-RMI models in NuevoMatch

Range: a single rule field

<table>
<thead>
<tr>
<th>Row num</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0–100</td>
</tr>
<tr>
<td>2</td>
<td>130–700</td>
</tr>
<tr>
<td>3</td>
<td>800–803</td>
</tr>
<tr>
<td>4</td>
<td>807–890</td>
</tr>
<tr>
<td>5</td>
<td>970–990</td>
</tr>
<tr>
<td>6</td>
<td>991–991</td>
</tr>
</tbody>
</table>

Supervised learning on \([\text{range, \#row}]\) tuples

Sample the table for generating \([\text{range, \#row}]\) tuples on the fly
Background: RQ-RMI models in NuevoMatch

<table>
<thead>
<tr>
<th>Row num</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-100</td>
</tr>
<tr>
<td>2</td>
<td>130-700</td>
</tr>
<tr>
<td>3</td>
<td>800-803</td>
</tr>
<tr>
<td>4</td>
<td>807-890</td>
</tr>
<tr>
<td>5</td>
<td>970-990</td>
</tr>
<tr>
<td>6</td>
<td>991-991</td>
</tr>
</tbody>
</table>

Training

- Supervised learning on \([\text{range, \#row}]\) tuples
- Sample the table for generating \([\text{range, \#row}]\) tuples on the fly
Background: RQ-RMI models in NuevoMatch

Inference

Port: 80

RQ-RMI Model

Packet header field

<table>
<thead>
<tr>
<th>Row num</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-100</td>
</tr>
<tr>
<td>2</td>
<td>130-700</td>
</tr>
<tr>
<td>3</td>
<td>800-803</td>
</tr>
<tr>
<td>4</td>
<td>807-890</td>
</tr>
<tr>
<td>5</td>
<td>970-990</td>
</tr>
<tr>
<td>6</td>
<td>991-991</td>
</tr>
</tbody>
</table>
Background
Challenge
This paper
Integration with OVS
Evaluation
introducing NuevoMatch UP
speeding-up NuevoMatch updates
introducing **NuevoMatch UP**

speeding-up NuevoMatch updates

Three orthogonal improvements:

- Bucketizing Ranges
- Efficient SIMD implementation
- Optimized sampling (see paper)
introducing **NuevoMatch UP**

speeding-up NuevoMatch updates

Bucketizing Ranges

<table>
<thead>
<tr>
<th>Row num</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-10</td>
</tr>
<tr>
<td>2</td>
<td>20-30</td>
</tr>
<tr>
<td>3</td>
<td>40-50</td>
</tr>
<tr>
<td>4</td>
<td>60-70</td>
</tr>
<tr>
<td>5</td>
<td>80-90</td>
</tr>
<tr>
<td>6</td>
<td>100-110</td>
</tr>
</tbody>
</table>
introducing **NuevoMatchUP**

speeding-up NuevoMatch updates

**Bucketizing Ranges**

<table>
<thead>
<tr>
<th>Row num</th>
<th>Bucket</th>
<th>Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-30</td>
<td>0-10 20-30</td>
</tr>
<tr>
<td>2</td>
<td>40-70</td>
<td>40-50 60-70</td>
</tr>
<tr>
<td>3</td>
<td>80-110</td>
<td>80-90 100-110</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Row num</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-10</td>
</tr>
<tr>
<td>2</td>
<td>20-30</td>
</tr>
<tr>
<td>3</td>
<td>40-50</td>
</tr>
<tr>
<td>4</td>
<td>60-70</td>
</tr>
<tr>
<td>5</td>
<td>80-90</td>
</tr>
<tr>
<td>6</td>
<td>100-110</td>
</tr>
</tbody>
</table>
introducing NuevoMatchUP
speeding-up NuevoMatch updates

Bucketizing Ranges

<table>
<thead>
<tr>
<th>Row num</th>
<th>Bucket</th>
<th>Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-30</td>
<td>0-10 20-30</td>
</tr>
<tr>
<td>2</td>
<td>40-70</td>
<td>40-50 60-70</td>
</tr>
<tr>
<td>3</td>
<td>80-110</td>
<td>80-90 100-110</td>
</tr>
</tbody>
</table>
introducing **NuevoMatchUP**
speeding-up NuevoMatch updates

Bucketizing Ranges

<table>
<thead>
<tr>
<th>Row num</th>
<th>Bucket</th>
<th>Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-30</td>
<td>0-10 20-30</td>
</tr>
<tr>
<td>2</td>
<td>40-70</td>
<td>40-50 60-70</td>
</tr>
<tr>
<td>3</td>
<td>80-110</td>
<td>80-90 100-110</td>
</tr>
</tbody>
</table>

![Graph showing training and lookup times with bucket size l]
introducing NuevoMatchUP
speeding-up NuevoMatch updates

Bucketizing Ranges

<table>
<thead>
<tr>
<th>Row num</th>
<th>Bucket</th>
<th>Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-30</td>
<td>0-10, 20-30</td>
</tr>
<tr>
<td>2</td>
<td>40-70</td>
<td>40-50, 60-70</td>
</tr>
<tr>
<td>3</td>
<td>80-110</td>
<td>80-90, 100-110</td>
</tr>
</tbody>
</table>
introducing NuevoMatchUP

speeding-up NuevoMatch updates

Efficient SIMD Implementation
introducing **NuevoMatch UP**
speeding-up NuevoMatch updates

Efficient SIMD Implementation

500K Rules

![Bar Chart](chart.png)

- **Training (s)**
- **Rule-set**

 NuevoMatch – TensorFlow & Native C Code
introducing **NuevoMatchUP**

speeding-up NuevoMatch updates

**Efficient SIMD Implementation**

500K Rules

![Training Time Graph](image)

- **NuevoMatch – TensorFlow & Native C Code**
- **NuevoMatchUp – Hand-tailored SIMD**
introducing NuevoMatchUP
speeding-up NuevoMatch updates

Efficient SIMD Implementation

22x geomean improvement

500K Rules

training (s)

Rule-set

1 2 3 4 5 6 7 8 9 10 11 12

NuevOMatch – TensorFlow & Native C Code
NuevOMatchUp – Hand-tailored SIMD
introducing **NuevoMatchUP**

speeding-up NuevoMatch updates

Together:

**Three orders of magnitude** faster training over NuevoMatch
NuevoMatchUP can be integrated into OVS data-path!

Three orders of magnitude faster training over NuevoMatch

Together:

introducing NuevoMatchUP

speeding-up NuevoMatch updates
Background
Challenge
This paper
Integration with OVS
Evaluation
NuevoMatchUP → OvS

Two Design Options for OVS data-path
NuevoMatchUP → OvS

Two Design Options

(1) OVS with a Computational Cache

Packet → Exact Match Cache → MISS → NuevoMatchUP. Megaflows → MISS → Megaflow Cache → MISS → Upcall

This is new
Two Design Options

(1) OVS with a **Computational Cache**

- Packet
  - Exact Match Cache
  - MISS
  - **NuevoMatchUP**
  - MISS
  - Megaflow Cache
  - MISS
  - Upcall

**This is new**

- Works with **Megaflows**
- Minor changes to OVS
NuevoMatchUP \(\rightarrow\) OvS

Two Design Options

(1) OVS with a Computational Cache

Packet \(\rightarrow\) Exact Match Cache \(\rightarrow\) MISS NuevoMatchUP Megaflows \(\rightarrow\) MISS Megaflow Cache \(\rightarrow\) Upcall

Computational Cache

This is new

Works with Megaflows
Minor changes to OVS

Major Bottleneck!
NuevoMatchUP → OvS

Two Design Options

(1) OVS with a Computational Cache

Packet

Exact Match Cache ➔ MISS ➔ NuevoMatchUP: Megaflows ➔ MISS ➔ Megaflow Cache ➔ MISS ➔ Upcall

Everything here is new

(2) OVS with Computational Flows

Packet

Exact Match Cache ➔ MISS ➔ NuevoMatchUP: Computational Flows ➔ Control Path Constant Updates

Everything here is new
NuevoMatchUP \rightarrow OvS

Two Design Options

(1) OVS with a Computational Cache

Packet ~-> Exact Match Cache ~-> MISS ~-> NuevoMatchUP: Megaflows ~-> MISS ~-> Megaflow Cache ~-> MISS ~-> Upcall

Computational Cache

(2) OVS with Computational Flows

Packet ~-> Exact Match Cache ~-> MISS ~-> NuevoMatchUP: Computational Flows ~-> Control Path

Everything here is new

A paradigm shift: data-path matching on OpenFlow rules

No Upcalls!
A paradigm shift: data-path matching on OpenFlow rules

No Upcalls!

NuevoMatchUP → OvS

Two Design Options

(1) OVS with a Computational Cache

Packet

Miss

Exact Match Cache

NuevoMatchUP: Megaflows

Miss

Megaflow Cache

Upcall

(2) OVS with Computational Flows

Packet

Miss

Exact Match Cache

NuevoMatchUP: Computational Flows

Everything here is new

We design, implement, and integrate both into OVS DPDK data-path

A paradigm shift: data-path matching on OpenFlow rules

No Upcalls!
Background
Challenge
This paper
Integration with OVS
Evaluation
End to End Results

10Gbps NICs
ClassBench
CAIDA
Max 1% Drops
NuevoMatchUP → OvS

End to End Results

Num. of OpenFlow Rules

Thr. (Mpps)

Rule-set

Speedups

1K

500K

10^1

10^0

10^{-1}

10^{-2}
End to End Results

Num. of OpenFlow Rules

OVS (baseline)
NuevoMatchUP → OvS

End to End Results

Num. of OpenFlow Rules

Thr. (Mpps)

Rule-set

Speedups

1K

500K

1.2 0.9 1.3 1.3 1.6 1.0 0.7 0.4 1.7 0.7 0.9 2.0 2.4 2.4 1.0 2.0 1.0 2.1 5.0 4.0 3.0 0.8

OVS (baseline)

OVS w/ Computational Cache (Fist Option)
NuevoMatchUP → OvS

End to End Results

Num. of OpenFlow Rules

Thr. (Mpps)

Rule-set

Speedups

10^-2

10^-1

10^0

10^1

1K

1.6

500K

2.0

OVS (baseline)

OVS w/ Computational Cache (First Option)
NuevoMatchUP → OvS

End to End Results

Graph showing the relationship between the number of OpenFlow rules and throughput (Mpps) with speedups. The x-axis represents the number of rule-sets (1 to 12), and the y-axis represents throughput in Mpps on a logarithmic scale.

Legend:
- OVS (baseline)
- OVS w/ Computational Flows (no upcalls)
End to End Results

Num. of OpenFlow Rules

Thr. (Mpps)
NuevoMatchUP ➔ OVS
Open vSwitch

Scalable to #Cores

Thr. (Mpps)

Max. Throughput

1K Rules

Number of cores

OVS (baseline)
NuevoMatchUP ➔ OvS

Scalable to #Cores

1K Rules

![Graph showing scalability to number of cores](image)

- **OVS (baseline)**
- **OVS w/ Computational Cache (first option)**

Number of cores: 2, 3, 4, 5, 6, 7, 8, 9, 10

Max. Throughput: 16 Mpps

Throughput (Mpps): 0, 4, 8, 12, 16

Factors: 1.3X, 1.3X, 1.3X, 1.3X, 1.3X, 1.3X, 1.3X, 1.2X
NuevoMatchUP → OvS
Scalable to #Cores

Number of cores

0 1 2 3 4 5 6 7 8 9 10

Thr. (Mpps)

0 4 8 12 16

Max. Throughput

1K Rules

OVS (baseline)
OVS w/ Computational Cache (first option)
OVS w/ Computational Flows (no upcalls)
See Paper for

1. Throughput over time experiments
2. Throughput in the presence of updates
3. Microbenchmarks
4. And more...
Conclusions

1. Fast neural-net training for high update rates
2. New design of the OVS data-path
3. Readily deployable to commodity hardware
Future Work

1. NuevoMatchUP for P4
2. Reducing the need for training
3. Hardware acceleration
Future Work

1. NuevoMatchUP for P4
2. Reducing the need for training
3. Hardware acceleration

Thank You