Cebinae:
Scalable In-network Fairness Augmentation

Liangcheng Yu, John Sonchack, Vincent Liu
Public networks care about fairness

RTT' ≠ RTT

RTT

CCA

CCA' ≠ CCA

Increasing heterogeneity

Public Networks

RTT

CCA

RTT

Inter-CCA Unfairness

RTT

CCA Unfairness

RTT' ≠ RTT

Increasing heterogeneity
Fairness enforcement at the end hosts?

- Hard to deploy and upgrade the same CCA
- Few incentives for self-policing mechanism
In-network fairness enforcement

Incentives coming from operators of the in-network devices
In-network fairness enforcement

• Existing approaches suffer from limited practicalities
  • Assumption: specialized hardware for per-flow queues, end-host cooperation…
In-network fairness enforcement

• Existing approaches suffer from limited practicalities
  • **Assumption**: specialized hardware for per-flow queues, end-host cooperation…

• AFQ [NSDI ’18]: practical emulation of ideal FQ on COTS hardware
  • **Constraints**: e.g., # priorities, queues, buffers
In-network fairness enforcement

• Existing approaches suffer from limited practicalities
  • **Assumption**: specialized hardware for per-flow queues, end-host cooperation…

• AFQ [NSDI ’18]: practical emulation of ideal FQ on COTS hardware
  • **Constraints**: e.g., # priorities, queues, buffers

*Challenging to strictly enforce FQ on each individual flow*
Cebinae: a simpler approach

• Relaxation of fairness \textit{at every instance in time}
  • Penalize/redistribute BW from flows exceeding fair share to others
Cebinae: a simpler approach

• Relaxation of fairness **at every instance in time**
  • Penalize/redistribute BW from flows exceeding fair share to others

• **Binary classification** of flows
  • Efficiently implement various subroutines (e.g., leaky-bucket filter)
Cebinae: a simpler approach

- Relaxation of fairness **at every instance in time**
  - Penalize/redistribute BW from flows exceeding fair share to others
- **Binary classification** of flows
  - Efficiently implement various subroutines (e.g., leaky-bucket filter)

**Cebinae router architecture for binary taxation**

- **Zero modifications and coordinations** to/with legacy host CCAs
- Requirement of only **two queues/priorities**
- Compatibility with CCAs operating on **both loss and delay** signals
Outline

1. Conceptual foundation for binary classification
2. Cebinae’s taxation mechanism
3. Evaluation
Max-min fairness

For every flow $i$ there exists at least one bottleneck link $l$ where:

1. $l$ is saturated
2. $r_i$ is among the largest flows sharing the link $l$
An allocation of rates \( \{r_i\} \)

For every flow \( i \) there exists at least one bottleneck link \( l \) where:

1. \( l \) is saturated
2. \( r_i \) is among the largest flows sharing the link \( l \)

Implication: distributed verification of max-min fairness
Local verification

Each link $l$ can determine the set of bottlenecked flows:

If $l$ non-saturated:

**All flows not bottlenecked**

Else, for each flow $i$:

If $i$ is among $l$’s largest rate(s)

$i$ is **bottlenecked** at $l$

Else

$i$ is **not bottlenecked** at $l$
Local verification

Each link $l$ can determine the set of bottlenecked flows:

If $l$ non-saturated:

**All flows not bottlenecked**

Else, for each flow $i$:

If $i$ is among $l$’s largest rate(s)  

\[ i \text{ is bottlenecked at } l \]

Else

\[ i \text{ is not bottlenecked at } l \]

**Observation:**

1. Each conditional can be determined using *only local information*

2. **Binary classification:** bottlenecked ($\top$), not bottlenecked ($\bot$)
Naive enforcement

Each link $l$ can determine the set of bottlenecked flows:

If $l$ non-saturated:

\[ \text{NOP} \]

Else, for each flow $i$:

If $i$ is among $l$’s largest rate(s)

\[ \text{Drop packets of all } i \text{ is per their current rate} \]

Else

\[ \text{NOP} \]
Naive enforcement

Each link $l$ can determine the set of bottlenecked flows:

If $l$ non-saturated:

\[ \text{NOP} \]

Else, for each flow $i$:

If $i$ is among $l$’s largest rate(s)

\[ \text{Drop packets of all } i \text{ is per their current rate} \]

Else

\[ \text{NOP} \]

**Drawbacks:**

1. Can not push an already-unfair allocation fair
2. CCAs may not be responsive to loss signals
Cebinae taxation

Each link $l$ can determine the set of bottlenecked flows:

If $l$ non-saturated:

NOP

Else, for each flow $i$:

If $i$ is among $l$’s largest rate(s)

**Penalize** $i$ with their **taxed rate**

Else

NOP

**Note:**

1. Penalty box includes **non-loss signals** such as delay

2. Taxed rate to **collectively redistribute** the bandwidth to non-bottlenecked flows
Each link $l$ can determine the set of bottlenecked flows: If non-saturated: NOP Else, for each flow $i$: If $i$ is among $l$'s largest rates: Penalize $i$ with their taxed rate Else: NOP

Note:
1. Penalty box includes non-loss signals such as delay
2. Taxed rate to collectively redistribute the bandwidth to non-bottlenecked flows

Tax bottlenecked-flows exceeding fair bandwidth share

Redistribute to non-bottlenecked flows
Instantiation: Cebinae router architecture

1. **Egress-pipeline cache**: port saturation and $T$ flow status tracking

2. **Ingress-pipeline leaky-bucket filter**: $T$ flow taxation

3. **Local CPU dynamic shuffling agent**: egress state polling and reconfiguration of $T$ flow membership, redistributed rates, and queues
Normal operation

Ingress

Classifier

Membership Classification

Egress

Control Plane

headq

~headq
Normal operation

Dynamic rate enforcement with 2 flow groups and FIFO queues
Normal operation

Ingress

Classifier

LBF

Headq

~Headq

Egress

All buffer is available at all times

Control Plane

Port Saturation
Detector
Flow Bottleneck
Detector
Ingress
Egress

Control Plane
Normal operation

No \( \bot \) should be taxed, i.e., no false positives
Per-round reconfiguration
Per-round reconfiguration

Virtual pacing: guarantee no reordering and avoid violation of draining deadline in the worst case
Per-round reconfiguration

Atomic transactions: LBF states and egress caches
Implementation and evaluation

Hardware prototype on a Wedge100BF Tofino switch testbed and NS-3 module

- Is Cebinae agnostic to CCAs?
- Can Cebinae mitigates unfairness (RTT, inter-CCA)?
- Can Cebinae move towards max-min fairness?
- Is Cebinae easy to configure?
- Does Cebinae resource usage scale?
- ...

Cebinae mitigates unfairness

16 TCP Vegas (0–15) v.s. 1 NewReno (16)
Cebinae mitigates unfairness

Mitigates the *skewed and persistent unfairness* with little efficiency impact: *JFI from 0.093 to 0.984*
Cebinae mitigates unfairness

Preventing aggressiveness

Mitigating starvation
Cebinae mitigates unfairness

**Preventing aggressiveness**

CDF of Goodput [Mbps]

**Mitigating starvation**

CDF of Goodput [Mbps]
# Cebinae mitigates unfairness

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FIFO</td>
<td>FQ</td>
<td>Cebinae</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{20.8, 28}</td>
<td>250</td>
<td>{NewReno:2, NewReno:8}</td>
<td>98.95</td>
<td>95.62</td>
<td>95.92</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{20.4, 40}</td>
<td>350</td>
<td>{Cubic:8, Cubic:2}</td>
<td>98.96</td>
<td>98.95</td>
<td>98.00</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{20.4, 60}</td>
<td>500</td>
<td>{Vegas:2, Vegas:8}</td>
<td>98.88</td>
<td>98.83</td>
<td>98.88</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{200}</td>
<td>1700</td>
<td>{NewReno:16, Cubic:1}</td>
<td>98.28</td>
<td>90.99</td>
<td>94.53</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{100}</td>
<td>850</td>
<td>{NewReno:16, Cubic:1}</td>
<td>98.72</td>
<td>91.45</td>
<td>95.58</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{50}</td>
<td>420</td>
<td>{NewReno:16, Cubic:1}</td>
<td>98.90</td>
<td>93.86</td>
<td>95.37</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{50}</td>
<td>420</td>
<td>{Vegas:16, Cubic:1}</td>
<td>98.90</td>
<td>98.90</td>
<td>95.47</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{100}</td>
<td>850</td>
<td>{Vegas:16, NewReno:1}</td>
<td>98.71</td>
<td>97.77</td>
<td>95.67</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{100}</td>
<td>850</td>
<td>{Vegas:128, NewReno:1}</td>
<td>98.88</td>
<td>98.74</td>
<td>97.45</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{60}</td>
<td>500</td>
<td>{Vegas:8, NewReno:8, Cubic:2}</td>
<td>98.87</td>
<td>98.02</td>
<td>96.52</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{5}</td>
<td>420</td>
<td>{NewReno:32, Cubic:8}</td>
<td>989.8</td>
<td>989.8</td>
<td>985.4</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:128, Cubic:1}</td>
<td>989.8</td>
<td>989.8</td>
<td>968.0</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:1024, Cubic:2}</td>
<td>989.8</td>
<td>989.8</td>
<td>949.2</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{50}</td>
<td>4200</td>
<td>{NewReno: 128, BBR: 1}</td>
<td>988.7</td>
<td>923.6</td>
<td>981.6</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{50}</td>
<td>4200</td>
<td>{NewReno: 128, BBR: 2}</td>
<td>988.9</td>
<td>953.9</td>
<td>979.9</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{50}</td>
<td>21000</td>
<td>{NewReno: 128, BBR: 2}</td>
<td>988.8</td>
<td>953.9</td>
<td>963.8</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100}</td>
<td>8350</td>
<td>{NewReno: 128, BBR: 2}</td>
<td>986.9</td>
<td>938.2</td>
<td>956.3</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:64, NewReno:1}</td>
<td>989.8</td>
<td>898.9</td>
<td>976.2</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100}</td>
<td>8500</td>
<td>{Vegas:4, NewReno:128}</td>
<td>986.9</td>
<td>917.6</td>
<td>957.3</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100, 64}</td>
<td>8500</td>
<td>{Vegas:4, NewReno:128}</td>
<td>988.4</td>
<td>941.1</td>
<td>959.8</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100}</td>
<td>8500</td>
<td>{Vegas:8, NewReno:128}</td>
<td>987.0</td>
<td>936.1</td>
<td>964.4</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:128, BBR:1}</td>
<td>989.8</td>
<td>898.9</td>
<td>987.3</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100}</td>
<td>8500</td>
<td>{Vegas:8, Cubic:32}</td>
<td>985.1</td>
<td>960.3</td>
<td>952.6</td>
</tr>
<tr>
<td>10 Gbps</td>
<td>{50, 44}</td>
<td>41667</td>
<td>{NewReno:128, Cubic:16}</td>
<td>9876</td>
<td>9705</td>
<td>9780</td>
</tr>
<tr>
<td>10 Gbps</td>
<td>{28, 28}</td>
<td>25000</td>
<td>{NewReno:128, Cubic:128}</td>
<td>9891</td>
<td>9856</td>
<td>9787</td>
</tr>
</tbody>
</table>
Cebinae is agnostic to CCAs

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>CCAs</td>
<td>FIFO</td>
<td>FQ</td>
<td></td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{20.8, 28}</td>
<td>250</td>
<td>{NewReno:2, NewReno:8}</td>
<td>98.95</td>
<td>95.62</td>
<td>95.92</td>
</tr>
<tr>
<td></td>
<td>{20.4, 40}</td>
<td>350</td>
<td>{Cubic:8, Cubic:2}</td>
<td>98.96</td>
<td>98.95</td>
<td>98.00</td>
</tr>
<tr>
<td></td>
<td>{20.4, 60}</td>
<td>500</td>
<td>{Vegas:2, Vegas:8}</td>
<td>98.88</td>
<td>98.83</td>
<td>98.88</td>
</tr>
<tr>
<td></td>
<td>{200}</td>
<td>1700</td>
<td>{NewReno:16, Cubic:1}</td>
<td>98.28</td>
<td>90.99</td>
<td>94.53</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{100}</td>
<td>850</td>
<td>{NewReno:16, Cubic:1}</td>
<td>98.72</td>
<td>91.45</td>
<td>95.58</td>
</tr>
<tr>
<td></td>
<td>{50}</td>
<td>420</td>
<td>{Vegas:16, Cubic:1}</td>
<td>98.90</td>
<td>93.86</td>
<td>95.37</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{50}</td>
<td>420</td>
<td>{Vegas:16, Cubic:1}</td>
<td>98.90</td>
<td>98.90</td>
<td>95.47</td>
</tr>
<tr>
<td></td>
<td>{100}</td>
<td>850</td>
<td>{Vegas:16, NewReno:1}</td>
<td>98.71</td>
<td>97.77</td>
<td>95.67</td>
</tr>
<tr>
<td></td>
<td>{100}</td>
<td>850</td>
<td>{Vegas:128, NewReno:1}</td>
<td>98.88</td>
<td>98.74</td>
<td>97.45</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>{60}</td>
<td>500</td>
<td>{Vegas:8, NewReno:8, Cubic:2}</td>
<td>98.87</td>
<td>98.02</td>
<td>96.52</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{5}</td>
<td>420</td>
<td>{NewReno:32, Cubic:8}</td>
<td>98.98</td>
<td>98.98</td>
<td>98.54</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:128, Cubic:1}</td>
<td>98.89</td>
<td>98.89</td>
<td>968.0</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:1024, Cubic:2}</td>
<td>98.98</td>
<td>98.98</td>
<td>949.2</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{50}</td>
<td>4200</td>
<td>{NewReno:128, BBR:1}</td>
<td>98.87</td>
<td>923.6</td>
<td>981.6</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{50}</td>
<td>4200</td>
<td>{NewReno:128, BBR:2}</td>
<td>98.89</td>
<td>953.9</td>
<td>979.9</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{50}</td>
<td>21000</td>
<td>{NewReno:128, BBR:2}</td>
<td>98.88</td>
<td>953.9</td>
<td>963.8</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100}</td>
<td>8350</td>
<td>{NewReno:128, BBR:2}</td>
<td>98.69</td>
<td>938.2</td>
<td>956.3</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:64, NewReno:1}</td>
<td>98.98</td>
<td>98.98</td>
<td>976.2</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100}</td>
<td>8500</td>
<td>{Vegas:4, NewReno:128}</td>
<td>98.69</td>
<td>917.6</td>
<td>957.3</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100, 64}</td>
<td>8500</td>
<td>{Vegas:4, NewReno:128}</td>
<td>98.84</td>
<td>941.1</td>
<td>959.8</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{100}</td>
<td>8500</td>
<td>{Vegas:4, NewReno:128}</td>
<td>98.70</td>
<td>936.1</td>
<td>964.4</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>850</td>
<td>{Vegas:128, BBR:1}</td>
<td>98.98</td>
<td>989.8</td>
<td>987.3</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>{10}</td>
<td>8500</td>
<td>{Vegas:2, Cubic:32}</td>
<td>98.51</td>
<td>960.3</td>
<td>952.6</td>
</tr>
<tr>
<td>10 Gbps</td>
<td>{50, 44}</td>
<td>41667</td>
<td>{NewReno:128, Cubic:16}</td>
<td>98.76</td>
<td>9705</td>
<td>9780</td>
</tr>
<tr>
<td>10 Gbps</td>
<td>{28, 28}</td>
<td>25000</td>
<td>{NewReno:128, Cubic:128}</td>
<td>98.91</td>
<td>9856</td>
<td>9787</td>
</tr>
</tbody>
</table>
Summary

• **No modifications nor coordinations** to/with legacy host CCAs
  • Real-time switch architecture serializing in-network compute modules

• COTS hardware and **minimal resource requirements**
  • Two queues/priorities are sufficient

• Compatible with CCAs using **both loss and non-loss signals**
  • Generic support of a wide range of Internet CCAs and environments

https://github.com/eniac/Cebinae

Thank you for your attention!