PIQoS: A Programmable and Intelligent QoS Framework

Udaya Lekhala and Israat Haque
Computer Science
Dalhousie University, Halifax
Importance of Quality of Service (QoS)

• Different applications (e.g., VoIP, file-transfer) demand different quality factors (e.g., delay, throughput).
• Service providers must satisfy users as per the QoS demand of their applications.
• Demand-based resource allocation is a must to offer QoS.
Dynamic and Intelligent QoS

• According to CISCO VNI 2018 report
  • Broadband speed will nearly double by 2022.
  • IP video traffic will globally be 82% of all IP traffic by 2022.
  • Internet gaming traffic will grow nine fold from 2017 to 2022.

• The explosion of applications and traffic dynamics pose challenges on QoS
  • Congestion, high bandwidth demand, dynamically changing network conditions, applications’ requirements, and failures

• Automated error detection and root cause analysis can lead to an efficient QoS provisioning
  • We need a programmable and intelligent QoS framework
Road Map of PIQoS

• Background
• PIQoS architecture
• Experimental setup
• Model evaluation data
• Discussion on results
• Conclusions
Software Defined Networking (SDN)

SDN decouples **control plane** (control logic) from **data plane** (forwarding hardware)

Traditional networking

Software-Defined networking
SDN Architecture

SDN Logical Layers and Architecture (IETF RFC 7426)

- **Application Layer**
  - Monitoring
  - Data Center
  - Mobility and Wireless
  - Security
  - Traffic Engineering

- **Control Layer**
  - Centralized Controllers
  - Distributed Controllers
  - Northbound API
  - Southbound API

- **Infrastructure Layer**
  - Network Devices (Switches, Routers, etc)

- **Applications, Orchestrations**
  - e.g., Path computation
  - Loop avoidance
  - REST API (works with http)
  - NOX, ONOS, ODL, ...
  - OpenFlow, NetConf, ...
  - Switch, Router, ...

REST API (works with http)
Supervised vs. Unsupervised Machine Learning Algorithms

- Supervised algorithms use labeled data sets to create models, while unsupervised learning uses unlabeled data.
- Each of these algorithms is better suited for a particular type of application:
  - classification, regression or clustering
- Different learning algorithms can be used to solve the same problem.
- The best choice depends on the requirements of each scenario:
  - performance and accuracy constraints

Source: https://www.researchgate.net/publication/329533120_Background_Augmentation_Generative_Adversarial_Networks_BAGANs_Effective_Data_Generation_Based_on_GAN-Augmented_3D_Synthesizing
Failure Recovery Scheme: Restoration and Protection

**Restoration**

- Flow Table: H1--H5--2
- Flow Table: H1--H5--1
- Group Table: G1:2: 4
- Group Table: G1:2: 3

**Protection**

- Flow Table: H1--H5--G1
- Group Table: G1:2: 4
- Flow Table: H1--H5--G1
- Group Table: G1:2: 3
PIQoS Architecture

Application Layer
- Network Management
- Routing
- Orchestration

Control Plane
- Policy Prediction
- Error Prediction

Data Plane
- OpenFlow 1.3
- OVS

Workflow:
1. Start
2. Monitor Traffic
3. Error Prediction
4. Alarms DB
5. Policy Prediction
6. Predicted Policy DB
7. Network Management
Experimental Setup

**Environment:**
1) Emulator: Mininet
2) Data plane elements: Open vSwitch
3) Controller: Ryu
4) Scikit and Weka

**Topologies:**
1) Ten switch Mininet topology
2) USNET topology

**Performance Metrics:**
1) Failure recovery time
2) Throughput
3) Prediction accuracy
### The list of features used in PIQoS

<table>
<thead>
<tr>
<th>Feature</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time stamp</td>
<td>When a flow starts at a link</td>
</tr>
<tr>
<td>Flow ID</td>
<td>ID of a flow or a path</td>
</tr>
<tr>
<td>Source Node</td>
<td>Source node of a link in a flow</td>
</tr>
<tr>
<td>Destination Node</td>
<td>Destination node of a link in a flow</td>
</tr>
<tr>
<td>Source Outport</td>
<td>Port from which a source node sends packets</td>
</tr>
<tr>
<td>Destination Inport</td>
<td>Port from which a destination node receive packets</td>
</tr>
<tr>
<td>Packets</td>
<td>Total number of packets sent in a link</td>
</tr>
<tr>
<td>Bytes</td>
<td>Number of bytes sent in a link</td>
</tr>
<tr>
<td>Linkspeed (B/Sec)</td>
<td>Speed of a link</td>
</tr>
<tr>
<td>Source Output Capacity</td>
<td>Maximum capacity of a source output</td>
</tr>
<tr>
<td>Destination Inport Capacity</td>
<td>Maximum capacity of a destination import</td>
</tr>
<tr>
<td>Source port Speed (B/s)</td>
<td>Speed at which a source outport sends packets</td>
</tr>
<tr>
<td>Destination port Speed (B/s)</td>
<td>Speed at which a destination inport receives packets</td>
</tr>
<tr>
<td>Delay (ms)</td>
<td>Time taken for a packet to be transferred from a source to a destination in a flow</td>
</tr>
<tr>
<td>Error Prediction Classification</td>
<td>A flow has an issue or not (error/no error)</td>
</tr>
<tr>
<td>Policy Prediction Classification</td>
<td>Classification type (link failure/congestion)</td>
</tr>
</tbody>
</table>

**The number of flows and links used for training and testing models**

<table>
<thead>
<tr>
<th>Model</th>
<th>Total flows</th>
<th>Total links</th>
<th>Training flows</th>
<th>Training links</th>
<th>Testing flows</th>
<th>Testing links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error prediction</td>
<td>318</td>
<td>1020</td>
<td>225</td>
<td>714</td>
<td>93</td>
<td>306</td>
</tr>
<tr>
<td>Policy prediction</td>
<td>151</td>
<td>465</td>
<td>106</td>
<td>326</td>
<td>45</td>
<td>139</td>
</tr>
</tbody>
</table>
Resiliency of PIQoS

- PolicyCop: throughput drops and delay increases as it needs to contact the controller
- PIQoS: local recovery improves both the throughput and delay
- Local recovery can furthermore offer scalability
• Naive Bayes and SVM have the worst accuracy.
• Random Forest and Decision Tree have the best accuracy.
• Decision Tree is less complex.
• **Decision tree is the winner.**
Intelligence of PIQoS (unsupervised algorithms)

- Supervised algorithms need label on data, which may not scale with large networks.
- K-means offers best accuracy for two clusters.
- This accuracy is quite low compared to that of DT.
- There is a trade-off between accuracy and overhead.
Conclusions

• We proposed a programmable and intelligent QoS framework, called PIQoS.
• PIQoS pushes link failure recovery at the data plane to improve the delay and throughput.
• We proposed two learning models for efficient network state diagnosis and selecting corresponding management policies.
• Implemented the proposed framework in a realistic simulation environment.
• **PIQoS can accurately predict link failure or network congestion and update policy accordingly.**
Thank you 😊
israat@dal.ca
Quality of Service (QoS)

• An important factor for service providers
  • To offer consistent user experience
• Emerging service requirements, network traffic and their dynamics pose a new challenge
• Automated error detection and root cause analysis can lead to an efficient QoS provisioning

• How can we design a programmable and intelligent QoS framework?
Decision Tree and K-means

**Decision Tree**

```
Attribute with Highest Information Gain

B: Decision 1
A: Attribute with Highest Information Gain given A

D: Decision 1

E: Decision 2

F: Attribute with highest Information Gain given C

C: Decision 1

G: Decision 2
```

**K-means**

Before K-Means

After K-Means


Decision Tree and K-means

**Decision tree:**
- Data is stored in a tree-like structure a node, branch, and leaf-node represent test, decision, and class label, respectively.
- A decision tree path defines the data classification rules.
- An attribute that best classify training data is chosen as the root and repeat the same process at each branch until all attributes are covered.

**K-means:**
- Partitions data points into clusters.
- Each cluster of data points has a centroid defined by those data points.
- At the beginning of the clustering, we can randomly choose K centroids and iteratively optimize their positions.