Towards Chain-Aware Scaling Detection in NFV with Reinforcement Learning

Lin He, Lishan Li, Ying Liu
Network Function Virtualization

• **Introduce custom packet processing functions into the network**  
  • offer the potential to enhance service delivery flexibility and reduce overall operational expenses  
  • enable elastic scaling by creating and destroying VNF instances

• **Primary goals of elastic scaling**  
  • satisfy service level agreements (SLAs)  
  • minimize VNF operating cost
Existing Solutions

• Rate-based
  • Estimate the upcoming traffic rate and then compute the number of required instances that can process the estimated traffic demand.
  • Examples: Wang et al. [Cloud’16], SLFL [CloudNet’15], Zhang et al. [INFOCOM’17], VPCM [INFOCOM’18], Tang et al. [TPDS’19]
  • The dynamics of upcoming traffic in packet size and type affect instance number computation.

• Status-based
  • Achieve scaling detection based on VNFs’ runtime status, including the application- and hardware-level parameters.
  • Examples: ENVI [ANCS’18]
  • Affected by the collected “raw” status information, which causes imprecise scaling decisions.
The Problem

**Existing Solutions:** Designed based on a simplified or inaccurate understanding of deployment environments.

**Problem:** system environment affects the scaling mechanism

**Challenge:** How to adjust the scaling strategy and parameters in real time with system changes?
Reinforcement Learning

Environment

Agent

Solution: Chain-Aware Scaling Detection (CASD)
Talk Outline

- Motivation
- \textit{CASD Architecture}
- Evaluation Results
- Conclusion
CASD Architecture

CASD Agent

Scaling Detection Neural Network

Chain-Aware Representation Generation

Scaling Action

VNFs’ features: runtime status, traffic rate

NFV System

Controller

Data Plane

VNFs: VNF₁, ..., VNF₄
Talk Outline

• Motivation
• CASD Architecture
  • Chain-aware Representation Generation
  • Scaling Detection Model
• Evaluation Results
• Conclusion
Chain-aware Representation Generation

- **VNF Representation**
  - Initial state: input, output, latency, cpu, memory
  - Not only capture its explicit state but also depict the effects of its children in the chain

- **Global Chain Representation**
  - regard the chain as a particular VNF summary node

- **Chain-aware Representation**
  - VNF Representation + Global Chain Representation
Scaling Detection Model

• **Neural Network Model**
  - Input: Chain-aware representation sequence
  - GRU: Capturing relationship of sequences

• **Training Method: A3C**
  - Actor Network: Obtain the probability distribution of scaling actions
  - Critic Network: Measure how well the policy performs
Talk Outline

• Motivation
• CASD Architecture
• Evaluation Results
• Conclusion
Implementation

- **CASD prototype**
  - OpenNetVM and DPDK
  - TensorFlow
  - Controller loading training model

- **Two Chains**

- **Two types of traffic patterns**
  - Moderate Increase
  - Sharp Increase

![Diagram of Chain A and B with IDS, Monitor, and VPN nodes with rules for dst.port conditions]
Implementation

• **Status-NN**
  - Trained with RL
  - Online status
  - ENVI

• **Composite-NN**
  - Trained with RL
  - Online status + traffic rate

• **Composite-DT**
  - Decision tree
  - Online status + traffic rate
Evaluation

- **Overall Cost** = Packet Loss + Instance Cost (Too Early) + Instance Cost (Too Late)

<table>
<thead>
<tr>
<th>Packet Loss</th>
<th>Instance Cost</th>
<th>Overall Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>0.0</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>0.0</td>
<td>0.3</td>
<td>0.4</td>
</tr>
<tr>
<td>0.0</td>
<td>0.4</td>
<td>0.5</td>
</tr>
</tbody>
</table>

- Chain A Diagram:
  - IDS
  - Monitor
  - Cache
  - dst.port == 80
  - dst.port != 80
  - VPN

Bar charts showing normalized average cost for different methods:

- Status-NN
- Composite-DT
- Composite-NN
- CASD
Evaluation

• **Overall Cost** =
  
  Packet Loss\(\uparrow\) + Instance Cost\(\uparrow\)

  (Too Early)\hspace{1cm} (Too Late)

---

Chain B

- **Cache**
- **Monitor**
- **VPN**

- **IDS**
  
  \(\text{dst.port} = 80\)
  
  \(\text{dst.port} \neq 80\)

---

### Normalized Average Cost

- **Status-NN**
- **Composite-NN**
- **Composite-DT**
- **CASD**
Evaluation

- **CASD Working Process**
  - Dynamic change in traffic rate → Add/remove instances
Talk Outline

• Motivation
• CASD Architecture
• Evaluation Results
• Conclusion
Conclusion

- **We present CASD which utilizes reinforcement learning and neural networks to automatically learn scaling detection policies without any human instructions.**
  - To further improve agility and system performance, CASD incorporates global chain information into control policies to efficiently plan the scaling sequence of VNFs within the chain.
  - To build CASD, we develop scalable representations for VNFs and global chain, design neural networks based on feature sequence, and utilize the A3C algorithm for model training.
  - We have implemented a prototype on top of the NFV system and compare it with multiple baseline algorithms over different traffic patterns and chains.
  - Evaluation results show that CASD outperforms the state of the arts in terms of overall system cost and packet processing rate.
Thanks!