REAL-TIME LARGE-SCALE DATA PROCESSING
USING HETEROGENEOUS NETWORK COMPUTER ENVIRONMENT
- An Application Platform using SDN Technology -

Tatsuya FUJII
Senior Research Engineer/Supervisor, NTT Network Innovation Labs.

Part of this work is supported by SCOPE (150100004) of Ministry of Internal Affairs and Communications, Japan
Real-time Large-scale Data Processing using Heterogeneous Network Computer Environment

CONTENTS

• Background & our research goal
• Our proposal
• Experiments
• Conclusion
Background

• Need for cloud base high-performance computing environment using multiple VMs
  • Super HD video processing, 10G/100G Network log analysis, etc.
  -- Customers; “Can we use clouds for large data processing?” , because they prefer
    • Low cost. (especially initial and maintenance cost.)
    • On-demand usage.

• Use of idling (non-working) VMs all around the Internet on a temporary basis.

• However…
  -- Difficulty to configure a virtual high-performance computer with VMs using existing technologies.
    • No precise performance estimation tool. “When the task of data processing finishes?”
    • CPUs, memories, and networks are virtual. Performance depends on where VMs are installed.
  -- Need for new parallelizing method for heterogeneous computing environment w/o pre-configuration.
Our research goal – “Virtual Cloud”

- Realize an application platform for low-cost on-demand large-scale network computer.
  - By dynamic configuration of idle VMs and virtual NWs for parallel data processing.
  - By continuous performance monitoring of VM and network with load-balancing control.
System Model for Parallel Data Processing

- “Source-process-cache” model is introduced to simplify system structure, shown below.
- Divide complex calculation into multiple simple calculations.
- Propose optimization methods for the simple calculation.

Original calculation

Expanded calculation

Parallelize and optimize
Basic approach for Parallelization and Optimization

Step 1. Data in source is divided into number of fragments.
Step 2. Divided data is transferred to VMs dynamically assigned by monitored VM status.
Step 3. VMs calculate and transfer results to cache.
Implementation to VMs and virtual NW systems

Management node continuously monitors VMs’ performance, NW status, etc. All data sets are sent to a selected VM via OpenFlow switches.
Dynamic load balancing technique using OpenFlow switch

“Data Source Node” does NOT take care of selection of VMs. Just transmit to OF switches. OF switch executes IP address translation depending on which VM is selected.
Application example: Video composing

Synchronization of OF switches is needed, done by IP packet header inspection.
Monitoring System

• Original monitoring program is developed and implemented to all VMs.

• Each VM sends monitoring data to management node via Fluentd every one second.
  – CPU utilization, network utilization, application processing time and so on.

• Management node received monitored data and stores into REDIS data base in monitoring server.
  – Very fast response is needed to control OF switches to avoid TCP SYN response time-out.
  – High speed on-memory DBMS in management node.
Protocols

• REST (Representational State Transfer) based protocol

• Merits
  – “Source node” and “Cache node” are placed inside “Firewall” (or NAT router)
    • TCP should be adopted for transport (not UDP) because of its transparency
  – “Source node” is implemented easily with CURL OSS software
  – Enables to send data with message over single TCP connection

• Demerits
  – Requires slightly more complicated flow-control in OpenFlow switch than UDP
Software structure - Implementation

- Platform software design so that many kind of application other than video composing can be easily implemented.
- Simple API design.
  - Just copy data into shared memory with defined format.
Experiment in SC15

- **Objective:** Pre-test using world wide distributed VMs.
- **Application:** Two sets of uncompressed 4K video data in video servers (source nodes) were transferred and composed by distributed VMs frame-by-frame. After that, the composed video was sent to cache node (cache server) to form a movie.
Experiment in SC15 cont.

- 4K display
- Composed 4K video
- Real-time resource monitor
Field Test in Interop Tokyo 2016

- Objective: Confirm our system (video composing application) works well
- Application: Uncompressed 4K real-time composing
- Real-time processing 4K60P video using 144 VMs. (Each VM has <10% power of CPU.)
Conclusion

• New technique for large-scale could data processing.

• Low-cost, on-demand processing utilizing idle VMs and NW resource in the internet.
  – Parallelization, Dynamic load balancing by OpenFlow switch, VM monitoring system.

• Software development for parallel data processing and resource monitoring.

• Implementation to real clouds and IP networks
  – SC15 and Interop Tokyo 2016, real-time processing of 4K SHD videos
Acknowledgement: Special thanks
WE NEED YOU!

**Lagopus:**
High-performance SDN/OpenFlow Software Switch BoF
@Room Merrick I (next to this session)
- http://lagopus.github.io/
- Let’s discuss
  - Software switch technology for performance and scalability
  - Experiments with SDN-IX and NFV middleware
  - Future collaboration on Internet2

**Lagopus Live Demonstration**
@ NTT Demo Booth
- Copycast of 4K Uncompressed Video Streaming
- Benchmark test of high-performance IP software router
REAL-TIME LARGE-SCALE DATA PROCESSING USING HETEROGENEOUS NETWORK COMPUTER ENVIRONMENT

- An Application Platform using SDN Technology -

Tatsuya FUJII

Senior Research Engineer/Supervisor, NTT Network Innovation Labs.

Part of this work is supported by SCOPE (150100004) of Ministry of Internal Affairs and Communications, Japan