Dynamic Controller Provisioning in Software Defined Networks

Presented By

Md. Faizul Bari
David R. Cheriton School of Computer science
University of Waterloo

with Arup Raton Roy, Shihabur Rahman Chowdhury, Qi Zhang, Mohamed Faten Zhani, Reaz Ahmed, and Raouf Boutaba,
Outline

- Overview of SDN
- Motivation
- Proposed management framework
- Problem formulation
- Proposed heuristics
- Simulation results
- Conclusion & future work
Traditional vs. Software Defined Networking

Distributed Control

Centralized Control

Traditional Networking

Software Defined Networking

OpenFlow
SDN and OpenFlow

Flow setup time: time needed to set up the rules associated with a new flow in switches involved in the routing path.
Outline

• Overview of SDN
• Motivation
• Proposed management framework
• Problem formulation
• Proposed heuristics
• Simulation results
• Conclusion & future work
Traditional SDN with a Single Controller

- A single controller controls all switches in the network
- Advantages:
  - Centralized control
  - Ease of management
  - Network-wide view
- Disadvantages:
  - High switch-to-controller latency
  - Limited processing capacity of controller
  → Higher flow setup time
Multiple Controller

- Each controller controls a subset of the switches
- A switch communicates with just one controller
- **Advantages:**
  - Less processing capacity is required for each controller
  - Lower switch-to-controller latency
- **Disadvantages:**
  - Require state synchronization between controllers
    → Large control traffic overhead
  - Static switch-to-controller assignment
    → Overloaded controllers
Dynamic Controller Provisioning Problem (DCPP)

- Dynamically provision controllers based on
  - Changing network conditions (traffic dynamics)
  - Switch-to-controller latency requirement
- Goals
  - Dynamically decide the number of controllers and their locations
  - Minimize flow setup time and control traffic
  - Minimize switch-to-controller reassignments
Outline

• Overview of SDN
• Motivation
• Proposed management framework
• Problem formulation
• Proposed heuristics
• Simulation results
• Conclusion & future work
Management Framework

- **Monitoring Module**
  - Monitors controllers and collects statistics about the traffic

- **Reassignment Module**
  - Decides the number of controllers, their locations and the switch-to-controller assignment based on network conditions

- **Provisioning Module**
  - Provisions controllers and assigns switches to them
Outline

• Overview of SDN
• Motivation
• Proposed management framework
• Problem formulation
• Proposed heuristics
• Simulation results
• Conclusion & future work
Problem Formulation

- DCPP can be formulated as an ILP
- Objective function
  \[ \text{Minimize } \alpha C_l + \beta C_p + \gamma C_s + \lambda C_r \]
- Where
  - \( C_l \) = Statistics collection cost
  - \( C_p \) = Flow setup cost
  - \( C_s \) = Synchronization cost
  - \( C_r \) = Switch reassignment cost
- Constraints
  - Controller capacity constraint
  - Switch-to-controller delay constraint

DCPP is NP-hard.
Outline

• Overview of SDN
• Motivation
• Proposed management framework
• Problem formulation
• Proposed heuristics
• Simulation results
• Conclusion & future work
Proposed Heuristics

- We propose two heuristics to solve DCPP
  - Greedy Knapsack based (DCP-GK)
  - Simulated Annealing based (DCP-SA)

- DCP-GK provides quick but inferior solutions
- DCP-SA provides good solutions, but requires longer time to find solutions
Greedy Knapsack Based (DCP-GK)

- Each controller is modeled as a knapsack
  - Capacity of the knapsack = number of flow-setups/sec
- Each switch is an object to be included in a knapsack
  - Weight = number of flow setup requests/sec
  - Profit = Inverse of switch to current controller’s distance

Procedure
1. Repeat the following steps until either all switches are assigned to a controller or the controller set is exhausts
   - Step 1: Pick the controller with minimum total distance from all switches
   - Step 2: Use Greedy Knapsack approach to assign unassigned switches to the controller (subject to delay constraint)
2. Randomly assign the remaining switches
Simulated Annealing Based (DCP-SA)

• DCPP is solved in two phases:
  • Phase 1: find a feasible assignment from the current one
    • For each overloaded controller
      • Select the switch sending maximum requests to it
      • Assign the switch to the most underused controller if the delay and capacity constraint are satisfied
      • Otherwise provision a new controller
    • Repeat until capacity and delay constraints are satisfied for all controllers
  • Phase 2: optimize the assignment by local search moves
    • Relocate switch
    • Swap switches
    • Activate a new controller
    • Merge controllers
Outline

• Overview of SDN
• Motivation
• Proposed management framework
• Problem formulation
• Proposed heuristics
• Simulation results
• Conclusion & future work
Simulation Results

- We consider 3 scenarios
  - 1-CTRL: A single controller for all switches
  - N-CTRL: One controller for each switch
  - DCP: Dynamic controller provisioning

- Topology
  - 108 nodes, 306 links (from RocketFuel [1])

- Traffic
  - Based on a realistic traffic trace [2]
  - End-to-end TCP flows

Flow-setup Time CDF

- N-CTRL provides minimal flow-setup time
- DCP-GK and DCP-SA both are better than 1-CTRL
- DCP-SA performs better than DCP-GK
Number of Controllers and Flow-setup Time

- DCP-SA required less controllers than DCP-GK
- In case of 1-CTRL flow-setup time varies with traffic
- DCP-GK and DCP-SA adapt to traffic changes
Summary of Overhead and Flow Setup Time

- N-CTRL has lowest flow-setup time, but largest overhead
- 1-CTRL has lowest overhead, but highest flow-setup time
- DCP-SA performs much better than DCP-GK
Outline

• Overview of SDN
• Motivation
• Proposed management framework
• Problem formulation
• Proposed heuristics
• Simulation results
• Conclusion & future work
Conclusion

- We identified the Dynamic Controller Provisioning Problem (DCPP)
- Proposed a management framework for dynamically deploying multiple controllers
- Provided a mathematical formulation of DCPP as an ILP
- Proposed two heuristic algorithms to solve DCPP
- Identified the trade-off between flow-setup time and communication overhead
Future Work

• Improve the convergence time of DCP-SA
  • Generate quick solutions using DCP-GK and then optimizing them using DCP-SA.
• Explore other heuristic algorithms
• Perform experiments on a real testbed
  • Distributed OpenFlow Testbed (DOT) [dothub.org]
Questions?
Background Slides
Path Setup Model

Initial path setup request

Intermediate path setup request

Rule Setup

Forwarding Path

Controller

Switch
Single-source Unsplittable Flow Problem

- Directed graph $G(V,E)$
- Capacity on edges $c : E \rightarrow \mathbb{R}^+$
Single-source Unsplittable Flow Problem

- A single source $s$ and $k$ terminals $t_i$ with demands $d_i \in \mathbb{R}^+$
- A vertex may contain an arbitrary number of terminals
Single-source Unsplittable Flow Problem

• Each commodity flows along a single path from s to $t_i$ (unsplittable)
• Must satisfy edge capacity
• There are many variations of this problem
• In our case, we minimize the sum of edge weights
DCPP $\rightarrow$ SSUFP

- Controller capacity used as edge capacity
- Flow-setup & stat. collection costs are used as edge weights
- Edges are allowed only when they satisfy latency constraint