3U-03

OpenFlow における経路切り替えコストを考慮したネットワーク制御手法の一検討

後谷 浩輔 †1　高平 寛之 †1　畑 美純 †1　Guillen Luis†1
和泉　諭 †1　阿部 亨 †1,†2　菅沼 拓夫 †1,†2

†1 東北大学大学院情報科学研究科　†2 東北大学サイバーサイエンスセンター

1　はじめに

クラウドの普及に伴い，データセンタ内のサーバやスイッチの数が増加し，フローの経路制御やトポロジの管理が煩雑化している．この問題への対処として，コントローラによりネットワーク内のスイッチを集中制御することで，柔軟な運用を実現する OpenFlow が注目されている．OpenFlow では，障害の復旧やボトルネックの解消のためにフローの経路を切り替える際，コントローラがフローの経路情報を記述したフローエントリをスイッチのフローテーブルに追加・変更・削除する処理を行う．

しかし，これらの処理による経路切り替えには様々なコストがかかる．例えば，フローエントリの追加や変更の処理には時間がかかり，それが通信の途絶や遅延の原因になる可能性がある．本研究では，経路切り替えにかかるコストを考慮した，高速に経路を切り替えるネットワーク制御手法を提案する．本稿では，機種によってスイッチにフローエントリを追加する処理時間が異なる点に着目し，スイッチの処理時間を考慮して経路切り替えにかかる時間を最小化する問題として定式化を試みる．

2　関連研究と課題

OpenFlow における経路切り替えの高速化を目的とした研究としては，特定の経路のみの切り替えに着目し，経路切り替えを高速化する手法を提案した文献 [2] [3] が挙げられる．文献 [2] では，経路を切り替える際にフローエントリの追加数に上限を設けて経路を選択することにより，経路切り替えの高速化を図っている．文献 [3] は，切り替え前の経路を部分的に再利用することにより，フローエントリの追加数を削減することで高速化を図っている．

しかしこれらは，多様な機種のスイッチが混在する状況を考慮していない．企業やその他の機関では，予算の制約や購入時期の差異などにより，多様な機種のスイッチを組み合わせてネットワークを構築する場合も多い．このような様々な機種のスイッチが混在する状況では，機種によってフローエントリの処理時間が異なることが報告されている [1]．そのため，処理が低速なスイッチに多数のフローエントリを追加すると，経路切り替えに時間がかかり，通信の途絶や遅延を引き起こすなどの課題がある．したがって，それら特性を考慮したネットワーク制御手法が必要である．

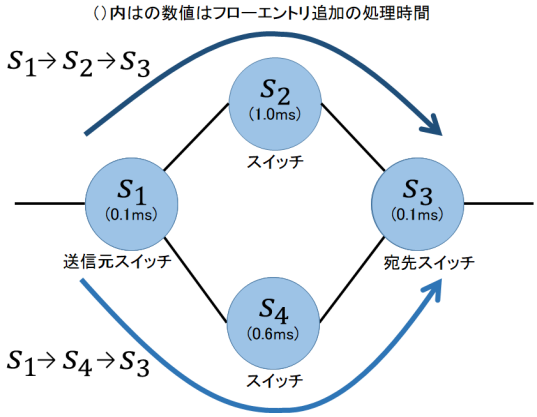

図 1: 提案手法を適用する問題の例

3　経路切り替えコストを考慮したネットワーク制御手法の提案

3.1　概要

本研究では各スイッチにおけるフローエントリの追加にかかる処理時間を考慮したネットワーク制御手法を提案する．提案手法の例として，図 1 のような，フローエントリの追加にかかる処理がそれぞれ異なるスイッチ s_1 から s_4 が混在するネットワークを用いて説明する．ここでは，s_1 から s_3 への通信を中継しているネットワークで，s_1 から s_3 の間の経路を切り替える場合を考える．この場合，切り替え後の経路の候補としては $s_1 \rightarrow s_2 \rightarrow s_3$ と $s_1 \rightarrow s_4 \rightarrow s_3$ の 2 つが挙げられる．

この時，経路切り替え時間は，経路中に含まれるスイッチにおいて，フローエントリの追加にかかる処理時間が最も長い値となる．したがって，図 1 における経路切り替えの例では，経路 $s_1 \rightarrow s_2 \rightarrow s_3$ への切り替え時間は 1.0ms，経路 $s_1 \rightarrow s_4 \rightarrow s_3$ への切り替え時間は 0.6ms となるため，提案手法では，経路切り替え時間が小さくなる経路 $s_1 \rightarrow s_4 \rightarrow s_3$ を選択する．

A Study on Network Control Method Considering Route Switching Cost for OpenFlow
Kosuke GOTANI†1, Hiroyuki TAKAHIRA†1, Misumi HATA†1, Luis GUILLEN†1, Satoru IZUMI†1, Toru ABE†1,†2, and Takuo SUGANUMA†1,†2
†1Graduate School of Information Sciences, Tohoku University
†2Cyberscience Center, Tohoku University

3.2 経路切り替え時間の最小化問題の定式化

本節では提案手法の設計として，送信元と宛先のペアが複数存在し，さらにその間の経路も複数切り替える状況を想定した，経路切り替え時間の最小化問題を定式化する．ここでは，スイッチを $s_i \in S(i = 1, 2, ...)$, スイッチ s_i と s_j 間のリンクを $e_{i,j} \in E$，切り替える経路数を n とし，フローを識別する $k(k = 1, 2, ..., n)$ を用いて，s_s^k(切り替え後のフローの経路の送信元スイッチ) から s_t^k(切り替え後のフローの経路の宛先スイッチ) への経路を p^k とし，フローの経路 p^k の集合を $P^m(m = 1, 2, ...)$ ($P^m = \{p^1, p^2, ..., p^n\}$)，$P^m \in P_{all}$ となる P^m の集合を P_{all} とする．なお，p^k は切り替え後の経路の送信元から宛先までのスイッチのリスト ($p^k = (s_s^k, ..., s_t^k)$) である．

本研究では経路切り替え時間に関する目的関数を定義し，フローの経路 p^k の組み合わせ方，すなわち，目的関数が最小となる経路の集合 P^m を求めることで提案手法を実現する．本稿で定義した経路切り替え時間を最小化する問題の目的関数を式 (1) に示す．ここでは，スイッチのフローエントリ 1 つの追加にかかる処理時間を c_i とし，切り替え後に p^k のリストが s_i を含むかを示す変数を z_i^k（式 (2)）としている．

$$\min_{\{P^m \in P_{all}\}} \{ \max_{\{i|s_i \in S\}} \{c_i \times \sum_{k|p^k \in P^m} z_i^k\}\} \quad (1)$$

$$z_i^k = \begin{cases} 1 & (p^k \text{が } s_i \text{を含む場合}) \\ 0 & (\text{その他}) \end{cases} \quad (2)$$

なお，経路を切り替える場合，切り替え前のフローエントリを削除せず，新たなフローエントリを高いプライオリティで追加することで，新たな経路への切り替えが可能である．したがって，ここでは経路を切り替える際には，切り替え前のフローの経路は考慮しない．

3.3 制約条件

本研究では，フローの経路は枝分かれしないことを条件とする．そのため，フローの経路 p^k が枝分かれしない制約条件を定義する．具体的には，p^k が枝分かれしない経路である場合，各フローの経路 p^k において，転送するフローの数と自身に転送されるフローの数の差がそれぞれ，フローの送信元スイッチでは 1，フローの宛先スイッチでは -1，それ以外では 0 となる．したがって，p^k の制約条件は，フローの経路 p^k がリンク e_{ij} に i から j にフローが通過しているかを示す変数 x_{ij}^k（式 (4)）を用いて，式 (3) のように定義する．

$$\forall k, i \sum_{j|e_{ij} \in E} x_{ij}^k - \sum_{j|e_{ij} \in E} x_{ji}^k = \begin{cases} 1 & (s_i = s_s^k) \\ -1 & (s_i = s_t^k) \\ 0 & (\text{その他}) \end{cases} \quad (3)$$

表 1: 予備実験に用いたスイッチの基本性能

スイッチの性能	機種 A	機種 B
スイッチング容量	176Gbps	176Gbps
転送性能	132Mpps	131Mpps
最大フローエントリ数	12,000	160,000

$$x_{ij}^k = \begin{cases} 1 & (p^k \text{が } e_{ij} \text{を } s_i \text{から } s_j \text{に通過する場合}) \\ 0 & (\text{その他}) \end{cases} \quad (4)$$

4 予備実験

スイッチの機種によってフローエントリの追加にかかる処理時間は異なることを確認するために，2 つの機種の OpenFlow スイッチ（機種 A，機種 B）と OpenFlow コントローラ（OpenDaylight）を用いて，フローエントリ 1 つの追加にかかる処理時間を測定した．実験に用いた 2 つのスイッチの基本性能を表 1 に示す．実験は，コントローラとスイッチを直接接続し，コントローラ側の NIC をキャプチャすることによって，コントローラが制御パケットを送り始めてから，スイッチからの処理の完了を通知する確認応答が返ってくるまでの時間を 100 回測定した．

測定の結果，機種 A の処理時間の平均は 0.83ms，機種 B は 2.8ms となった．この測定結果から，フローエントリ 1 つの追加にかかる処理時間は，機種によって異なることを確認した．

5 おわりに

本稿では，多様な機種が混在する環境における経路切り替え時間を考慮したネットワーク制御手法を提案し，その設計として，経路切り替え時間を最小化する問題を定式化した．

今後は，予備実験の結果や設計した内容を基に定式化した問題を解く方法を検討する．さらに，実ネットワークに適用して，その有効性を検証する．

参考文献

[1] Nguyen-Ngoc, Anh., et al.: Performance evaluation mechanisms for FlowMod message processing in OpenFlow switches, *Proc. ICCE2016*, pp.40-45, 2016.

[2] Astaneh, S. A., et al.: Optimization of SDN Flow Operations in Multi-Failure Restoration Scenarios, *IEEE Transactions on Network and Service Management*, vol.13, no.3, pp.421-432, 2016.

[3] Malik, A., et al.: Optimisation Methods for Fast Restoration of Software-Defined Networks, *IEEE Access*, vol.5, pp.16111-16123, 2017.