論　文

災害時におけるSNS情報を活用したプログラマブルなQoS制御システムの検討

柳田　晴香[†a)]　中尾　彰宏[††]　山本　周[††]　山口　実靖[†††]
小口　正人[†]

Investigation of Programmable QoS Control Using SNS Information in Disaster Situation

Haruka YANAGIDA[†a)], Akihiro NAKAO[††], Shu YAMAMOTO[††], Saneyasu YAMAGUCHI[†††], and Masato OGUCHI[†]

あらまし　東日本大震災時，電話やインターネットが使えない事態が生じたことから，大規模災害時には，従来のネットワーク機器監視による制御だけではなく新たな情報を利用した制御が期待されている．我々は，多くの人間の目や口コミによるネットワーク障害に対する集合知が災害の状況把握に有益であるという仮説に基づいて，SNS 情報に基づいたネットワーク制御システムを提案している．従来の機器監視制御には，制御プレーンをプログラム可能とする SDN(Software Defined Networking) による柔軟な集中制御が期待されている．しかし現在の SDN では，依然として，ヘッダ情報に基づく経路制御など従来と同様な制御の効率化に限定される．そこで我々は，データプレーン処理とその API(Southbound Interface) をプログラム可能とする SDN を拡張する DPN (Deeply Programmable Network) の概念に基づいて，SNS の情報が含まれるパケットペイロードのデータを監視することで制御する手法を提案する．本論文では，FLARE を用いて，OpenFlow の制御処理やアプリケーションの種類に基づいたトラフィックの分類を実装し，アプリケーション毎の QoS 制御を検討する．

キーワード　SNS, SDN, OpenFlow, DPN, FLARE

1. はじめに

2011 年 3 月に発生した東日本大震災において，一部ネットワークが断絶し，電話やメールなどの通信手段が利用できない事態が発生した．この際，復旧に時間が掛かった要因として以下の 2 点が挙げられた．経路切り替え等の設定が一部手作業であったこと，また，ネットワーク機器からの監視アラート情報が膨大になり，どこが深刻なダメージを受けているのか迅速に把握できなかったことである [1].

† お茶の水女子大学，東京都
Ochanomizu University, 2-1-1 Otsuka, Bunkyou-ku, Tokyo, 112-8610 JAPAN
†† 東京大学，東京都
University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8654 JAPAN
††† 工学院大学，東京都
Kogakuin University, 1-24-2 Nishi-Shinjuku, Shinjuku-ku, Tokyo, 163-8677 JAPAN
a) E-mail: g1120541@is.ocha.ac.jp

これらの背景に対し，本研究では，大きく 2 つのアプローチをとる．第一に，従来のネットワーク機器監視に加え，多くの人の目に依る SNS 情報を利用することで，大震災のような大規模な災害時にネットワーク状況を迅速に把握することである．我々は主要な SNS のひとつである Twitter [2] からリアルタイムにツイートを解析することで，高い正確性でネットワーク通信障害を迅速に検知するシステムを構築した [3]. この手法により，大規模な災害時に，ネットワーク全体でどこが深刻なダメージを受けているのか速やかに特定することができる．

第二に，ネットワーク経路の柔軟な自動集中制御である．災害時において経路切替を自動で柔軟に制御することは重要である．近年では制御プレーンをプログラム可能とする SDN(Software Defined Networking) や OpenFlow [4] による柔軟な制御が期待されている．しかし現在の SDN では，依然として，ヘッダー情報に基づく SPI(Stateful Packet Inspection) 制御な

ど従来と同様な制御の効率化に限定され，DPI(Deep Packet Inspection) のように IP パケットのデータペイロードの情報を基にフィルタリングなどの処理方法を決める，高度なパケット処理は考えられていない．インターネットの利用が一般化し，その上で動くサービスやアプリケーションが複雑になった現代のネットワークでは，SPI では不十分になってきた [5]．そこで最近，SDN を拡張し，データプレーン処理とその API(Southbound Interface) をプログラマブルにすることで DPI 処理を可能にする，DPN (Deeply Programmable Network) の概念に基づいた，ネットワーク機器の開発がなされている [6] [7] [8]．大規模な災害時には，ネットワークリソースの不足が予測され，特定のアプリケーションの優先制限など，DPI を活用する制御が効果的であると判断できる．よって本研究では，DPN 環境を構築し，より高度で柔軟なネットワーク制御を検討する．

本稿では，OpenFlow の機能とアプリケーションの中身に基づく制御機能の両方を実装可能な FLARE スイッチ [6] [7] を用いて DPN 環境を構築し，災害時に最適なアプリケーション毎の QoS 制御を，SNS による障害検知と合わせて達成する．

本論文の構成は以下の通りである．まず 2. 章で関連研究について述べ，3. 章で SDN と DPN について詳しく説明する．次に，4. 章で提案システムの概要を紹介する．そして，5. 章で実験環境を示し，6. 章で各実験について述べる．最後に，7. 章で本稿をまとめる．

2. 関 連 研 究

SDN や OpenFlow 技術を利用して，災害時に適切なネットワーク制御を自動的に行う障害対策手法が数多く存在する [9] [10] [11] [12] [13] [14] [15]．NTT ドコモら [9] は，OpenFlow 技術を用いて，低優先トラフィックに対する抑制を行うことで通信サービスの優先度に応じた制御を実現している．具体的には，Diff-Serv (Differentiated Services) を用いたクラスベースの CoS 制御で，受信パケットの通信フローとクラスのマーキングを行うことで，相対的に優先度を識別している．しかし，この優先制御は 1 人のユーザが複数の通信サービスを使用する状況下では，正常な識別は可能でない．よって，1 ユーザが複数のアプリケーションを使用する場合でも正常に識別でき，各アプリケーションに対して各々に適切な制御を絶対的に行う本研究手法とは異なる．また，小川ら [10] [11] は，IP アドレスと位置情報をマッピングして DB で管理し，DB と OpenFlow コントローラとの連携をとる手法を提案している．加えて，被災地の通信パケットに対して，IP ヘッダ中の ToS フィールドに災害 ID を付与することで被災地の通信パケットを識別し，QoS を実現する提案をしている．さらに OpenFlow コントローラに，緊急地震速報や気象情報などの外部災害情報を収集する機能を付加する点でも本研究と似ている．しかし本研究は，SNS による集合知を用いる点と，アプリケーションの種類毎に QoS 制御を行うという点で新規性がある．[12] [13] [14] については，無線アドホックネットワークに OpenFlow を適用するもので，ユーザが端末上で評価基準の重みを事前に登録し，それを基にした QoS 制御を提案している．江戸ら [15] は，災害リスクのモデル化を行い，モデル化された災害シナリオを基に経路制御を行っている．しかし，これらの研究はネットワークのトラフィックデータなど内部情報を基に通信制御を行おうとしており，SNS データという外部情報を基に通信制御を行う本研究とは異なる．

加えて，既存の研究はどれも，コントローラによるネットワーク機器の集中管理という SDN の概念に留まっている．すなわちこの場合，DPI などのより高度で柔軟な制御は考えられていない．

3. SDN と DPN

3.1 SDN/OpenFlow による制御

OpenFlow の仕組みは，コントロールプレーンとデータプレーンの機能の分離である．コントロールプレーンとはデータの転送経路を決定する経路制御の頭脳であり，データプレーンはコントロールプレーンの指示に従ってデータを転送する．これら二つの機能は，従来の物理スイッチではどちらも同じ機器内に組み込まれていたが，OpenFlow ではコントロールプレーンをネットワーク機器外部のサーバ上にソフトウェアとして分離し，スイッチではデータ転送機能のみを実行する．OpenFlow プロトコルはこれらを接続する標準的なインタフェースで，インタフェース経由でのデータプレーンの制御を可能にする．

3.2 DPN/FLARE による制御

3.1 節より，SDN ではデータプレーンは，データ転送という固定的な動きしか実行しないことがわかる．このデータプレーンをもプログラム可能にするのが，DPN の概念である．つまり，DPN ではネットワークをフルにプログラム可能にする事を目的とする．

そして DPN を実現するために開発されたアーキテクチャの一つが，FLARE である [6]．FLARE では，データプレーンを Click [16] という言語で実装する．Click の特徴は，「フレームを受け取る」「フレームを転送する」といった様々な基本機能が，モジュールとして用意されている点である．モジュールを組み合わせることで，簡単にスクリプトでネットワーク機器の動作を記述できる．また，独自のモジュールを作成することも可能で，ユニークな動作をするネットワーク機器を作成することが可能である．東京大学では，OpenFlow1.3 スイッチを Click のモジュールで独自に開発しており，FLARE の OpenFlow による操作を可能にしている．

図 1 に FLARE のメカニズムを示す．スリバーと呼ばれるコンテナの中に，Click を用いて作成された，プログラム可能な仮想スイッチが実装される．これらスリバーの集合で，スライスと呼ぶ仮想ネットワーク空間を作成する．

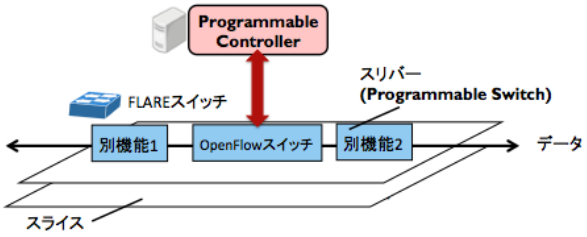

図 1　FLARE の仕組み

4.　提案システムの概要

本研究では，Twitter の解析から得られた障害情報をトリガとして，トラフィックの最適化をアプリケーション毎に自動的・自律的に行う．提案システムの概要を図 2 に示す．

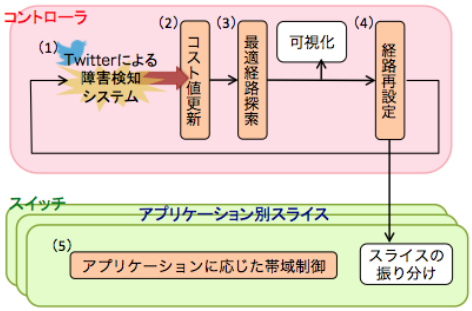

図 2　提案システム概要

まず，ネットワーク障害情報を Twiter のリアルタイム解析システムから受け取り，その情報に従い 60 秒間隔でリンクのコスト値を更新する．コスト値を元にダイクストラ法で最適経路を決定し，スイッチ側へ REST-API で指示を出す．スイッチ側ではアプリケーション別にスライスを用意しておき，スライス毎に帯域制御することで，アプリケーション毎の QoS 制御を達成する．

5.　FLARE 実機環境

5.1　物 理 構 成

本研究では，図 3 に示す物理構成で実機実験を行う．図中で 1G は 1Gbps，10G は 10Gbps のネットワーク接続を示す．FLARE Central は FLARE 管理用のサーバでありスライスの作成等ができる．図 2 に示す提案システムのコントローラ部を，この FLARE Central サーバ上に構築する．このコントローラから，4 台の FLARE スイッチを制御し，様々な制御モデルを検討する．各マシンの仕様は表 1 の通りである．

表 1　Specifications of machines

FLARE switch1 ～ FLARE switch4	Model	Sunwaytech SWS (barebone)
	CPU	Core i7-3612QE Mobile 2.1GHz
	Memory	8GB
	OS	CentOS 6.4
h1～h4	Model	Toshiba dynabook R734
	CPU	Core i5-4210 M 2.6GHz
	Memory	8GB
	HDD	SATA 500GB 5400RPM
	OS	Ubuntu14.04
h5, h6	Model	Dell PowerEdge R220
	CPU	Xeon E3-1241 v3 3.5GHz
	Memory	8GB
	HDD	SATA 1TB 7200RPM
	OS	Ubuntu14.04

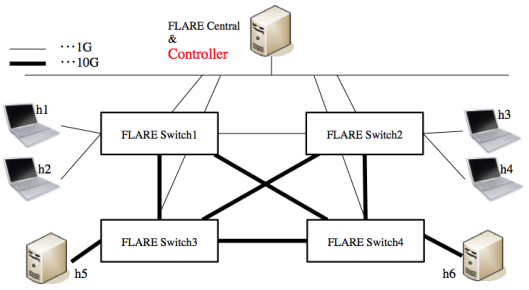

図 3　FLARE 物理構成

6.　FLARE 実機実験

本章では，提案システムの動作確認を経路制御と QoS 制御に分けて行う．経路制御実験では，図 2 の (1)-(4) までの動作を確認し，スループット測定と，遅延時間測定を行う．QoS 制御実験では，(5) のアプリ

ケーションの識別とアプリケーション毎の帯域測定までを行う．

6.1 経路制御実験

図 2 の (1)-(4) について動作を確認する．(5) については，OpenFlow1.3 を実現する Ofswitch モジュールを実装した Click プログラムを動作させる．これにより，FLARE のプログラミング環境の上に OpenFlow を実装した，SDN レベルの実験を行う．本実験では，東日本大震災時の 2011 年 3 月 11 日 14 時から 15 時の実際のツイートを用いる．図 4 のように FLARE スイッチを 1 から順に，岩手，京都，東京，福岡の地域名に対応づけ，通信としては岩手付近の h1 からスタートし，東京付近の h5 をゴールとして設定する．

まず全てのリンク間のコスト値はデフォルトで 1 なので，コスト値最小の Route1 の経路が設定される．コスト値の更新は 60 秒間隔で行われ，Twitter の解析システムより岩手東京間で障害が検知されたことによって，2 回目の 120 秒後の更新の際に岩手と東京間のコスト値が＋ 1 に更新される．その後も 60 秒間隔で岩手東京間で障害が検知され続けたことにより，Route1 の経路はコスト値が＋ 1 され続ける．3 回目の 180 秒後の更新の時点で，岩手と東京間のコスト値が 3 になるため，コスト値最小の 2 である京都を通る経路 Route2 が選択される．ダイクストラ法によって Route2 が選択されると，Route2 に経路を設定する REST-API が呼び出される．これにより，フローエントリがスイッチ側に投げられて，経路が再設定される．本実験により，災害時に Twitter 情報に基づいて，障害地区を迂回する経路切り替えを確認できる．

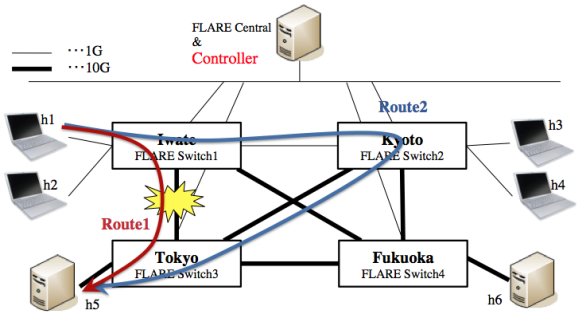

図 4 障害前後による経路切り替え

6.1.1 可 視 化

図 2 の提案システム中に示す，可視化の出力結果を図 5 に示す．(3) より出力された最適経路のリストを基に Google Map 上に可視化を行う．Route1 から Route2 への経路切替が GUI 上でも確認できる．

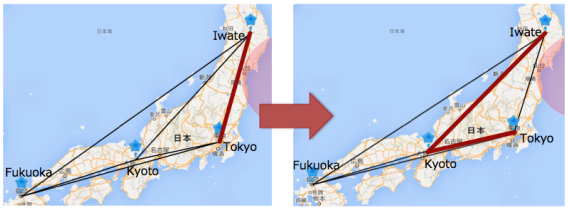

図 5 経路可視化 web サイト

6.1.2 スループットの測定実験

次に，本提案システムを評価するため，iPerf を用いたスループット測定実験を行う．Route2 において，提案システムを適応させた場合は 588 Mbits/sec, 適応させていない場合は 590 Mbits/sec のスループット性能を示した．ほとんど差がないことから，本提案システムは経路切り替えのオーバーヘッドなく，ハードウェア本来の性能を出せることがわかる．

6.1.3 遅延時間の測定実験

次に，経路切替にかかる時間を把握するため，Ping を 1ms 秒毎に 1 つ飛ばし遅延時間を測定する．結果を図 6 に示す．経路切替時に 20-30ms 程度の遅延が発生しているのは，コントローラにフローエントリを問い合わせるためである．

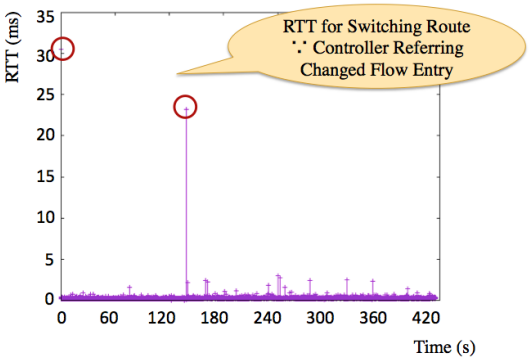

図 6 往復遅延時間（RTT）

6.2 QoS 制御実験

図 2 の (5) について動作を確認する．アプリケーション毎の QoS 制御を達成するために，ユーザ端末でのアプリケーションの識別子の実装と，アプリケーション毎の帯域制御について記述する．

6.2.1 アプリケーション識別子の実装

システム上でアプリケーションの識別を行うために，ユーザ端末でアプリケーションの識別子を付加する．具体的には，まずユーザ空間で SYN パケットをフックし，プロセステーブルと宛先ポート番号でアプリケー

ションの識別を行う．次に識別したアプリケーションの名前と名前の長さをトレーラとして SYN パケットの後ろに付加する．後はチェックサムの再計算をしたところでアプリケーションの情報を付加した SYN パケットを戻す（図 7）．暗号化された通信でも，トレーラは暗号化されないので，アプリケーション情報は自由にアクセス可能となる [17] [18]．

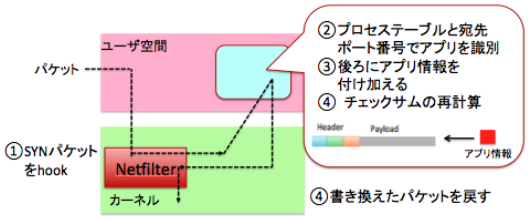

図 7　ユーザ端末でのアプリケーション識別子の付加

6.2.2　アプリケーションの識別と帯域制御

スイッチ上では，前節でアプリケーション識別子を付加された SYN パケットを，コントローラに渡す．コントローラでは，トレーラの中身を見て，アプリケーションの識別を行う．更に，識別したアプリケーション毎に ID を与え，トレーラを外す．ID を指定してキューの設定を行うことで，帯域を調整し，アプリケーション毎の QoS 制御を実装する．図 8 は，FLARE 上で QoS 制御を行った結果である．結果から分かる通り，ID が 0 の YouTube を想定した Firefox のトラフィックは帯域制限により 500Kbps 以下にシェーピングされ，ID が 1 の Skype のトラフィックは 800kbps の帯域保証が行われていることが分かる．

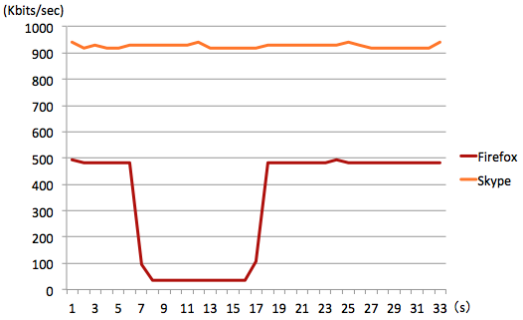

図 8　アプリケーション毎の QoS 制御

7.　まとめと今後の課題

本論文では，SNS によるリアルタイム障害検知システムに基づいて，アプリケーション毎の QoS 制御を行う高度で柔軟なプログラマブル経路制御手法を提案する．本論文の貢献は以下の通りである．

第一に，SNS による障害検知システムからの情報を元に，自動的に障害地区を迂回するシステムの構築を行った．実機実験より，経路切替のオーバーヘッドなくハードウェア本来の性能を出せることに加え，20-30ms 程度の RTT 値で経路切替が出来ることが確認できた．よって，広域ネットワーク上でも提案システムが充分な性能で動作可能であることが導かれる．

第二に，アプリケーションの識別と QoS 制御の実装である．アプリケーション毎の帯域制御が確認できた．

今後の課題としては，アプリケーション毎の QoS 制御を含めた提案システム全体の性能評価を行うことである．また，Twitter からユーザ側の状況をより詳細に抽出し，それを反映したユニークなネットワーク制御を行う．これにより，さらに高度できめ細やかなネットワーク制御を実現する．

謝　　辞

本研究は一部，総務省戦略的情報通信研究開発推進事業（SCOPE）先進的通信アプリケーション開発推進型研究開発および科学技術振興機構戦略的創造研究推進事業（CREST）によるものである．

文　　献

[1] NTT DOCOMO, "Improvement of Credibility for Operation System in the Case of Large Disaster", https://www.nttdocomo.co.jp/binary/pdf/corporate/technology/rd/technical_journal/bn/vol20_4/vol20_4_026jp.pdf, Technical Journal Vol. 20 No. 4, 2013.

[2] Twitter, http://twitter.com/

[3] Chihiro Maru, Miki Enoki, Akihiro Nakao, Shu Yamamoto, Saneyasu Yamaguchi, and Masato Oguchi: "Development of Failure Detection System for Network Control using Collective Intelligence of Social Networking Service in Large-Scale Disaster ” In Proc. the 27th ACM Conference on Hypertext and Social Media (HT2016), pp.267-272, Halifax, Canada, July 2016.

[4] N.McKeown, T.Anderson, H.Balakrishnan, G.Parulkar, L.Peterson, J.Lexford, S.Shenker, and J.Turner. "OpenFlow: enabling innovation in campus networks." ACM SIGCOMM Computer Communication Review, Vol.38, No.2, pp.69-74, 2008.

[5] 日経ネットワーク, http://itpro.nikkeibp.co.jp/article/Keyword/20121112/436523/?rt=nocnt

[6] Akihiro Nakao, "FLARE: Open Deeply Programmable Network Node Architecture," Stanford Univ. Networking Seminar, October 2012. http://netseminar.stanford.edu/10_18_12.html

[7] Akihiro Nakao, "Software-Defined Data Plane En-

hancing SDN and NFV", Special Section on Quality of Diversifying Communication Networks and Services, IEICE Transactions on Communications, vol.E98-B, No.1, pp.12-19, Jan. 2015.
[8] Barefoot Networks, http://www.barefootnetworks.com
[9] NTT ドコモ，東北大学，日本電気株式会社，富士通株式会社，株式会社日立ソリューションズ東日本：「大規模災害時における移動通信ネットワーク動的通信制御技術の研究開発」総務省平成 23-24 年度研究開発.
[10] 小川康一, 吉浦紀晃:「災害 ID 付与方式による災害時のネットワーク優先配送 OpenFlow による実装と評価 」, 電子情報通信学会技術報告, vol.2014-IOT-24, no.23, pp.1-6, 2014-02-20
[11] 小川康一，吉浦紀晃：「通信の信頼性確保を考慮した位置情報に基づくネットワーク運用手法」,DICOMO 2015, 8B-2, pp.1646-1652, 2015
[12] 熊谷友来, 関野雄人, 内田法彦, 柴田義孝：「OpenFlow をベースとした災害時における End-to-End 通信路の選択方法の実現」, 情報処理学会第 75 回全国大会, pp.3-337-338, 2013 年 3 月.
[13] 関野雄人，柴田義孝，内田法彦，白鳥則郎：「OpenFlow をベースとした災害情報ネットワークにおけるリンク切り替え技法の実現に関する研究」, 情報処理学会 SIG, Vol.2013-DPS-154 No.49，2013 年 3 月.
[14] 佐藤剛至，柴田義孝，内田法彦：「SDN によるコグニティブ無線技術を基盤とした災害に強いネバー・ダイ・ネットワークに関する研究」, マルチメディア通信と分散処理, IPSJ-DPSWS2013006, pp.46-52，2013 年 12 月.
[15] 江戸麻人，和泉諭，阿部亨，菅沼拓夫：「災害リスクを考慮したスマートルーティングの設計と実装」, DICOMO 2015, 7E-3, pp.1520-1524，2015 年 7 月.
[16] The Click Modular Router Project, http://www.read.cs.ucla.edu/click/
[17] Akihiro Nakao, Ping Du. "Application and Device Specific Slicing for MVNO", 2014 International Science and Technology Conference (Modern Networking Technologies) (MoNeTeC), Oct. 2014.
[18] Akihiro Nakao, Ping Du, Takamitsu Iwai. "Application Specific Slicing for MVNO through Software-Defined Data Plane Enhancing SDN", IEICE TRANSACTIONS on Communications Vol.E98-B, No.11 pp.2111-2120, 2015.