Short Report

TI-LFA を用いた IP 高速迂回路計算手法の提案

鈴木一哉 [1]

[1] 秋田県立大学システム科学技術学部経営システム工学科

近年，従来ネットワークに接続されていなかったセンサー，自動車，産業機器などあらゆるモノをネットワークに接続する Internet of Things（IoT）と呼ばれる考え方に基づき，実社会における様々な課題解決を目指す動きが拡がっている．IoT が広く普及するに従い，それらの通信を支える IP ネットワークには従来より高い信頼性が求められている．IP ネットワークにおいて故障発生後復旧時間を短縮する技術として，IP 高速迂回（IP Fast Reroute）と呼ばれる手法が提案されている．特に．すべての宛先を迂回対象とできるセグメントルーティングを用いた高速迂回手法である TI-LFA（Topology Independent Loop-Free Alternate）が近年提案され，注目を浴びている．しかし，TI-LFA においてすべての宛先に対する迂回路を効率的に計算する手法はまだ知られていない．そこで本論文では TI-LFA における迂回路を効率的に計算するための手法を提案する．

キーワード：IP ネットワーク，IP 高速迂回，セグメントルーティング，Topology Independent Loop-Free Alternate

1. はじめに

近年，従来ネットワークに接続されていなかったセンサー，自動車，産業機器などあらゆるモノをネットワークに接続する Internet of Things（IoT）と呼ばれる考え方に基づき，実社会における様々な課題解決を目指す動きが拡がっている．IoT が広く普及するに従い，それらの通信を支える IP ネットワークには従来より高い信頼性が求められている．

IP ネットワーク内で故障が発生すると，ネットワーク内の各ルータは，故障を検知したルータから故障情報を取得し，故障箇所を迂回するよう経路を再構成し，パケット転送動作を復旧させる．故障発生から復旧までの間はパケット転送動作が停止するため，故障箇所を経由する通信ではパケットロスが生じる．例えば，従来から存在する動画ストリーミングサービスにおいてパケットロスが生じた場合，一時的に動画再生が停止したり，コマ落ちが生じたりする程度である．しかし，IoT においては，緊急性が高いメッセージを送れないと，重大な事故につながる可能性がある．そのため，IoT 時代には，ネットワーク内の故障発生時にはより短い時間での復旧が必要となる．

故障発生後復旧するまでの時間を短縮する技術として，IP 高速迂回（IP Fast Reroute）と呼ばれる各種手法が提案されている．IP 高速手法では，故障発生時に使用する迂回用経路表を事前に用意することで，復旧時間の短縮を実現している．我々は，これまでいくつかの IP 高速迂回手法を提案してきた（鈴木 2008, Suzuki 2010-1）．従来の IP ネットワークで使用されている IP パケット転送方式に適用可能なこれらの手法には，迂回対象の宛先が限定される（鈴木 2008），経路更新を促すための新たな制御メッセージの導入が必要である（Suzuki 2010-1）といった課題があった．

近年，高速迂回やトラフィックエンジニアリングを目的に，新たなパケット転送方法であるセグメントルーティング（Filsfils 2018）が提案され，実ネットワークへの適用が始まっている．パケットヘッダ中の宛先フィールドのみを用いて転送動作を行う

責任著者連絡先：鈴木 一哉　〒015-0055 由利本荘市土谷字海老ノ口 84-4　公立大学法人秋田県立大学システム科学技術学部経営システム工学科．E-mail: kazuya-suzuki@akita-pu.ac.jp

IP パケット転送方式と異なり，セグメントルーティングでは，経由するルータを示すセグメント ID（IP アドレスや MPLS ラベル）を用いてパケットを転送する．セグメント ID は，パケットの送信元（もしくはセグメントルーティングドメインへの入口となるルータ）が付与する．このように，セグメントルーティングは，経由するルータの指定により，パケットの転送パスを明示的に指定することができる．

このセグメントルーティングを用いて高速迂回を実現する手法である TI-LFA（Topology Independent Loop-Free Alternate）が提案されている（Bashandy 2018）．本論文では，TI-LFA における迂回路を効率的に計算するための手法を提案する．

2. 従来技術と残課題

ここでは，IP 高速迂回と呼ばれる手法の従来技術を説明した上で，本提案で解決を目指す課題を明らかにする．

IP 高速迂回

IP 高速迂回は，リンク故障を検知したノードが，そのリンクが出力先となっている経路を，事前に計算しておいた経路へと切り替えることで，パケット転送の中断時間を短縮する技術である．しかし，IP 高速迂回では，リンク故障に伴う他ノードの経路更新がなされる前に，故障検知ノードが迂回動作を開始するため，故障検知ノードと経路更新前のノードとの間で，ループが発生する可能性がある．そのため，事前計算により代替経路表を用意する段階で，ループが生じるかを判定し，ループが生じない代替経路だけを IP 高速迂回の対象とする Loop Free Alternates（LFA）と呼ばれる手法が提案されている（Atlas 2008）．また我々も，ループが生じない代替経路を効率よく計算する手法（鈴木 2008）を提案している．

文献（Bryant 2015）は，トンネルを使うことでループを避けつつ，高速迂回を実現する Remote LFA（rLFA）と呼ばれる手法を提案している．また，我々は文献（Suzuki 2010-2）にて，迂回用トンネルのエンドポイントを効率的に計算する手法を提案している．トンネルを使用すると，パケットは，トンネル出口を宛先とするトンネルヘッダが付与されて転送される．トンネル出口のルータがトンネルヘッダを除去し，転送することで，パケットは本来の宛先へと到達できる．ただし，トンネルヘッダを付与されたパケットがループせずにトンネル出口まで到達できる必要があるため，rLFA では全てのパケットが高速迂回できるわけではない．

rLFA の課題であるトンネル出口までループせずにパケットを転送するために，セグメントルーティング（Filsfils 2018）を利用する Topology Independent LFA（TI-LFA）と呼ばれる手法が提案されている（Bashandy 2018）．セグメントルーティングは，パケットの転送パスを明示的に指定することができるため，故障箇所迂回の際にループを生じないパスでパケットを宛先まで転送することができる．

これらの手法（LFA, rLFA, TI-LFA）に関して，転送機能変更の必要性と，迂回できない宛先の存在について表 1 にまとめた．宛先への迂回路が存在したとしても，LFA, rLFA では迂回できない宛先が存在する可能性があるのに対し，TI-LFA はすべての宛先を迂回対象とすることができる．LFA の適用には転送機能の変更が必要ないのに対し，rLFA および TI-LFA はそれぞれトンネリング，セグメントルーティングを用いた転送機能を必要とする．このように TI-LFA は全宛先を迂回対象とすることができるが，その適用にはセグメントルーティングに対応したルータが必要となる．

表 1 *高速迂回手法の比較*

	転送機能変更の必要性	迂回できない宛先の存在
LFA	-	✓
rLFA	✓(Tunneling)	✓
TI-LFA	✓(Segment Routing)	-

経路更新が必要となる条件

我々が文献（鈴木 2008, Suzuki 2010-1）で示した，リンク故障発生時，ノードが経路更新する必要があるかの判定条件を 図 1 を用いて説明する．図 1 の（a），(b）は，それぞれノード C, H 間のリンク故障が発生前後を表しており，また図中の矢印は各ノードから宛先ノード K への経路を表している．

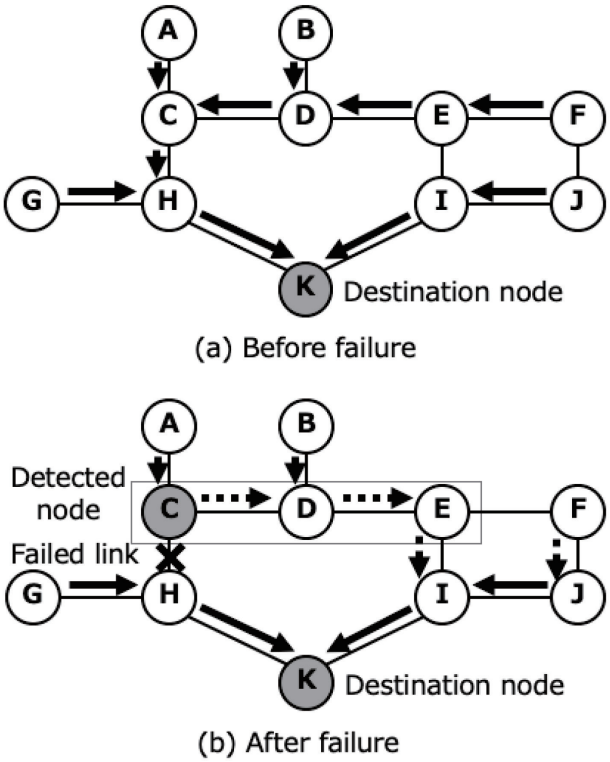

図 1 経路更新が必要となるノード

故障前後に経路が変わる可能性があるのは，故障リンクの上流にあるノードである．図 1 のノード A, B, C, D, E, F が該当する．これらのノードのうち，ノード C, D, E, F は故障前後で経路が変わっている．さらに，ノード C, D, E に関しては，故障前の最短パスはノード C → ノード D → ノード E であるが，故障後の最短パスはノード E → ノード D → ノード C と向きが逆になっている．つまり，これらのノードのいずれかが経路更新を行わなかった場合，ループが発生してしまう．一方ノード F に関しては，リンク故障発生後経路更新を行わなくても，ループとなる事がないため，必ずしも経路更新を必要としない．

つまり，以下の条件を満たすノードが，経路更新をしないとループが生じるノードである．

[条件 1] 故障発生前，自身の最短パスツリー中における宛先までのパス上に故障リンクが存在する．

[条件 2] 故障発生後，故障リンクに接続するノード（検知ノード）を起点とする最短パスツリーにおいて宛先までのパス上に自身が存在する．

これらの両条件を満たすノードは経路更新をしないとループが生じるため，セグメントルーティングを用いて，これらのノードを通過するトンネルを設定する．また，条件 2 を満たすが，条件 1 を満たさないノードをエンドポイントとすることで，ループを起こさず, 宛先までパケットを送ることができる．

3. 提案手法

本章では，セグメントルーティングを用いた迂回路のトンネルエンドポイントを効率的に計算する手法を提案する．

条件を満たすノードの判別手法

セグメントルーティングを用いた迂回路の設定では, 故障を検知したノード(図 1 におけるノード C) が, 本来故障リンクを経由して転送するパケットを，迂回用トンネルへと迂回させる．そのため，故障検知ノード自身が,「経路更新が必要となる条件」において示した条件を満たすパケットの宛先毎のトンネルエンドポイントを，効率的に計算する手法が求められる. 条件 2 を満たすノードを見つけるためには，故障検知ノードが自身を起点とする最短パスツリーをダイクストラ法で計算し，宛先までのパス上にあるノードを列挙すればよい．これらのノードが条件 1 を満たすかを判定するには，単純に考えると，各ノードを起点とする最短パスツリーをそれぞれ計算すればよい. ノード数, リンク数が N, L のネットワークでの計算コストが（$N + L$）× $\log N$ のオーダーになるダイクストラ法を何度も計算するのは効率が良くない．そこで，条件 2 を満たすノードが条件 1 を満たすかどうかの判定を効率的に行う手法を示す．この手法の基本的な考え方は，著者が過去に提案済みである（鈴木 2010-1）．ここでは文献（鈴木 2010-1）にて提示済みの考え方を，迂回用のトンネルエンドポイント決定のアルゴリズムに適用する．

このとき故障リンク e に隣接したノード（故障検知ノード）を n_r, 宛先ノードを n_d とする．また，故障前のネットワークにおいてノード n_i からノード n_j への最短パスのメトリックを $m_{i\to j}$ と表し，故障後のネットワークにおける同メトリックは $m'_{i\to j}$ と表すこととする．これらのノード，メトリックの関係を図 2 に示す．

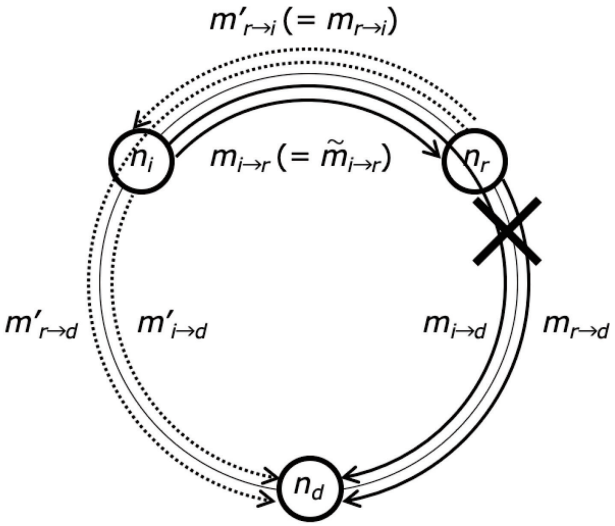

図 2 **各ノード間におけるメトリックの関係**

ノード n_i は条件 1 を満たすと仮定する．このとき，リンク e に接続する故障検知ノード n_r は n_i から n_d への最短パス上のノードであるため，以下の式が成り立つ．

$$m_{i\to d} = m_{i\to r} + m_{r\to d} \tag{1}$$

次に条件 2 が成り立つための必要十分条件を示す．ノード n_r が条件 2 を満たすとき，以下の式が成り立つ．逆もまた真である．

$$m'_{r\to d} = m'_{r\to i} + m'_{i\to d} \tag{2}$$

式(1)と式(2)の関係を示す次の不等式を用いる．始点ノードから終点ノードへの最短パスにおけるメトリックは，故障前の方が故障後よりも小さいかもしくは等しいことは自明なので，以下の不等式が成り立つ．

$$m_{i\to d} \leqq m'_{i\to d} \tag{3}$$

式（1）と式（2）を式（3）に代入すると，以下の不等式が得られる．

$$m_{i\to r} + m_{r\to d} \leqq m'_{r\to d} - m'_{r\to i} \tag{4}$$

リンク e の故障は n_r から n_i への最短パスには影響を与えないため，以下の式が成り立つ．

$$m_{r\to i} = m'_{r\to i} \tag{5}$$

式（4）に式（5）を代入すると以下の式が得られる．

$$m_{i\to r} + m_{r\to d} \leqq m'_{r\to d} - m_{r\to i} \tag{6}$$

$$m_{i\to r} + m_{r\to i} \leqq m'_{r\to d} - m_{r\to d} \tag{7}$$

故障検知ノード n_r を始点とする故障発生前および後のネットワークにおける最短パスツリー $\mathcal{T}_r$, $\mathcal{T}'_r$ を使うことで，任意のノード n_i に対する $m_{r\to d}$, $m_{r\to i}$, $m'_{r\to d}$ を求めることができる．さらに，故障検知ノード n_r を終点とする逆方向の最短パスツリー $\tilde{\mathcal{T}}_r$ を用いることで，任意のノード n_i に対して $m_{i\to r}$ を求めることができる．つまり，最短パスツリーの計算は 3 回のみ（$\mathcal{T}_r$, $\mathcal{T}'_r$, $\tilde{\mathcal{T}}_r$）で，条件 1 を満たす任意のノード n_i が条件 2 を満たすかを判定できる．

提案アルゴリズム

提案アルゴリズムを，図 3 に示す．また，図 3 に示した記号の説明を表 2 に示す．

```
1:  def find_endpoint( n_r, e ):
2:    T_r ⇐ SPF(n_r, V, E)
3:    T'_r ⇐ SPF(n_r, V, E − {e})
4:    T̃_r ⇐ RevSPF(n_r, V, E)
5:    F ⇐ recursive_procedure(n_r, n_r, [n_r], φ)
6:    return F
7:  def recursive_procedure( n_r, n_d, P, F ):
8:    t_d ⇐ Metric(n_r, n_d, T'_r) − Metric(n_r, n_d, T_r)
9:    for n_i in P:
10:     if Metric(n_i, n_r, T̃_r) + Metric(n_r, n_i, T_r) ≤ t_d:
11:       F ⇐ F ∪ {n_d → n_i}
12:       break
13:   for n_j in children(n_d, T'_r):
14:     P' ⇐ P + [n_j]
15:     F ⇐ recursive_procedure(n_r, n_j, P', F)
16:   return F
```

図 3 End-point finding algorithm

表 2 *Notations*

Notation	Mean
$\mathcal{V}$	A set of nodes in network
$\mathcal{E}$	A set of links in network
e	Failed link
$SPF(n,\mathcal{V},\mathcal{E})$	Shortest path tree from source node n with $\{\mathcal{V},\mathcal{E}\}$
$RevSPF(n,\mathcal{V},\mathcal{E})$	Reverse shortest path tree to destination node n with $\{\mathcal{V},\mathcal{E}\}$
$Metric(n_s,n_d,\mathcal{T})$	Metric from n_s to n_d calculated from normal or reverse shortest path tree $\mathcal{T}$
$\mathcal{T}_r$	$SPF(n_r,\mathcal{V},\mathcal{E})$
$\mathcal{T}'_r$	$SPF(n_r,\mathcal{V},\mathcal{E}-\{e\})$
$\widetilde{\mathcal{T}}_r$	$RevSPF(n_r,\mathcal{V},\mathcal{E})$
$\mathcal{P}$	A list of nodes on a path
$\mathcal{F}$	A function from destination nodes to end-point nodes : $\{\mathcal{V}\rightarrow\mathcal{V}\}$
$m_{i\rightarrow d}$	$Metric(n_i,n_d,\mathcal{T}_i)$
$m_{i\rightarrow r}$	$Metric(n_i,n_r,\mathcal{T}_i)$
$m_{r\rightarrow d}$	$Metric(n_r,n_d,\mathcal{T}_r)$
$m'_{r\rightarrow d}$	$Metric(n_r,n_d,\mathcal{T}'_r)$
$m'_{r\rightarrow i}$	$Metric(n_r,n_i,\mathcal{T}'_r)$
$m'_{i\rightarrow d}$	$Metric(n_i,n_d,\mathcal{T}'_i)$
$\widetilde{m}_{i\rightarrow r}$	$Metric(n_i,n_r,\widetilde{\mathcal{T}}_r)$

図 3 に示す提案手法の詳細について説明する．図中の 1 行目から 6 行目がトンネルエンドポイントを求める find_endpoint プロシージャである．このプロシージャは，トンネルの始点ノード n_r と故障リンク e を引数にとる．1 行目から 3 行目では，ノード n_r を始点とする故障前の最短パスツリー $\mathcal{T}_r$，リンク e 故障後の最短パスツリー $\mathcal{T}'_r$，そしてノード n_r を始点とする逆方向の最短パスツリー $\widetilde{\mathcal{T}}_r$ を計算する．5 行目では，宛先ノードからエンドポイントノードへの関数である $\mathcal{F}$ を recursive_function プロシージャを使って求めている．$\mathcal{T}_r$，$\mathcal{T}'_r$ および $\widetilde{\mathcal{T}}_r$ は，後に説明する recursive_function プロシージャ内でも使用する．そのためには本来であれば，これらを recursive_function プロシージャに引数として与える必要がある．しかし，記述が煩雑になるのを避けるため，図 3 中では引数に含めていない．

次に recursive_function プロシージャを説明する．このプロシージャは，始点ノード n_r，判定対象ノード n_i，始点ノードから判定対象ノードへのパスを表すリスト $\mathcal{P}$，これまで計算した宛先ノードからエンドポイントノードへの関数 $\mathcal{F}$ を引数にとる．8 行目では，式（7）の右辺を計算する．9 行目から 12 行目は，始点ノード n_r から宛先ノード n_d へのパス $\mathcal{P}$ 上の各ノード n_i が式（7）を満たしているかを判定し，満たしている場合には宛先ノード n_d に対するトンネルエンドポイントとして n_i を $\mathcal{F}$ に登録する．ノード n_i が式（7）を満たすかの判定は，パス $\mathcal{P}$ 上の始点ノード n_r 側から順に，条件を満たすノードが見つかるまで行われる．このため，宛先ノード n_d に対するトンネルエンドポイントが複数存在したとしても，始点ノード n_r により近いノードがエンドポイントとして選択される．13 行目から 15 行目は，ノード n_r を起点とする最短パスツリー上において，現在判定を行っている宛先ノード n_d の全ての子ノード n_j を宛先ノードとして，recursive_function プロシージャを再帰的に計算する．

find_endpoint プロシージャを実行すると得られる関数 $\mathcal{F}$ は，宛先ノードからエンドポイントノードへの関数である．13 行目から 15 行目では，始点 が n_r であるツリー $\mathcal{T}'_r$ 上の全ノードに対して，処理が行われている．つまり，関数 $\mathcal{F}$ は，定義域が $\mathcal{V}$ であり，$\forall n \in \mathcal{V}$ に対して，それぞれのノードが宛先ノードであるときのエンドポイントノードの情報を保持している．

4. 評価

本章では，シミュレーションを用いた提案手法の評価結果について述べる．

シミュレーションにおけるトポロジー生成には，広く用いられているソフトウェアツールである BRITE を用いた．使用するトポロジーモデルには，ネットワークの評価で広く用いられる Waxman モデルおよび Barabasi-Albert（BA）モデルを使用した．

議論を単純化するために，ネットワークトポロジーの生成に関して以下の仮定を置いた．

- すべてのリンクはポイントトゥポイント型であるとする．
- すべてのリンクは双方向通信可能であり，そのコストは対称であるとする．
- ネットワーク中のすべてのノードはそれぞれアドレスを一つ持ち，リンクはアドレスを持っていないとする．つまり，各ノードが持つ経路表には，宛先としてリンクではなく各ノードが登録される．

オーバーヘッド

セグメントルーティングでは，経由するホップ数分だけヘッダが付与され，その分だけパケットサイズが大きくなる．ここでは付与されるヘッダによるオーバーヘッドについて評価した．

故障発生後のトポロジーにおいて，検知ノードから宛先ノードまでの平均パス長を図 4 および図 6 に，また平均トンネル長を図 5 および図 7 に示す．図 4 と図 5 は Waxman モデルの結果，図 6 と図 7 は BA モデルの結果を示す．

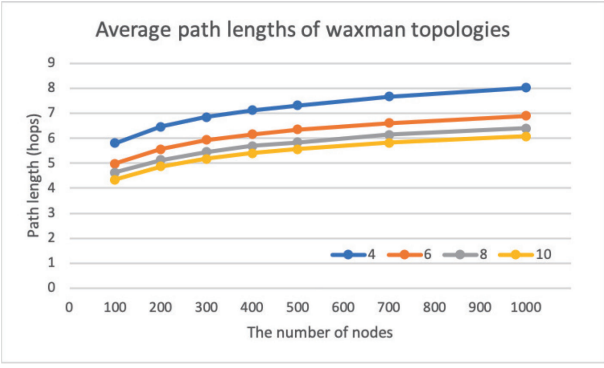

図 4　宛先までの平均パス長（Waxman）

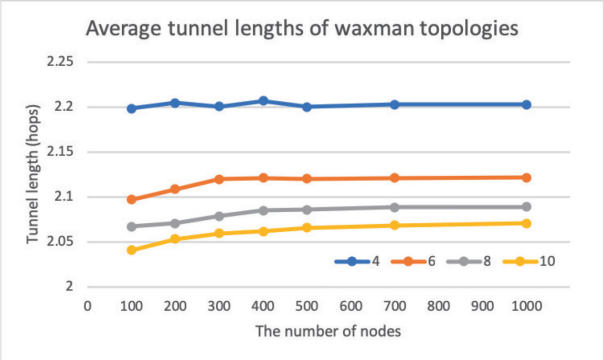

図 5　宛先までの平均トンネル長（Waxman）

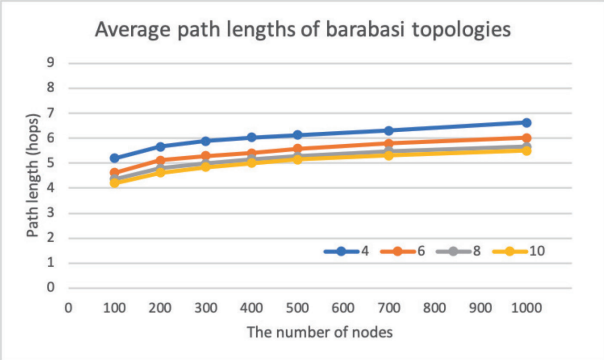

図 6　宛先までの平均パス長（BA）

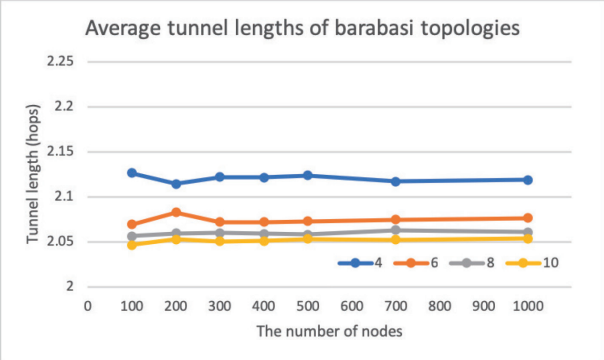

図 7　宛先までの平均トンネル長（BA）

図 4 および図 6 を見ると，シミュレーションに用いたトポロジーにおける平均パス長は，Waxman トポロジーで 4 から 8 程度，BA トポロジーで 4 から 7 程度である．また，いずれもネットワークのノード数が多くなるにつれて，またノード次数が少なくなるにつれてパス長が長くなる傾向が確認できる．

図 5 および図 7 を見ると，提案方式を用いた場合の平均トンネル長は 2.0 から 2.2 程度である．Waxman トポロジー（図 5）の次数 6, 8, 10 では，ノード数が 100 から 500 に増えるにつれて，僅か

であるがトンネル長が長くなる傾向にある. しかし, ノード数が 500 から 1000 にかけては, ノード数に関わらずトンネル長はほぼ一定である．BA トポロジーにおいては，ノード数が少ないときに多少の増減はあるが，トンネル長はノード数にかかわらずほぼ一定であることが図 7 からわかる．つまり，提案手法を用いた場合，ネットワークの規模に関わらず故障発生時の迂回路に用いるトンネル長は 2.0 から 2.2 ホップ程度ですむということを示している．

5. おわりに

本稿では, IP ネットワーク上でのセグメントルーティングを用いた高速迂回に用いるトンネルエンドポイントを効率的に計算する手法を提案した．また評価結果から，提案手法を用いた場合，ネットワークの規模に関わらず故障発生時の迂回路に用いるトンネル長は 2.0 から 2.2 ホップ程度になることを示した．

提案技術の実用化を進めるためには，迂回時におけるトラフィック変化の影響を明らかにする必要がある．故障発生に伴いトラフィックの迂回が発生すると，迂回先にもともと流れていたトラフィックに影響を与える可能性がある．どのような影響がでるのかを明らかにし，悪影響を与える場合にはそれを回避するための手法が必要になる．これらは，今後の研究課題である．

謝辞

本研究は, 平成 30 年度秋田県立大学新任教員スタートアップ支援研究「IP ネットワーク高信頼化のための高速迂回技術に関する研究」の成果です.

文献

A. Atlas, A. Zinin (2008). Basic Specification for IP Fast Reroute: Loop-Free Alternates, RFC5286, IETF.

A. Bashandy, C. Filsfils, B. Decraene, S. Litkowski, P. Francois, D. Voyer (2018). Topology Independent Fast Reroute using Segment Routing, https://tools.ietf.org/html/draft-francois-segment-routing, IETF.

S. Bryant, C. Filsfils, S. Previdi, M. Shand, N. So (2015). Remote Loop-Free Alternate (LFA) Fast Reroute (FRR), RFC7490, IETF.

L. Csikor, G. Retvari (2012). IP Fast Reroute with Remote Loop-Free Alternates: the Unit Link Cost Case, in Proceedings on 4th International Workshop on Reliable Networks Design and Modeling (RNDM 2012), pp.663–669.

C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski and R. Shakir (2018). Segment Routing Architecture, RFC8402, IETF.

鈴木一哉, 地引昌弘（2008）.「IP 高速迂回を実現する迂回可能経路の判別手法」『信学技報』TM2007-59, pp. 37‑42.

K. Suzuki, M. Jibiki, and K. Yoshida (2010-1). Selective Precomputation of Alternate Routes using Link-State Information for IP Fast Restoration, IEICE Transactions on Communications, Vol. E93-B, No. 5, pp. 1085–1094.

K. Suzuki, M. Jibiki, and K. Yoshida (2010-2). Comparison of Proactive and Reactive Methods for IP Fast Restoration using Localization Algorithm, in Proceedings of 4th International Conference on Signal Processing and Communication Systems (ICSPCS), IEEE.

2019 年 6 月 30 日受付
2019 年 7 月 9 日受理

An Algorithm to Efficiently Calculate Alternate Routes in TI-LFA

Kazuya Suzuki[1]

[1] *Department of Management Science and Engineering, Faculty of Systems, Science and Technology, Akita Prefectural University*

Recently, the Internet of Things (IoT) has brought significant benefits by solving various social problems. As the IoT is widely deployed in our social environment, the availability of IP networks carrying IoT data is becoming more important day by day. Various types of IP fast reroutes, which are new technologies to shorten recovery times after failure, are proposed. Topology Independent Loop-Free Alternate (TI-LFA), which is one type of IP fast reroute technology, is attracting increasing attention, because TI-LFA can recover all destination traffic in a network by using Segment Routing. However, the algorithm required to efficiently calculate alternate routes for all destinations in a network remains known. The paper proposes an efficient algorithm to calculate alternate routes in TI-LFA.

Keywords: IP Network, IP Fast Reroute, Segment Routing, Topology Independent Loop-Free Alternate

Correspondence to Taro Akita, Department of Management Science and Engineering, Faculty of Systems, Science and Technology, Akita Prefectural University, 84-4 Tsuchiya-Ebinokuchi, Yurihonjo city, Akita 015-0055, Japan. E-mail: kazuya-suzuki@akita-pu.ac.jp