WIDE Technical-Report in 2008

WIDE-CNRS間の交換留学活動報告

wide-tr-mawi-widecnrs-sora-00.pdf

WIDE Project : http://www.wide.ad.jp/

If you have any comments on this document, please contact to ad@wide.ad.jp

WIDE-CNRS 間の交換留学活動報告

空閑 洋平 (sora@sfc.wide.ad.jp)

2008 年 12 月 15 日

1 概要

WIDE プロジェクトとフランス国立科学研究センター（CNRS）の間での研究協力の一環として，両組織間で人的交流・学術的交流を目的とした，学生の交換留学制度を設けている．慶應義塾大学大学院 村井研究室 修士 2 年の空閑 洋平は，本プログラムの交換留学生として，2008 年 7 月 20 日から 2008 年 10 月 13 日までの約 3 ヶ月間，フランスのパリで現地の研究活動に参加した．受け入れ先は，LIP6 (Laboratoire d’Informatique de Paris 6)[1] の Timur Friedman らの研究室である．滞在中は，Timur らによるインターネットトポロジの計測と解析を目的とした TopHat プロジェクトに参加し，本グループのメンバである Thomas Bourgeau と共に本システムのアーキテクチャについて，議論と実装をおこなった．

TopHat プロジェクト [6] は，2008 年に始まった研究プロジェクトであり，現在も基盤アーキテクチャの議論とシステムの研究開発が続いている．今後は，引き続き Timur 指導の元で Tophat の研究グループに参加していく予定であり，研究プロジェクト間の交流を続けていく．

本報告書では，はじめに第 2 節で滞在中参加した TopHat グループの概要を述べる．次に，第 3 節では，ネットワーク環境に協調したトポロジ探索手法と現在の進捗，そして，今後の作業予定を述べる．第 4 節では，ユーザへのトポロジ情報の提供システムについて述べる．最後に第 5 節では，本交換留学のまとめを述べる．

2 TopHat

滞在中，私は Timur らが活動している OneLab プロジェクト [2] の複数ある研究グループのうち，大規模にインターネットトポロジ情報の収集と解析，そして，収集したトポロジ情報をユーザへ提供するシステムを研究開発している TopHat のグループに参加した．OneLab は，次世代インターネットのテストベット環境構築を目的として， PlanetLab[3] の普及と高度化を進めているヨーロッパ圏の研究グループである．実際に，TopHat グループでは，PlanetLab 上に計測環境を構築し，トポロジデータの収集を開始している．

滞在時，TopHat グループでは，ネットワーク環境に協調したトポロジ探索手法の検討とユーザへのトポロジ情報の提供システムの開発を開始した状況であった．前者の計測手法については， DoubleTree[7] と呼ばれるトポロジ探索アルゴリズムを提案している．また，計測基盤のアーキテクチャは，Timur らが過去に構築したシステムである traceroute@home[5] のアーキテクチャを基にして構築されている． 本アーキテクチャは，PlaneLab 上に計測ノードを配置し，計測ノード間の情報共有に WIDE プロジェクトのメンバである益井 賢次氏が研究開発している N-TAP[4] を採用している．滞在中は，TopHat のトポロジ探索手法の検討とユーザへのトポロジ情報の提供システムを担当して作業した．

3　トポロジ計測手法

本留学では，はじめに TopHat の計測基盤アーキテクチャの理解と議論をおこなった．具体的には，Timur らの論文と実際に動作している TopHat のコードを参照し，Tophat の Problem statement をまとめた．その上で，滞在中の後半では，トポロジ情報の提供システムを担当して，*StitchRoute* アルゴリズムの提案と実装をおこなった．

本節は，*StitchRoute* アルゴリズムを説明するために必要である TopHat の計測機能である DoubleTree アルゴリズムを述べる．*StitchRoute* については，次節で扱う．

3.1　DoubleTree アルゴリズム概要

TopHat はインターネット上に分散配置された PlanetLab 上のノードがそれぞれ traceroute を用いてトポロジ情報を収集し，得られた情報を統合することで，インターネット全体のトポロジ情報を探索する．このような，インターネット全体のトポロジを複数の計測ノードから分散して探索する手法は，他の研究プロジェクトでも採用されている一般的なトポロジ探索手法である．しかし，複数の計測ノードを用いた手法は，探索パケットが重複した経路や同一の宛先ノードを対象とすることで，探索途中のネットワークに対して，高負荷をかける恐れがある．また，重複して経路を探索するため，1 回のトポロジ探索により多くの時間を消費する．

図 1 にインターネット上のノードに対して，高負荷をかける状況を示す．左図では，同一ネットワーク上に複数存在する計測ノードからトポロジ情報を探索した結果，イントラドメインの共通する経路を重複探索している．これらの計測ノードから同一のタイミングでトポロジ探索することで，対象ルータに対して必要以上の負荷をかける恐れがある．また，右図では，分散した計測ノードから，同一の宛先ノードをを対象にトポロジ探索することで，宛先ノードとその近くのネットワークを重複探索している．

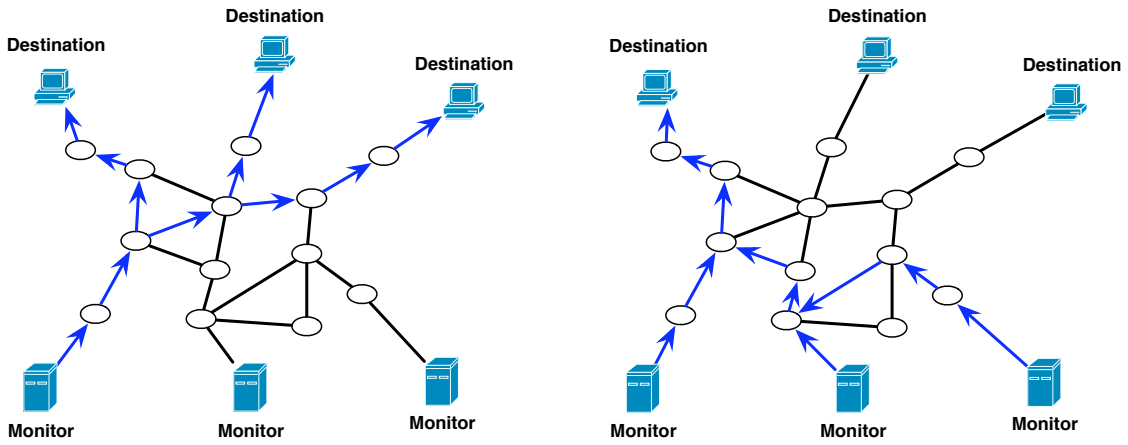

図 1　DoubleTree の扱うトポロジ計測時の問題点

DoubleTree は，計測ノード間で *Stop set* と呼ばれる探索した経路の情報を共有することで，このような重複経路の探索を排除する探索アルゴリズムである．それにより，大規模に展開されるインターネットのトポロジ情報を既存の手法に比べて素早く，また，探索対象のネットワークを構成するノードの負荷を削減できる．

3.2 DoubleTree によるトポロジ探索

DoubleTree によるトポロジ探索手法を述べる．DoubleTree では，traceroute と同じように IP パケットの TTL 値を漸増させることで，計測ノードと宛先ノード間のトポロジ情報を収集する．traceroute との違いは，計測開始時の TTL 値である．計測開始時に，TTL 値 h で計測を開始する．探索は，TTL 値を $h+1, h+2, \ldots$ と漸増させながら宛先ノードまでの経路を探索する *Forward probing* と，TTL 値を $h-1, h-2, \ldots$ と漸減させながら計測ノードまでの逆向きの経路を探索する *Backward probing* を交互におこなう．それにより，経路の中央近くから末端のノード方向へ探索していく．TTL の初期値 h の選定には，事前に計測した，任意の 2 ノード間における直接疎応答された確率 p を基に決定している．また，計測ノード間で探索済みのトポロジ情報を共有することで，重複した経路を探索しないよう調整する．*Forward probing* と *Backward probing* を実行する際，毎試行時に *Stop set* を参照することで，重複探索を判断する．*Forward probing* では，*Global Stop Set* と呼ばれるインタフェース IP アドレスと宛先 IP アドレス，計測ノードから成るデータを用いて判断する．*Global Stop Set* は，計測ノード間で共有される．探索は，*Forward probing* の毎試行時に *Global Stop Set* を参照する．*Global Stop Set* 内から計測ノード自身の宛先 IP アドレスと直前に発見したインタフェース IP アドレスのペアを発見した場合，探索を停止する．一方，*Backward probing* では，*Local Stop Set* と呼ばれるインタフェース IP アドレスのデータから判断する．*Local Stop Set* は，各計測ノードのみで参照され，共有しない．

図 2 に DoubleTree の動作概要を示す．計測ノード (Monitor) A, B, C から宛先ノード (Destination) P に対して DoubleTree を用いてトポロジ情報を収集する．TTL の初期値は $h = 2$ とする．

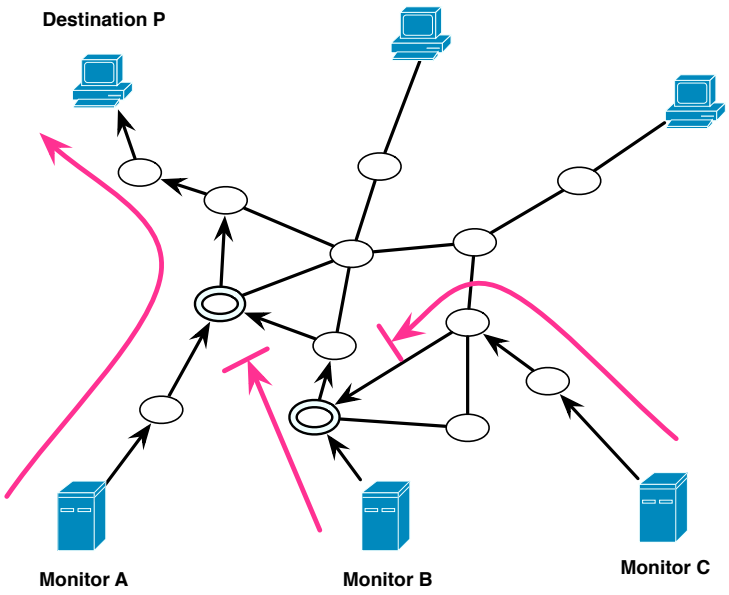

図 2　DoubleTree 動作概要

(1) 計測ノード A から P までの経路を探索する．A は，探索した経路情報を *Global Stop Set* として保存し，さらに A, B, C 間で共有する．

(2) 計測ノード B が宛先ノード P までの経路を探索する．B は，A と同様に TTL 値 $h = 2$ から探索を開始する．B の *Forward probing* は，探索途中で *Global Stop Set* から，すでに A が発見した重複経路部分を発見し，探索を停止される．

(3) 計測ノード C が宛先ノード P までの経路を探索する．C は，B が発見済みの経路までを探索する．

3.3 現状と今後

現在，TopHat グループでは，PlanetLab 上で DoubleTree を用いたトポロジの探索を開始している．今後の予定は，インターネット上の経路ループやロードバランスされた経路の検出に対応する目的で，Paris-traceroute による経路探索手法の置き換え作業をおこなう．

また，グループでは，DoubleTree の TTL 初期値 h の決定方法や，計測結果の提示方法についての議論を続けている．DoubleTree で発見されるトポロジの情報量は，TTL の初期値 h で大きく変動するためである．今後は，実際に TopHat システムで収集したトポロジ情報を解析することで TTL 値を検討し，計測手法を改善していく予定である．

4 StitchRoute

滞在中おこなった作業は，第 3 節で述べた計測手法についての議論に加え，本システムの基幹機能である任意の IP アドレス 2 点間の経路を返答する機能を検討し，実装した．以後，本機能を *StitchRoute* と呼ぶ．

Tophat グループでは，traceroute@home から続くトポロジ計測基盤アーキテクチャの機能をほぼ実装し，実際にトポロジ計測をはじめている．次のステップとして，TopHat グループでは，収集したトポロジ情報を一般ユーザに提供するアーキテクチャを検討している．本システムは，ユーザの XML-RPC によるリクエストに応答する手法で任意の 2 点間の IP アドレスまたは AS 番号の経路情報を提供することを考えている．

4.1 目的

StitchRoute の目的を述べる．本機能は，TopHat が収集した経路の断片 (*piece* と呼ぶ) をつなぎ合わせることで，任意の 2 点間の IP アドレス間の経路を算出するものである．

用語を定義する．計測ノードから宛先ノードまでの完全な経路は，$\mathbf{r} = (r_o, r_1, r_2, \ldots, r_\ell)$ と定義する．r_0 は，ある計測ノードのソース IP アドレスであり，それぞれの値である $r_i, i > 0$ は，ホップ i ごとで発見されたインタフェース IP アドレスである．r_ℓ は，宛先 IP アドレスである．次に，*piece* は，$\mathbf{p} = (p_o, p_1, p_2, \ldots, p_\ell)$ と定義する．p_0 は，*piece* の先頭 IP アドレスであり，p_ℓ は，*piece* の末尾の IP アドレスである．p_0 と p_ℓ は，*Stopset* で経路探索が終了している可能性があるため，必ずしもそれぞれ，計測ノードと宛先ノードの IP アドレスとはかぎならない．この時，*StitchRoute* の目的は，ユーザのリクエスト (S, D) に対して，$\mathbf{p} = (p_0, p_1, p_2, \ldots, p_\ell)$ をつなぎ合わせ，$\mathbf{r} = ((p_{0\_0}, \ldots, p_{0\_\ell}), (p_{1\_0}, \ldots, p_{1\_\ell}), \ldots, (p_{n\_0}, \ldots, p_{n\_\ell}))$ を返答することである．S は *source*，D は destination を表す．

4.2 背景

StitchRoute が必要となる背景を述べる．TopHat では，DoubleTree アルゴリズムを用いて，ネットワークに協調したトポロジ探索の計測基盤を構築した．TopHat では，通常の traceroute の結果と異なり，計測した

経路を断片化された状態でシステム内部で保持する．*piece* は，DoubleTree による経路探索がすでに発見した重複経路で探索を中止するために発生する．

図 3 に経路が断片化される様子を示す．計測ノード A, B, C は，宛先ノード P, Q に対して初期 TTL 値 $h = 3$ で経路を探索する．はじめに P に対して経路を探索する．この時点で，各計測ノードは，*Local Stop Set* を持つ．次に，宛先ノード Q に対して経路探索する．計測ノード A は，自身の *Local Stop Set* を参照することから，XQ 間の経路で探索を終了する．計測ノード B は，自身の *Local Stop Set* と，計測ノード A と共有して得た *Global Stop Set* を参照し，XY 間の経路のみを探索する．計測ノード C は，すでに CQ 間の経路探索が終了していることから，初期探索のみ実行し，探索を終了する．本図の状況で収集したトポロジ情報から CQ 間の経路を知るには，計測ノード C が探索した CX の経路情報と計測ノード B が探索した XY の経路情報，そして計測ノード A が探索した YQ 間の経路をつなぎ合わせる必要がある．

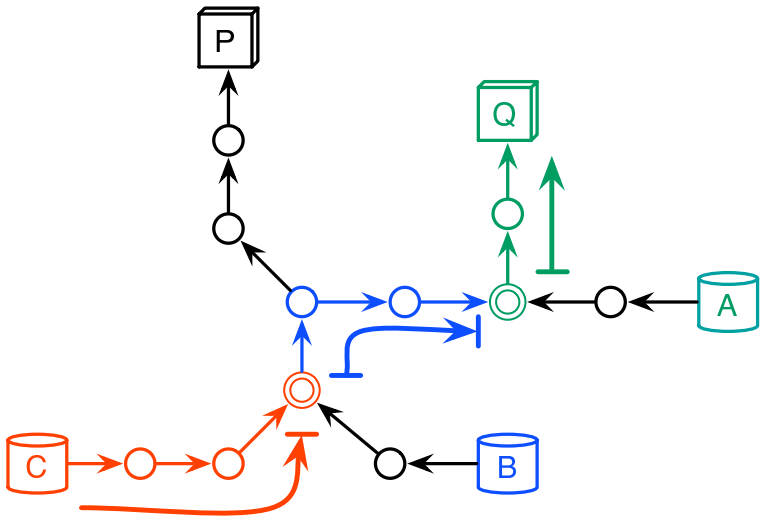

図 3　DoubleTree による探索経路の断片化

4.3　piece

TopHat では，ユーザからのリクエストに応答するため，事前に計測した *pieces* をさらに細かくし，ホップ間の IP アドレスペアに変換して保持する．細かくしたデータ構造は，$(s_n, d_{q+1}, star)$ であり，本データを *piece* と呼ぶ．s は *source*，d を *destinaton*，*star* は IP アドレスペア間に存在する *noreplay* の数をそれぞれ表す．*piece* を

star の初期値は 0 であり，dst_{q+1} が *noreplay* だった場合，*star* の値を漸増し，dst_{q+2} を *destination* とする．また，ロードバランスされたネットワークでは，経路探索時に，同一の TTL 値で複数 IP アドレスから返答される場合がある．このようなトポロジデータは，返答された IP アドレスの数だけ *piece* に分解する．

4.4　アルゴリズム

図 1 に *StitchRoute* アルゴリズムを示す．*StitchRoute* アルゴリズムは，*piece* をつなぎ合わせることで，経路 **r** を探索する．本手法は，インターネットトポロジを探索が計測ノードを幹とした木構造のように，探

索することに注目する．木構造の探索には，反復深化深さ優先探索 (iterative deepening depth-first search: IDSearch) を採用した．単純な深さ優先探索のみでは，経路ループがデータ内に含まる場合に探索が終了しない恐れがある．また幅優先探索では，より多くのメモリを消費することから，探索深度を 1 から漸増させながら深さ優先探索する IDSearch を採用した．

Algorithm 1 Stitchroute algorithm

```
 1: procedure StitchRoute(S, D)                          ▷ source, destination
 2:     i ← 1
 3:     loop
 4:         F̂ ← IdSearch(i, S)
 5:         if F̂ ∩ D then
 6:             response                                  ▷ Output
 7:         else if |F̂ ∩ B̂| > 0 then
 8:             response                                  ▷ No output
 9:         end if
10:         i ← i + 1
11:     end loop
12: end procedure

13: procedure IDSearch(l, P)                             ▷ limit, Path
14:     n ← |P|
15:     m ← P_ℓ
16:     L̂ ← ∅
17:     if n = l then
18:         L̂ ← L̂ ∪ {P_ℓ}
19:     else
20:         for all c ∈ Adjacent(m) do
21:             if c ∩ P = ∅ then
22:                 P ← P ∪ {c}
23:                 IdSearch(l, P)
24:                 P ← P − {P_ℓ}
25:             end if
26:         end for
27:     end if
28: end procedure
```

4.5 今後の予定

StitchRoute は，滞在中にの実装作業をおこなった，今後，実際に TopHat が収集したトポロジ情報を用いて，実行時間を計測する予定である．

5 まとめ

WIDE プロジェクトと CNRS 間の交換留学生として渡仏し，現地の研究プロジェクトに参加した．滞在中は，Timur 指導の元，インターネットトポロジの計測と解析を目的とした TopHat プロジェクトに参加し，本グループのメンバである Thomas Bourgeau と共に本システムのアーキテクチャについて，議論と実装をおこなった．現地での Tophat グループに参加しての作業は，今後も続けていく予定であり，研究協力関係はこれからも継続される．

参考文献

[1] Accueil LIP6. *http://www.lip6.fr/fr/*.

[2] OneLab. *http://www.onelab.eu/*.

[3] PlanetLabEurope. *http://planet-lab.eu/*.

[4] The N-TAP Project. *http://www.n-tap.net/*.

[5] The traceroute@home project's home page. *http://trhome.sourceforge.net/*.

[6] TopHat. *http://top-hat.info/*.

[7] Benoit Donnet, Philippe Raoult, Timur Friedman, and Mark Crovella. Efficient algorithms for large-scale topology discovery. In *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 327–338, New York, NY, USA, 2005. ACM.