社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

エンド間ネットワーク性能向上のための発見的手法に基づくアプリケーション層経路制御手法

松田　一仁† 　長谷川　剛† 　亀井　聡†† 　村田　正幸†

† 大阪大学大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 1–5
†† NTT サービスインテグレーション基盤研究所　〒 180–8585 東京都武蔵野市緑町 3–9–11
E-mail: †{k-matuda,hasegawa,m-murata}@ist.osaka-u.ac.jp, ††kamei.satoshi@lab.ntt.co.jp

あらまし　アプリケーション層において他エンドホストを経由するような経路をエンド間のネットワーク性能を指標として選択することにより，ユーザの体感性能が向上することが昨今の研究により明らかにされている．しかし，各々のエンドユーザが独自に経路選択を行うと，経路の重複によるユーザ性能の低下や，利用する ISP 間トランジットリンクが増加することによる ISP のトランジットコスト増大が考えられる．本稿では，アプリケーション層における経路選択の最適化問題を定義し，これを焼きなまし法を用いて解くことにより，ユーザ性能の向上やトランジットコストの削減を実現する経路制御手法を集中処理，分散処理の 2 つの形で提案する．また，PlanetLab ノード間で経路制御を行うことを想定して，提案手法の性能評価を行う．評価の結果，大幅なネットワーク性能の向上が得られ，特に利用可能帯域においては平均で 84% の性能向上が得られることを示す．

キーワード　ネットワーク最適化，オーバレイネットワーク，経路制御，ISP 間トランジットコスト，焼きなまし法

An application-level routing method for improving end-to-end network performance based on heuristic algorithm

Kazuhito MATSUDA†, Go HASEGAWA†, Satoshi KAMEI††, and Masayuki MURATA†

† Graduate School of Information Science and Technology, Osaka University
Yamadaoka 1–5, Suita-shi, Osaka, 565–0871 Japan
†† NTT Service Integration Laboratories　Midori-cho 3–9–11, Musashino-shi, Tokyo, 180-8585 Japan
E-mail: †{k-matuda,hasegawa,m-murata}@ist.osaka-u.ac.jp, ††kamei.satoshi@lab.ntt.co.jp

Abstract　Recent researches revealed that an application-level routing that chooses a route relaying other end-hosts can improve user-perceived performance. However, selfish selection of route by each application user would lead to decreasing the route performance due to route overlaps, and increasing the inter-ISP transit cost because of utilizing more transit links than IP routing. In the present paper, we first strictly define the application-level routing optimization problem using the routing matrix. We then propose an application-level routing method for improving end-to-end network performance and reducing the transit cost, which based on centralized and distributed processings based on simulated annealing. We evaluate the performance of the proposed method assuming that the PlanetLab nodes utilize an application-level routing. We show that the proposed method can achieve the large degree of improvement of network performance. Especially, in the case of using bandwidth as the routing metric, we can improve the performance by 87% on an average.

Key words　network optimization, overlay network, routing, inter-ISP transit cost, simulated annealing

1. ま え が き

昨今の研究により，アプリケーション層においてエンドホスト間経路を仮想的なリンクとして行う経路制御 (図 1) により，ユーザの体感性能 (以下，ユーザ性能と呼ぶ) である遅延時間，利用可能帯域，パケットロス率などのネットワーク性能が向上することが明らかにされている [1-3]．一方で，各々のユーザが利己的に自身の性能を向上を目的として独自に経路を選択すると，経路の重複によってトラヒックが特定のリンクに集中しユーザ性能が悪化することが考えられる．例えば [4] では複数のオーバレイネットワークが協調せずに動作した場合に，トラヒックが一部のリンクに集中して経路選択が振動することが指摘されている．また，ユーザ性能だけを考慮した経路選択では，利用に際してトラヒック流量に応じた金銭的なコスト (以

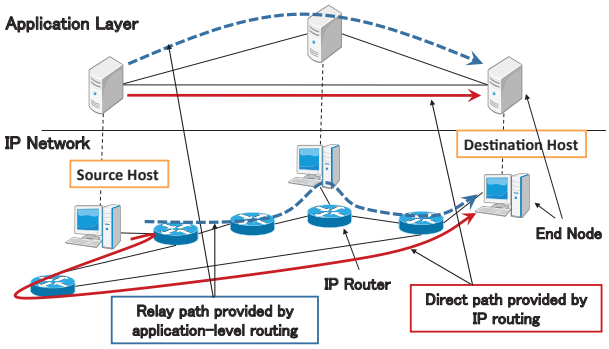


図 1　アプリケーション層における経路制御

下，トランジットコストと呼ぶ) が発生するトランジットリンクの利用数が増加し，その結果 Internet Service Provider (以下，ISP と記述する) のトランジットコストが増大することが考えられる [5]．なお，以降アプリケーション層で行う経路制御において送信元，中継者，宛先となるエンドホストを単にノードと呼ぶ．

我々は [6] で，ノード間のネットワーク性能計測結果を用いたアプリケーション層における経路選択によって，トラヒックが通過するトランジットリンク数を制御できることを示した．このことから，アプリケーション層において経路制御を行う際に増加する ISP のトランジットコストを削減できる可能性があるといえる．しかし，[6] では他ノードの経路選択の影響を考慮しておらず，ネットワーク全体で効率的なトランジットコストの削減ができるかは検証されていない．

ユーザ同士が協調すれば，各ユーザの経路選択状況を考慮した経路制御が可能となる．このような制御を考える場合，一般に 1 つのノードが必要な情報を全て集めて制御を行う集中処理と，複数のノードが自身に関係する部分の制御を行う分散処理が考えられる．集中処理と分散処理にはそれぞれに利点と欠点があり，適用する状況に応じてどちらかを選択するべきである．

本稿では，ユーザ同士が協調して性能の向上を図るアプリケーション層経路制御に着目する．まず，アプリケーション層における経路制御を定式化し，ユーザ性能向上や ISP のトランジットコスト削減を目的とした最適化問題として定義する．次に，定義した最適化問題を発見的手法である焼きなまし法 (Simulated Annealing，以降 SA と記述する) を用いて解く手法を提案する．この際，集中処理に基づく手法と分散処理に基づく手法の 2 つを提案する．これはより広範な状況に対応できるようにするためである．提案手法の性能評価は，PlanetLab [7] を構成するノード間でアプリケーション層において経路制御を行う環境を想定し，ノード間で実測したネットワーク情報を用いて行う．まず，少数ノードに限定した評価で提案手法が最適解に近い性能を得られることを示す．その後，多ノードにおける性能評価の結果を示し，提案手法によってネットワーク性能の大幅な向上が実現できることを示す．また，提案手法を用いる状況についての議論を行う．

2.　アプリケーション層経路最適化問題

本章ではまず，本稿で想定するネットワークモデルについて説明する．続けて，アプリケーション層経路制御を定式化し，最適化問題を定義する．

2.1　ネットワークモデル

本稿では，図 2 に示すようなネットワークモデルを想定する．下位層ネットワークは複数の AS から構成されており，各 AS は複数の IP ルータによって自身のネットワークを構築している．想定するネットワークモデルでは 1 つの AS が 1 つの ISP に対応しているものとして扱う．AS 同士はトランジットリンクまたはピアリングリンクで接続されており，トラヒックがトランジットリンクを通過する際には ISP 間トランジットコストが発生する．各 AS に属するエンドホストをノードとして，下位層ネットワーク上でアプリケーション層における経路制御が行われる環境を想定する．以降，下位層ネットワークによって

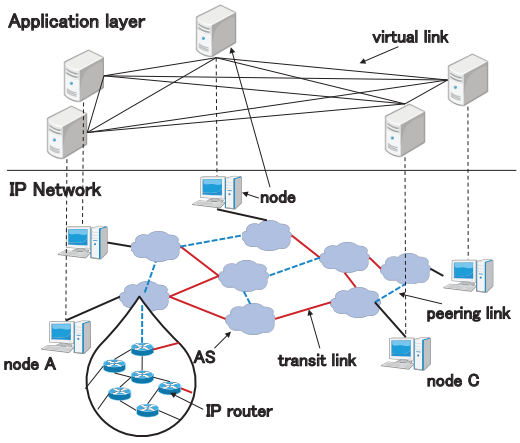


図 2　ネットワークモデル

提供されるノード間の経路を仮想リンクと呼ぶ．

2.2　最適化問題

まずルータレベルでの IP ルーティングの定式化を考える．N はネットワーク内ルータの総数，M はネットワーク内リンクの総数とし，ルータ間には最大で 1 本のリンクがあるとする．ルータ間経路は送信元ルータと宛先ルータで識別できるため，以降ルータを送信元，宛先のペアで考える．ルータペアは 1 から $(N-1)N$ までの識別子を持つとする．このとき，IP ルーティング行列 A^{IP} を次式で定義する．行の添字はルータペアを表し $1,2,\ldots,(N-1)N$ の順に並んでいるとし，列の添字はリンクを表し $1,2,\ldots,M$ の順に並んでいるとする．

$$A^{\mathrm{IP}} = \begin{pmatrix} \mathrm{IP}_1^1 & \cdots & \mathrm{IP}_1^{(N-1)N} \\ \vdots & \ddots & \vdots \\ \mathrm{IP}_M^1 & \cdots & \mathrm{IP}_M^{(N-1)N} \end{pmatrix} \tag{1}$$

要素 IP_i^j は，ルータペア j の経路上にルータ間リンク i が存在するときに 1，存在しなければ 0 となる．

次に，アプリケーション層において仮想リンクで構築されるネットワーク (以下，AL ネットワークと呼ぶ) のトポロジについて考える．すべての IP ルータがノードになる可能性を持つとする．ノードに関しても送信元，宛先のペアで考えるとし，ルータペアと同じ 1 から $(N-1)N$ までの識別子を持つ．このとき，AL ネットワークのトポロジが次式のように与えられるとする．添字は $1,2,\ldots,(N-1)N$ の順に並んでいるとする．

$$(e_1^{\mathrm{AL}}, e_2^{\mathrm{AL}}, \ldots, e_{(N-1)N}^{\mathrm{AL}}) \tag{2}$$

要素 e_j^{AL} が 1 であれば，ノードペア j 間に仮想リンクが存在することを意味する．

あるノードペア j 間で式 (2) で与えられた AL ネットワーク上に張られた仮想リンクを用いる AL ネットワーク上の経路 (以降，AL 経路と呼ぶ) の集合 R_j^{AL} は，次のように定義される．ただし，s_j，t_j はそれぞれノードペア j の送信元ノードと宛先ノードとする．

$$\begin{aligned} R_j^{\mathrm{AL}} = \{&(p_1, p_2, \ldots, p_h) | h \geq 1,\ s_{p_1} = s_j, t_{p_h} = t_j, \\ &t_k = s_{k+1}\ (2 \leq h,\ 1 \leq k \leq h-1), \\ &e_{p_k}^{\mathrm{AL}} = 1\ (1 \leq k \leq h)\} \end{aligned} \tag{3}$$

ここで，$(p_1, p_2, \ldots, p_h)$ はノードペア p_1 から p_h 間の仮想リンクを順に経由する AL 経路を表すとする．

アプリケーション層における経路制御において，ノードペア j $(1 \leq j \leq (N-1)N)$ に対して式 (3) で与えられる集合に含まれる AL 経路 r_j が選択されたとする．このとき，アプリケーション層におけるルーティング行列 A^{AL} を次のように定義する．添字は行，列共に $1,2,\ldots,(N-1)N$ の順に並んでいるとする．

$$A^{\mathrm{AL}} = \begin{pmatrix} \mathrm{AL}_1^1 & \cdots & \mathrm{AL}_1^{(N-1)N} \\ \vdots & \ddots & \vdots \\ \mathrm{AL}_{(N-1)N}^1 & \cdots & \mathrm{AL}_{(N-1)N}^{(N-1)N} \end{pmatrix} \tag{4}$$

表 1　各記号，関数の定義

I	焼きなまし法におけるステップ数.
T	焼きなまし法における温度.
S	焼きなまし法における状態.
Neighbor(S)	状態 S の近傍状態を返す関数.
Cost(S)	状態 S のコストを返す関数.
Random(x, y)	x 以上 y 未満の実数を返す関数.
Probability(T, S, S_{tmp})	コストが悪化する方向への遷移確率を返す関数.
Cooling(T)	次ステップにおける温度を返す関数.
Update(S)	実行時点で受信できた他ノードの情報を用いて最新の状態に更新する関数.
$\mathrm{Neighbor}_{\mathrm{dsa_i}}(S)$	状態 S のうち，ノード i に関係する経路だけを変更して近傍状態を返す関数.
SendNeighbor(S)	状態 S を近隣ノードに送信する関数.

ここで，ノードペア i 間のリンクを経由する AL 経路の集合 R_i^{ALL} を次式で定義する．

$$R_i^{\mathrm{ALL}} = \{(p_1, p_2, \ldots, p_h) | 1 \geqq h,\ \exists k\ p_k = i\} \tag{5}$$

各要素 AL_i^j は式 (5) を用いて次式で表せる．

$$\mathrm{AL}_i^j = \begin{cases} 1 & (r_j \in R_i^{\mathrm{ALL}}) \\ 0 & (r_j \notin R_i^{\mathrm{ALL}}) \end{cases} \tag{6}$$

次に，ネットワークを流れるトラヒックを，通常の IP ルーティングのみで運ばれるトラヒックとアプリケーション層経路制御によって運ばれるトラヒックに分けて考える．通常の IP ルーティングのみを利用する通信のトラヒックデマンドを $X^{\mathrm{IP}} = (x_1^{\mathrm{IP}}\ x_2^{\mathrm{IP}} \cdots x_{(N-1)N}^{\mathrm{IP}})$，アプリケーション層経路制御を利用する通信のトラヒックデマンドを $X^{\mathrm{AL}} = (x_1^{\mathrm{AL}}\ x_2^{\mathrm{AL}} \cdots x_{(N-1)N}^{\mathrm{AL}})$ とする．各要素 x_j^{IP}，x_j^{AL} はそれぞれルータペア j 間，ノードペア j 間のトラヒックデマンドを表すとする．このとき，各ルータ間リンクの負荷を表す行列 $Y = (y_1\ y_2 \cdots y_{(N-1)N})$ は次式で計算できる．

$$Y = A^{\mathrm{IP}} X^{\mathrm{IP}} + A^{\mathrm{IP}} A^{\mathrm{AL}} X^{\mathrm{AL}} \tag{7}$$

ここで，ノード間の各仮想リンクの遅延時間を並べた行列が Y を変数とする関数 f_D によって決定されるとすると，各ノード間の AL 経路の遅延時間を並べた行列 $D^{\mathrm{AL}} = (d_1^{\mathrm{AL}}\ d_2^{\mathrm{AL}} \cdots d_{(N-1)N}^{\mathrm{AL}})$ は次式で表せる．各要素 d_j^{AL} はノードペア j 間の AL 経路の遅延時間を表す．

$$D^{\mathrm{AL}} = f_D(Y) A^{\mathrm{AL}} \tag{8}$$

また，ノード間の各仮想リンクの利用可能帯域を並べた行列が A^{AL} を変数とする関数 $f_B(\mathrm{AL})$ によって決定されるとすると，各ノード間の AL 経路の利用可能帯域を並べた行列 $B^{\mathrm{AL}} = (b_1^{\mathrm{AL}}\ b_2^{\mathrm{AL}} \cdots b_{(N-1)N}^{\mathrm{AL}})$ は次式で表せる．各要素 b_j^{AL} はノードペア j 間の AL 経路の利用可能帯域を表す．

$$B^{\mathrm{AL}} = f_B(A^{\mathrm{AL}}) A^{\mathrm{AL}} \tag{9}$$

ただし，右辺は通常の行列積の演算のように積和を取るのではなく，積算したもののうち最小値を取るものとする．これは値が最小の仮想リンクの利用可能帯域がその経路の利用可能帯域となるためである．

同様に，ノード間の各仮想リンクのトランジットコストを並べた行列が Y を変数とする関数 f_C によって決定されるとすると，各ノード間の仮想リンクのトランジットコストを並べた行列 $C^{\mathrm{AL}} = (c_1^{\mathrm{AL}}\ c_2^{\mathrm{AL}} \cdots c_{(N-1)N}^{\mathrm{AL}})$ は次式で表せる．各要素 c_j^{AL} はノードペア j 間の仮想リンクのトランジットコストを表す．

$$C^{\mathrm{AL}} = f_C(Y) A^{\mathrm{AL}} \tag{10}$$

通信要求を持つノードペアの遅延時間の平均値を最小化する問題は各ノード間の AL 経路 $r_j (j \in K)$ を変数として，次のように与えられる．K は通信要求をもつノードペアの集合とする．

$$\begin{aligned} &\text{minimize :} && (\textstyle\sum_{j \in K} d_j^{\mathrm{AL}})/|K| \\ &\text{subject to :} && r_j \in R_j^{\mathrm{AL}} (j \in K) \end{aligned} \tag{11}$$

通信要求を持つノードペアの利用可能帯域の平均値を最大化する問題は各ノード間の AL 経路 $r_j (j \in K)$ を変数として，次のように与えられる．

$$\begin{aligned} &\text{maximize :} && (\textstyle\sum_{j \in K} b_j^{\mathrm{AL}})/|K| \\ &\text{subject to :} && r_j \in R_j^{\mathrm{AL}} (j \in K) \end{aligned} \tag{12}$$

全仮想リンクのトランジットコストの平均値を最小化する問題は各ノード間の AL 経路 $r_j (j \in K)$ を変数として，次のように与えられる．

$$\begin{aligned} &\text{minimize :} && (\textstyle\sum_{j=1}^{(N-1)N} c^{\mathrm{AL}_j})/((N-1)N) \\ &\text{subject to :} && r_j \in R_j^{\mathrm{AL}} (j \in K) \end{aligned} \tag{13}$$

Algorithm 1 焼きなまし法

```
I ← 0,   T ← T_init,   , S ← S_init
while T > 0 do
    S_tmp ← Neighbor(S)
    if Cost(S) ≧ Cost(S_tmp) then
        S ← S_tmp
    else
        r ← Random(0, 1)
        if r < Probability(T, Cost(S), Cost(S_tmp)) then
            S ← S_tmp
        end if
    end if
    I ← I + 1
    T ← Cooling(T, I)
end while
```

3. 焼きなまし法を用いた経路制御手法

本章では，2. 章において定義した最適化問題の制約条件を満たし，最適解に近い解を求めるための焼きなまし法を用いたアプリケーション層経路制御手法として，集中処理，分散処理に基づく手法をそれぞれ提案する．

3.1 集中処理に基づく手法

集中処理に基づく焼きなまし法を用いたアプリケーション層経路選択手法のアルゴリズムを Algorithm 1 に示す．アルゴリズム中の各記号および関数の定義は表 1 に示す．焼きなまし法は，各ステップで状態を近傍状態に変更するか否かを判断することで処理が進行する．本手法における状態 S とは，通信要求を持つノードペア間のために選択された AL 経路の集合を指す．

Algorithm 1 を用いる際に決定すべきパラメータは，初期状態 S_{init}，近傍選択関数 Neighbor()，コスト関数 Cost()，遷移確率関数 Probability()，初期温度 T_{init}，冷却関数 Cooling() の 6 つである．それぞれについての説明を以下に示す．

初期状態

通信要求を持つノードペア全てがアプリケーション層経路制御が行われない場合の経路，すなわち IP 層の経路制御で得られる経路を選択した状態を初期状態とする．これは，IP 層の経路制御で得られる経路が各ノード間の基本の経路であると考えられるからである．

近傍選択関数

ある状態 S の近傍は，S に含まれる AL 経路のうち，一部の経路を変更した状態とする．経路変更は，変更の候補となる経路から等確率で選択することで行われる．

Algorithm 2 ノード i における分散焼きなまし法

$I_i \leftarrow 0, \quad T_i \leftarrow T_{i_{init}}, \quad S_i \leftarrow S_{i_{init}}$
while $T_i > 0$ **do**
  Update($\mathrm{S_i}$)
  $S_{i_{tmp}} \leftarrow \mathrm{Neighbor_{dsa_i}}(S_i)$
  if $\mathrm{Cost}(S_i) \geqq \mathrm{Cost}(S_{i_{tmp}})$ **then**
    $S_i \leftarrow S_{i_{tmp}}$
  else
    $r_i \leftarrow \mathrm{Random}(0,1)$
    if $r_i < \mathrm{Probability}(T_i, \mathrm{Cost}(S_i), \mathrm{Cost}(S_{i_{tmp}}))$ **then**
      $S_i \leftarrow S_{i_{tmp}}$
    end if
  end if
  $I_i \leftarrow I_i + 1$
  $T_i \leftarrow \mathrm{Cooling}(T_i, I_i)$
  if S_i has been updated **then**
    SendNeighbor(S_i)
  end if
end while

コスト関数

2. 章で定義した最適化問題における目的関数 (式 (11) - (13)) を用いる．式 (11) - (13) で用いられている通信要求を持つノードペアの集合 K が焼きなまし法における状態 S に相当する．

遷移確率関数

遷移確率の算出は，焼きなまし法で一般に用いられている次式で行う．T，S，S_{tmp} はそれぞれ焼きなまし法における現温度，現状態，現状態の近傍状態とする．

$$\mathrm{Probability}(T, S, S_{tmp}) = e^{-\frac{\mathrm{Cost}(S_{tmp}) - \mathrm{Cost}(S)}{T}} \tag{14}$$

初期温度および冷却関数

焼きなまし法においては一般に，初期温度はコストの増減に関わらず遷移が起こる程度に十分に大きくなければならない [8]．具体的な値は適用する環境に合わせて設定する．冷却関数は，焼きなまし法で一般に用いられている次式で行う．

$$\mathrm{Cooling}(T, I) = \gamma T \ (0 < \gamma < 1) \tag{15}$$

3.2 分散処理に基づく手法

分散処理に基づく焼きなまし法を用いたアプリケーション層経路選択手法のアルゴリズムを Algorithm 2 に示す．各記号にはノード i で実行するものであることを示すため添字 i をつけているが，それぞれの記号が示すものは集中処理の場合と同様である．Algorithm 2 は, [9] を基にしたものである．分散処理では，各ノードは独立して処理を実行する．すなわち，各ノードは自身に関係する仮想リンクのネットワーク性能のみを計測し，自身に関係する AL 経路のみを変更可能であるとする．そのため集中処理とは異なり，他ノードが計測した仮想リンクのネットワーク性能，および選択された AL 経路を収集する必要がある．また，近傍選択関数として，分散処理に適したものを用いる必要がある．分散処理を行うことにより，各ノードが自身に関係する経路に対して独自の判断で経路選択が可能になる．反面，他ノードが持つ情報の収集に通信のオーバーヘッドがかかり，通信のオーバーヘッドを抑えるために情報収集の頻度を抑えると他ノードが現在選択している経路の情報にずれが生じて正しくコストを計算できなくなるという，トレードオフの関係を抱える．

以上を踏まえ，Algorithm 2 を用いる際に決定すべきパラメータは，3.1 節で示したものに加え，状態の更新を行う関数 Update()，状態を他ノードへ通知する関数 SendNeighbor() の 2 つと，分散処理に適した形に Neighbor() を変更した分散処理における近傍選択関数 $\mathrm{Neighbor_{dsa_i}}()$ である．それぞれについての説明を以下に示す．

状態の更新を行う関数

本関数では，他ノードが計測したノード間のネットワーク性能および他ノードが選択した経路を受信し，自身の状態の更新を行う．

分散処理における近傍選択関数

分散処理における近傍選択関数は，集中処理のそれとは異なり，自身に関係する経路のみを対象とする．本稿では，自身に関係する経路とは，自身を送信元とする経路とする．

状態を他ノードへ通知する関数

本関数では，他ノードへ変更した経路を通知する．コスト関数の値を計算する際に現時点で他ノードが選択している経路の情報が必要になるが，経路変更が行われる毎に通知を行うと相応の通信によるオーバーヘッドが発生する．そのため，通知の頻度は通信のオーバーヘッドを考慮して決定する．

4. 性 能 評 価

本章では，3. 章で提案した手法を PlanetLab に参加するノードがアプリケーション層経路制御を利用する状況を想定して評価した結果を示す．評価のため，データ取得時に接続可能であった 657 の PlanetLab ノードに関してノード間経路に関するデータを取得した．以下に，取得したデータとその取得方法を示す．

遅延時間，IP レベルホップ数およびパス

PlanetLab ノード間で `traceroute` コマンドを実行して取得した．本稿では 2010 年 10 月 19 日に取得したデータを用いた．

利用可能帯域，物理帯域

Scalable Sensing Service (S^3) [10] において公開されている PlanetLab ノード間のネットワーク性能計測結果を利用した．S^3 では PlanetLab ノード間の物理帯域，遅延時間，利用可能帯域及びパケット棄却率がおよそ 4 時間毎に計測され，その結果が公開されている．本稿では 2010 年 10 月 19 日に取得したデータを用いた．

AS レベルホップ数およびパス

IP レベルのパスに Route Views Project [11] において公開されている AS 番号と IP アドレスプレフィックスの対応を当てはめて取得した．本稿では，2009 年 4 月 16 日に取得したデータを用いた．

AS 間関係情報

CAIDA [12] において公開されている AS 間の関係情報を利用した．本稿では 2010 年 1 月 20 日に取得したデータを用いた．

また，式 (8) - (10) における関数 f_D，f_B，f_C は下記の方法で算出する．f_D は，仮想リンクを計測して得られる伝搬遅延時間とボトルネックとなる IP リンクのリンク利用率と物理帯域を用いて M/M/1 待ち行列モデルを用いて得られるキューイング遅延時間との和を用いる．具体的にはまず，以下の仮定を置く．

- 全ての仮想リンクは下層の IP リンクを共有しない．
- 全ての仮想リンクにおいて物理帯域，利用可能帯域のボトルネックとなる IP リンクは一致しており，エンド間の計測によって取得できる．
- 仮想リンクのキューイング遅延は主としてボトルネックとなる IP リンクで発生し，他の IP リンクで発生するキューイング遅延は無視できる．

以上の仮定を置くと，ある仮想リンク j のキューイング遅延時間 d_j^q は次式で算出される．

$$d_j^q = \frac{u_j}{1 - u_j} \times \frac{P}{w_j} \tag{16}$$

ここで，u_j は仮想リンク j を計測することによって得られた物理帯域，利用可能帯域，及びノード間のトラヒック要求量から計算できるボトルネックリンクの利用率とし，w_j は j の物理帯域，P はパケットサイズの平均値とし本稿では一般的な最大パケットサイズである 1500 バイトと TCP の ACK パケットのサイズである 40 バイトの平均値 770 バイトを用いる．仮想リンク j の遅延時間 d_j は，式 (16) によって求められるキューイング遅延時間 d_j^q と計測によって求められる伝搬遅延時間 d_j^p を用いて次式で計算できる．

$$d_j = d_j^q + d_j^p \tag{17}$$

f_B は，通信要求を持つノードペアに対し，計測した仮想リ

表 2　パラメータ設定

初期温度	0.15
冷却関数	現温度 × 0.999
近傍選択関数において変更する AL 経路数	AL 経路数 × 0.01
通信要求を持つノードペアのトラヒック要求量	1000 kbps
AL 経路が経由するノードの最大数	1

ンクの利用可能帯域を用いて Max-min アルゴリズムを適用して割り当てる関数とする．すなわち，次式の値が小さい仮想リンクから順に仮想リンクを利用するノードペアに均等に帯域を割り当てていく．

$$(b_i^A - \sum\nolimits_{j \in S^i} b_j^{\mathrm{AL}})/|\{k|b_k^{\mathrm{AL}} = 0, k \in S^i\}| \tag{18}$$

ここで，b_i^A は仮想リンク i の利用可能帯域，S^i は i を利用するノードペアの集合とする．

f_C は，トラヒック流量と AS 間リンクの種類によって次式によって決定する．ある仮想リンク j のトランジットコスト c_j^{AL} は，j に流れるトラヒック量 x_j を引数として以下のように決定されるとする．

$$c_j^{\mathrm{AL}} = \beta_j x_j \tag{19}$$

ここで，β_j は仮想リンク j のトラヒック量に対するトランジットコストの増加量を決定するパラメータとし，j に含まれる AS 間リンク毎にその種類によって加算された値の和とする．本評価ではトランジットリンクを 1，ピアリングリンクおよび関係が不明なリンクは 0.05 とした．すなわち，式 (1) の j 列 i 行目の要素を用いて次式のようにトランジットコストの増加量を決定する値 v_i^j を与える．

$$v_i^j = \begin{cases} 1 & (\mathrm{IP}_i^j = 1 \text{ and } i \text{ is a transit link}) \\ 0.05 & (\mathrm{IP}_i^j = 1 \text{ and } i \text{ is a peering link}) \\ 0 & (\mathrm{IP}_i^j = 1 \text{ and } i \text{ is not a AS-level link}) \\ 0 & (\mathrm{IP}_i^j = 0) \end{cases} \tag{20}$$

式 (20) を用いて β_j は次式で計算される．

$$\beta_j = \sum\nolimits_{i=1}^{M} v_i^j \tag{21}$$

なお，計測結果が得られず，上記手法によって遅延時間，利用可能帯域が算出できない場合は，遅延時間の場合は遅延時間が無限大，利用可能帯域の場合は帯域が 0 とし，利用できない仮想リンクとして扱う．

提案手法における各焼きなまし法で用いるコスト関数が返す値は初期状態のコストで正規化を行い，初期温度，冷却関数は全ての性能指標で統一する．各パラメータの設定は表 2 の通りである．また，温度が 10^{-6} を下回った時点で処理を終了するとした．

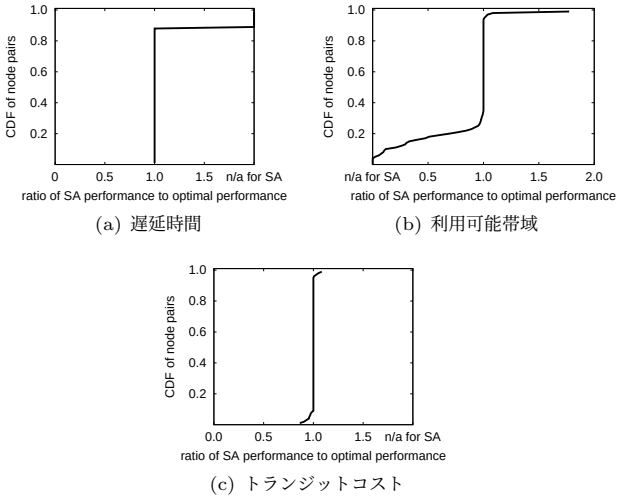
分散処理に基づく提案手法では，3.2 節で述べたように，各ノードが計測したネットワーク性能，および，各ノードが焼きなまし法のステップ毎に選択した AL 経路に関する情報を交換する必要がある．本稿の評価では，ネットワーク性能は全てのノード間で情報交換するとし，AL 経路に関する情報は焼きなまし法のステップ 100 回に対し 1 回の頻度で他の全てのノードと交換するとした．

4.1　最適解と提案手法によって得られる性能の比較結果

まず 3. 章で提案した手法によって得られる性能と最適解を比較する．最適解は総当りで求めるため，計算時間を考慮して少数ノードに制限した評価を行う．10 ノードを PlanetLab ノードよりランダムに選出し，そのうち 7 ノードペアで通信要求が発生するとした．ノードの選出を 10 回，選出したノードに対しノードペアの選出を 10 回行い，計 100 回の試行を行う．提案手法は集中処理に基づいた手法を用いる．評価指標として，各ノードペアの最適解と提案手法によって得られる性能の比を次式で求める．

$$I = b_{sa}/b_{opt} \tag{22}$$

b_{sa} は提案手法によって得られる性能，b_{opt} は最適解の性能とする．



(a) 遅延時間　(b) 利用可能帯域

(c) トランジットコスト

図 3　最適解と提案手法によって得られるネットワーク性能の比較

図 3 は，最適解と集中処理に基づいた提案手法によって得られる性能の比 (式 (22)) の分布である．図 3(a)，図 3(b)，図 3(c) はそれぞれ遅延時間，利用可能帯域，トランジットコストを経路選択の指標とした場合の結果を示す．各結果の一部ノードペアにおいて最適解よりも提案手法によって得られる性能が良い部分があるが，これは提案手法の目的関数が性能指標の平均値の最小化あるいは最大化であるためである．図 3 から，全ての性能指標において 70% 以上のノードペアにおいて最適解と提案手法によって得られる性能が一致しており，提案手法によって最適解に近い性能が得られることがわかる．

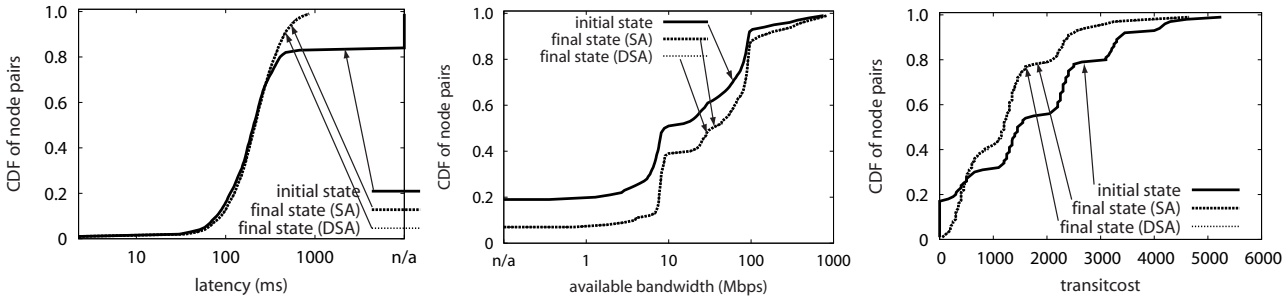
4.2　提案手法を用いた場合のネットワーク性能向上の評価結果

次に，多ノード環境における提案手法の性能評価の結果を示す．30 ノードを PlanetLab ノードよりランダムに選出し，そのうち 10 % のノードペアで通信要求が発生するとして評価を行う．ノードの選出を 10 回，選出したノードに対しノードペアの選出を 10 回行い，計 100 回の試行を行う．

4.2.1　集中処理に基づく手法

図 4 は，集中処理に基づく提案手法によって得られるネットワーク性能の分布である．図 4(a)，図 4(b)，図 4(c) はそれぞれ遅延時間，利用可能帯域，トランジットコストを経路選択の指標とした場合の結果を示しており，それぞれ初期状態 (図中 initial state) と終了状態 (図中 finalstate(SA)) の性能をプロットしている．遅延時間，利用可能帯域の場合は選択した AL 経路の性能の分布を示し，初期状態において直接経路が利用できない場合，および終了状態において利用可能な AL 経路に遷移出来なかった場合は遅延時間の場合はグラフの右端で，利用可能帯域の場合はグラフ左端で計上している．トランジットコストの場合は仮想リンクで発生するトランジットコストの分布を示しているが，初期状態，終了状態のどちらにおいても利用されなかった仮想リンクは計上していない．

遅延時間を指標とした場合 (図 4(a)) において，初期状態で AL 経路が利用可能なノードペアに関してはほぼ性能の変化がないが，初期状態で AL 経路が利用不可能だった経路に関してはほぼ全ての経路において利用可能な代替経路を見つけられていることがわかる．遅延時間に関して性能の向上がほぼ得られないことは [6] でも指摘している通りである．さらに，利用可能帯域を指標とした場合 (図 4(b)) には大幅な性能向上が得られており，初期状態で利用不可能だった経路を除いて，平均 84% の性能向上が得られた．これも [6] で得られている結果と同様の傾向である．トランジットコストを指標とした場合 (図 4(c)) では，試行毎の全仮想リンクのトランジットコストの総和を平均で 27% 削減できた．これは，提案手法によって他ノードを経由する AL 経路を選択することで，通過する仮想リンクは増えるが，逆にトランジットコストは下げられる可能性があることを示している．



(a) 遅延時間　(b) 利用可能帯域　(c) トランジットコスト

図 4　提案手法によって得られるネットワーク性能

4.2.2　分散処理に基づく手法

図 4 は，分散処理に基づく提案手法によって得られるネットワーク性能の分布である．図 4(a)，図 4(b)，図 4(c) はそれぞれ遅延時間，利用可能帯域，トランジットコストを経路選択の指標とした場合の結果を示す．性能比較のため，集中処理に基づいた手法の場合の結果と合わせてプロットしている．初期状態は集中処理と分散処理共に直接経路を用いているため 1 本だけプロットし，終了状態は集中処理の場合 (図中 final state (SA))，分散処理の場合 (図中 final state (DSA)) をプロットしている．図 4 から全ての性能指標において，分散処理を用いた場合でも集中処理と同等の性能が得られることがわかる．これは，分散焼きなまし法に基づいた手法を用いて各ノードが自身を送信元とする経路をそれぞれ独立して変更した場合でも，集中処理に基づく手法と同等の性能が得られることを示している．一方で，今後，情報交換が限定的な場合の性能についての評価を行っていく必要がある．

5. 議　　論

本章では，集中処理，分散処理に基づく提案手法がそれぞれどのような状況で有用かを議論する．また，提案手法の今後の展望について述べる．

集中処理に基づく提案手法の利点は，処理の単純さとネットワーク性能や経路情報の交換にかかるオーバーヘッドを回避できる点である．それ故，他のエンドホストとの多量の通信や計算処理を避けたいユーザに対し，Application Service Provider (ASP) などが一元的に管理してユーザに経路の指示を行う，というような利用が考えられる．

対して分散処理に基づく提案手法の利点は，自ノードの関係する経路を自身で決定できる点にある．この利点により，例えば各 ISP がノードを設置し，それぞれの ISP が自身の利害関係まで含めた経路選択を行うというような利用が考えられる．このような利用の場合には，自身のトポロジ情報等を他 ISP に知らせたくないなどの理由によりノード間で交換できる情報が限定的になる可能性がある．こういった場合の評価は本手法の今後の課題である．

また今後の展望として提案手法によって行う経路制御を現在のインターネットに導入する際，Application-Layer Traffic Optimization (ALTO) [13] で策定が進められているプロトコルを用いることが考えられる．ALTO のプロトコルは，ISP 等が管理するサーバがエンドユーザにネットワーク情報を伝え，トラヒックの流れを最適化しようとするものである．エンドユーザに伝える情報に本稿で提案した手法によって得られる経路選択の情報を取り入れることで，ISP 主導の AL 経路制御を行える可能性がある．こちらの検証も今後の課題である．

6. ま と め

本稿では，エンド間ネットワーク性能を向上のためのアプリケーション層経路制御を集中処理，分散処理のそれぞれに基づいて行う手法を提案した．具体的には，アプリケーション層経路制御を定式化して，経路選択を最適化問題として定義した後，この最適化問題の制約条件を満たす最適解に近い解を集中処理，分散処理に基づく焼きなまし法を用いて求める手法を提案した．また PlanetLab 環境を想定して提案手法の性能評価を行い，少数ノードに限定した評価で最適解に近い性能を得られることを示した後，多ノードでの性能評価によって提案手法がエンド間のネットワーク性能を向上できることを示した．また，提案手法が有用な状況，および提案手法の今後の展望について議論を述べた．

今後の課題として，5. 章で述べたように，分散処理においてノード間の情報交換が限定的な場合の評価を行う必要がある．また，現在のインターネットへの導入を踏まえて，ALTO プロトコルを用いた形への拡張を考える予定である．

謝　　辞

本研究の一部は総務省地球温暖化対策 ICT イノベーション推進事業 (PREDICT) の助成を受けたものである．

文　　献

[1] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller, “Construction of an efficient overlay multicast infrastructure for real-time applications,” in *Proceedings of INFOCOM 2003*, Apr. 2003.

[2] D. G. Andersen, A. C. Snoeren, and H. Balakrishnan, “Best-path vs. multi-path overlay routing,” in *Proceedings of IMC 2003*, Oct. 2003.

[3] Y. Zhu, C. Dovrolis, and M. Ammar, “Dynamic overlay routing based on available bandwidth estimation: A simulation study,” *Computer Networks Journal*, vol. 50, pp. 739–876, Apr. 2006.

[4] R. Keralapura, N. Taft, C. nee Chuah, and G. Iannaccone, “Can ISPs take the heat from overlay networks,” in *Proceedings of HotNets-III Workshop*, Nov. 2004.

[5] K. Matsuda, G. Hasegawa, and M. Murata, “Decreasing ISP transit cost in overlay routing based on multiple regression analysis,” in *Proceedings of ICOIN 2010*, Jan. 2010.

[6] K. Matsuda, G. Hasegawa, S. Kamei, and M. Murata, “Performance evaluation of a method to reduce inter-ISP transit cost caused by overlay routing,” in *NETWORKS 2010*, pp. 250–255, Sept. 2010.

[7] PlanetLab web site. available at `http://www.planet-lab.org/`.

[8] J. Hromkovic, *Algorighmics for Hard Problems*. Springer, 2005.

[9] M. Arshad and M. C. Silaghi, “Distributed simulated annealing and comparison to DSA,” in *Proceedings of the Fourth Workshop on DCR*, Aug. 2003.

[10] Hewlett-Packard Laboratories Scalable Sensing Service. available at `http://networking.hpl.hp.com/s-cube/`.

[11] University of Oregon Route Views Project. available at `http://www.routeviews.org/`.

[12] University of California CAIDA. available at `http://www.caida.org/home/`.

[13] IETF ALTO Working Group web site. available at `http://datatracker.ietf.org/wg/alto/`.