平成11年度知能システム科学専攻修士論文

双方向 AntNet による適応型ネットワーク経路制御の提案

土居 茂雄

Bidirectional AntNet with Loop-Free: An adaptive algorithm for distributed network routing problem

Shigeo DOI

提出年月日 平成 12 年 1 月 25 日

主査教官　山村 雅幸 助教授
審査教官　小林 重信　教授
審査教官　喜多 一　助教授

双方向 AntNet による適応型ネットワーク経路制御の提案

土居 茂雄

Bidirectional AntNet with Loop-Free: An adaptive algorithm for distributed network routing problem

Shigeo DOI

Abstract: AntNet approach has been applied to network routing problem and it has better performance than the classical routing algorithms such as OSPF, Q-Routing through some simulations. However, AntNet is pointed out that it is slow to adapt for dynamic traffic changing. We propose an algorithm BntNetL. it extends AntNet in two aspects: AntNet with Loop-Free. 1) it extends AntNet by collecting traffic statistics bidirectionally with time-estimations to move. 2) ants avoid any loops in their stacks. We show experiments to demonstrate BntNetL has better throughput and packet delays than AntNet and its derivations for situations of local congestion.

1 はじめに

AntNet [1] [2] [3] は Ant Colony System [4] [5] [6] (以下 ACS) をベースとした確率的ネットワークルーティングアルゴリズムである．AntNet は全体を統括することのできないような大規模分散ネットワークシステムの経路制御に適用できるエージェントベースのアルゴリズムである．AntNet は従来から使われているルーティングアルゴリズムである OSPF[7] や Q-Learning を用いたルーティングアルゴリズム Q-Routing [8] ，PQ-Routing [9] と比較して良好な性能を示している．

AntNet は，あるノードに向かうためにどの隣接ノードを選択するかに関する確率が格納されているルーティングテーブルをノード内に持ち，適時 Ant と呼ばれる情報収集エージェントをネットワーク上を徘徊させるため，常に一意の経路を選択するのではなく，ネットワークトラフィックに応じて適応的に経路を切り替えることが可能である．しかしながら，AntNet では，ユーザが任意に生成するトラフィックの変動に弱い現象が指摘されている．この現象を改善するために，ロックフリー AntNet として減衰法 (以下 DCY-AntNet)，負のフィードバック法 (以下 NFB-AntNet) が提案されている．前者はアリのフェロモンの揮発をシミュレートしたものであり，後者はアリの移動に時間がかかった場合に，負の報酬を与える方法である．前者は，Ant の移動時間が非常に長い場合に，ランダムにルーティングしてしまう可能性があるという欠点を有し．後者は，Ant の移動時間が非常に長い場合に，負のフィードバックが与えられるのに時間がかかり，混雑している経路を選んでしまう可能性があるという欠点を有する．

本研究では，AntNet をネットワークの輻輳に強いように改良した BntNetL (Bidirectional AntNet with Loop-Free) を提案し，ユーザが与える任意のトラフィックパターンに対して，シミュレーションを用い本アルゴリズムが有効な場合を示す．さらに，アルゴリズムの振る舞いについて考察することを目的とする．

以下，第 2 章に AntNet の元になった Ant Colony System の説明，第 3 章に本論文でのルーティング問題の定義および関連研究についての説明，第 4 章に BntNetL アルゴリズムの提案を示す．第 5 章に実験環境の設定，第 6 章に SimpleNet 環境下に置ける実験および考察，第 7 章に NSFNET 環境下における実験とその結果および考察を示す．最後に，第 8 章に今後の課題，第 9 章に結論を，それぞれ述べる．

2 Ant Colony System

ACS は，自然界のアリの振る舞いを工学的に応用したアルゴリズムである．ACS において，自然界のアリから模倣している点は以下の点である [6]:

1. 単純な機能しか持たない複数のアリが並行に協調して問題を解く．ACS では全部の巣があるタスクを解くために協調する．
2. 通った経路にフェロモンを落す．自然界のアリにおけるフェロモンは化学物質であるが，ACS ではある数値である．フェロモンのみがアリとアリの通信手段である．
3. 局所的な移動を組み合わせてタスク (最短パスを見つける) を解く．
4. 局所的な情報のみを用いて推計を行う．

自然界のアリと ACS におけるアリとの差違は以下の点である [6]:

1. ACS におけるアリは離散空間における問題を解く．
2. ACS におけるアリは，その中に内部状態 (過去の移動履歴等) を有する．
3. ACS におけるアリが落すフェロモンの量は解の質が良好なほど多くなる．
4. ACS におけるアリがフェロモンを落すタイミングは問題依存．

5. 自然界のアリにないヒューリスティックを導入.

ACS は，TSP，2 次元割当問題，JSP，交通制御問題，グラフ彩色問題などに応用されている [6]．TSP においては，Ant が辿った経路に関して，その経路のコストが短ければ短いほど多くのフェロモンが落される．さらに，Ant が辿った TSP の解候補の中で最良の経路に関してさらにフェロモンを落すということを行い，問題を解かせている [4].

3 ルーティング問題

本論文では，ネットワークルーティング問題に対して，AntNet を改良した BntNetL を提案し，局所的輻輳に関して AntNet よりも有効であることを示す．そこで，ルーティング問題の定義を行い，関連研究に対する説明を行う.

3.1 問題背景

近年，携帯電話などでの電子メール送受信や Web の閲覧，また個人でもパソコンを購入してインターネットに接続するユーザが急増している．それにともない，ユーザが生成するトラフィックも増加することが容易に考えられる．システム的立場では，インターネットは大規模分散システムであり，地球全体にネットワークの構成要素が散らばっているために，どこかでネットワーク全体を集中制御することは実質不可能である．また，最大フローの制限がある環境下で，全部のノードに関して任意のノードへの最適フロー割り当てを決定することは NP 完全問題であることが知られている [10]．また，トラフィックが動的に変化するため，ある瞬間ではこの経路が最適だったが，別の瞬間では別の経路が最適になる，という状況が起こりうる．また，部分的には最適であっても，全体としてパフォーマンスが落ちる場合も考えられ，これはネットワークサービスの質の低下につながる．一般的に，ネットワークが全体として混む場合もあるが，ファイルのダウンロードやある人気のある Web サーバへのアクセス集中など，局所的に負荷が集中する場合が多い.

現在の IPv4 クラス下におけるインターネットでは，OSPF [7] と呼ばれる手法が推奨されている．しかしこのアルゴリズムでは，同一コストの経路が複数ある場合には，適時分散して配送する．しかし，ある行き先について最小コストの経路が 1 つしかない場合には，すべてのパケットが最小コストの経路を経由するため，トラフィックが多くなる場合に最小コストの経路にパケットが集中する欠点を持つ.

3.2 問題設定

ネットワークのトポロジとして，有向グラフ $G = (N, E)$ を考える．N はノードの集合，$E \subset N \times N$ は N を接続するリンクの集合を表し，$(a, b) \in E$ ならば a から b にパケットを直接送信できると定義する．ノードは TCP/IP でいうところのルータとしての役割を果たす．リンクは物理的結線を表し，リンクの属性として，伝搬遅延と帯域幅とを定義する．伝搬遅延は，物理層における電気的信号が届く際に生じる遅延時間を表し，帯域幅はリンクが単位時間に転送できるパケットの最大量を表す．本論文では，TCP/IP プロトコル上のネットワークを対象とし，IP 層におけるルーティングおよびデータグラムパケットを対象とする．これは，パケットの到着順序は関係ないこと，パケットの損失は無視すること，パケットの再送は行わないことを意味する．以下，パケットと記述するときには，特に断りのない限り IP 層のパケットを意味するものとする．実験においては，ネットワークのノードおよびリンクに関する故障は起こらないものとし，パケットの損失は，有限時間内にパケットが目的地に辿り着けなかった場合のみ起こるものとする.

表 1: 記号の定義

$N(k)$	ノード k の隣接ノード集合
$F_{s \to d}$	ノード s から ノード d に向かう Forward Ant
$S_{s \to d}$	$F_{s \to d}$ に格納されているスタック
$B_{d \to s}$	ノード d から ノード s に向かう Backward Ant

3.3 アルゴリズムの評価規範

本論文では，ベストエフォート型のアルゴリズムについて扱う．すなわち，ネットワークに対して任意のトラフィックが生成されるとき，このトラフィックに対して 1) スループット (単位時間当りのパケット受信量) を上げること，2) パケットの伝送時間を短くすること，3) 全到着パケットに関する平均，分散，最長到達時間が小さいこと，4) パケットのロス率を小さいこと．がよいアルゴリズムの評価規範となる．本論文では，パケットをサイズ，送信元ノード，送信先ノードの 3-tuple として抽象化して扱う．すなわち，パケットのヘッダやデータグラムの内容には着目しない．また，トラフィックはパケット群から構成されるものとする.

3.4 関連研究

3.4.1 記法

以降，AntNet および BntNet を解説するために，表 1 に示す記号を用いる.

3.4.2 AntNet

上述した Ant Colony System をネットワーク経路制御問題に応用したのが AntNet [1] [2] [3] [11] であり，Ant と呼ばれるエージェントが適時ネットワークを巡回し，ルーティングテーブルを更新するという強化学習アルゴリズムである.

Ant には Forward Ant と Backward Ant という 2 種類の Ant が存在する．Forward Ant は実際にネットワークを移動することにより，経路がどのくらい輻輳しているかを測定する役割を担う．Backward Ant は Forward Ant が収集した情報をもとに，実際に Forward Ant が移動した経路上にあるノードのルーティングテーブルを，Ant が移動した道を選択する確率が高まるように更新する役割を担う.

任意のノード s は，s から ノード d にパケットを送信するとき，s の隣接ノード n を選択するための確率テーブル $P^s(n, d)$ を持つ．Ant 以外のパケットは，この確率に従いルーティングされる．このとき，$P^s(n, d)$ は次の条件を満たす.

$$\sum_{n \in N(s)} P^s(n, d) = 1, {}^{\forall} s \in N, {}^{\forall} d \in N - \{s\} \quad (1)$$

ルーティングテーブルの更新

ノード k は $^\forall d \in N-\{k\}$ について，統計情報として $^\forall d \in N-\{k\}$ の d に対する μ_d (平均所要時間) および σ_d^2 (所要時間に関する分散)，および最近の $|\mathcal{W}|$ 匹 (最大 $|\mathcal{W}|_{max}$ 匹) の Ant の移動時間を持つ．統計情報は，後述する強化値 r の算出に用いる．

● **Forward Ant の振る舞い**

1. $^\forall s \in N$ について，Forward Ant $F_{s\to d}$ は一定間隔 Δt 毎に生成される．$F_{s\to d}$ の行き先 d は，Ant 以外の通常のパケットの行き先に応じて比例的に選択される．
2. $F_{s\to d}$ がノード k にいるとき，隣接ノード $n \in N(k)$ に移動する確率は次の $P'^k(n,d)$ に従う．ここで，$Q_{k\to n}$ は ノード k における ノード n へのリンクの待ち行列長 [bit]，α は定数である．

$$\begin{cases} P'^k(n,d) &= \frac{P^k(n,d)+\alpha q_{k\to n}}{1+\alpha(|N(k)|-1)} \\ q_{k\to n} &= 1-\frac{Q_{k\to n}}{\sum_{t\in N(k)} Q_{k\to t}} \end{cases} \quad (2)$$

3. Ant がノード k からノード $n \in N(k)$ に移動したとき，$k \to n$ に関する Ant の移動時間をスタック $S_{s\to d}$ に積む．
4. Ant のスタック $S_{s\to d}$ にループが検出された場合，そのループが全体の経路の半分以上の長さならば $F_{s\to d}$ を消去する．そのループが全体の経路の半分未満の長さならば $S_{s\to d}$ のループを消去する．
5. $F_{s\to d}$ がノード d に辿り着くまで 1. ～ 4. を繰り返す．$F_{s\to d}$ がノード d に辿り着いたとき，$F_{s\to d}$ の $S_{s\to d}$ をそのまま受け継いだ $B_{d\to s}$ を生成し，$F_{s\to d}$ は消去する．

● **Backward Ant の振る舞い**

1. $B_{d\to s}$ は $F_{s\to d}$ が辿った経路をそのまま逆に辿る．
2. $B_{d\to s}$ が $d \to \cdots \to f \to k$ という経路を辿り，現在 k にいるとする．このとき，d ～ f 間のノード t (d, f を含む) について，強化値 r を次の式で求める．r は大きいほど $F_{s\to d}$ が短い時間で移動できたことを表す．

$$\begin{cases} r' &= c_1\frac{W_{Best}}{T_{k\to t}} \\ &+ c_2\frac{I_{sup}-W_{Best}}{(I_{sup}-W_{Best})+(T_{k\to t}-W_{Best})} \\ I_{sup} &= \mu + z\frac{\sigma}{\sqrt{|\mathcal{W}|}} \\ s(x) &= \left\{1+\exp\left(\frac{a}{x|N(k)|}\right)\right\}^{-1} \\ r &= \frac{s(r')}{s(1)} \end{cases} \quad (3)$$

ここで，c_1, c_2 は定数，W_{Best} はノードの統計情報から求められる Ant の最良移動時間，$|\mathcal{W}|$ はノードの統計情報に格納されている所要時間の数，$T_{k\to t}$ は $B_{d\to s}$ のスタック $S_{s\to d}$ から求められる $k \to t$ 間の移動時間である．

3. ノードの統計情報を次のようにして更新する．

$$\begin{cases} \mu_d \leftarrow \mu_d + \eta(T-\mu_d) \\ \sigma_d^2 \leftarrow \sigma_d^2 + \eta((T-\mu_d)^2-\sigma_d^2) \end{cases} \quad (4)$$

4. 上で求めた r を用い，$k \to t$ に関するルーティングテーブルの確率を更新する．以下で，$n \in N-\{f\}$ とする．

$$\begin{cases} P^k(f,t) \leftarrow P^k(f,t) + r(1-P^k(f,t)) \\ P^k(n,t) \leftarrow (1-r)P^k(n,t) \end{cases} \quad (5)$$

3.4.3 DCY-AntNet と NFB-AntNet

AntNet においてルーティングロックという問題が報告されている [12]．この現象は，ルーティングテーブルのうち，ある確率が 1 に収束した場合に，そこから離れづらくなることを指す．このルーティングロックが生じる場合に問題となるのは，環境の変化への適応が遅れやすくんなることである．この点を解決するために，[12] では 減衰法 (以下 DCY-AntNet) と負のフィードバック法 (以下 NFB-AntNet) というアルゴリズムが提案されている．

DCY-AntNet

上述した AntNet に加え，ルーティングロックを回避する目的で一定時間毎にフェロモンの揮発をシミュレートするアルゴリズムである．実際には，ノードのルーティングテーブルの確率が 1 / (隣接ノード数) になるように，一定間隔でルーティングテーブルの確率を操作する方法である．$^\forall s \in N$ は，減衰間隔 $\Delta = \frac{\Delta t}{\tau}$ 毎にルーティングテーブルを次のように更新する．v は減衰率を表す．このアルゴリズムは AntNet と並行に実行される．

$$P^s(n,d) \leftarrow (1-v)P^s(n,d) + \frac{v}{|N(s)|} \quad (6)$$

NFB-AntNet

上述した AntNet に加え，Ant が移動して情報を収集した際，長い時間がかかり，かつルーティングロックが起こってると判断されたときには，その経路に関して負の報酬を与えるアルゴリズムである．これにより，輻輳している経路を選択しないようにすることを目的としている．

$$r'' = \frac{T_{k,t}}{c' \cdot m_d} \quad (7)$$

$$\Delta V_d = \sqrt{\mu_d^2+\sigma_d^2} - \sqrt{\bar{\mu}_d^2+\bar{\sigma}_d^2} \quad (8)$$

ここで $\bar{\mu}_d, \bar{\sigma}_d^2$ はそれぞれ $B_{d\to s}$ が k に到着する前のノードの統計情報から得られる平均および分散，c' は定数である．$P^k(p,d) > \alpha'$ かつ $\Delta V_d > \delta$ かつ $r'' \geq 1$ であるとき，次の式を計算し，AntNet でのルーティングテーブルの更新は行わない．それ以外の場合は，AntNet でのルーティングテーブルの更新を行う．

$$P^k(p,d) \leftarrow (1-v')P^k(p,d) + \frac{v'}{|N(k)|} \quad (9)$$

3.4.4 Forward Routing

上述した AntNet, DCY-AntNet, NFB-AntNet らはいわゆる Backward Routing と呼ばれるカテゴリに属する．これは Forward Ant で情報を収集し，Backward Ant によってルーティングテーブルを更新することからこう呼ばれる．一方で，Forward Ant で，Forward Ant の移動方向とは逆方向に関するルーティングテーブルを更新するアルゴリズムが存在し [13]，経路のコストに関して対称性を仮定することにより，Forward Ant

で逆方向のルーティングテーブルの更新をおこなう．このため，半二重環境ではこのアルゴリズムでも問題ないが，全二重環境では，経路のコストの対称性がなくなるため，そのままの形では適用できない．そこで，[13] を拡張したアルゴリズムが複数提案されている．

[13] を拡張したアルゴリズムとして [14] がある．これは [13] では Ant の経由ノード数 (ホップ数) に基づいてノードの情報更新を行っていたものを， リンクの移動コストに基づいて変更したものである．[10] では，各パケットについてノード間の移動時間を付加することによって逆方向の伝送時間を予測する方式をとり，Ant がそのコストを収集することにより，[13] における経路の対称性の仮定を排除している．Van der Put らのアルゴリズムでは，[13] の Forward Routing を AntNet の Backward Routing に変更したものも提案されており，[13] で問題にされている経路のコストの対称性の問題を解決している [6]．最後に，[15] では，辿った経路に対して与えるフェロモン量を，出発地点から現在のノードまでに Ant の経由したノード数に反比例させるようなアルゴリズムを提案している．

3.4.5 その他

[16] では，交叉および突然変異という，遺伝的アルゴリズムで用いられる操作を元に，適応的にルーティングを行う試みがなされている．しかし，このアルゴリズムでは始点経路制御[1]の必要があるため，アルゴリズム的美しさに欠けるとともに，通常パケットの肥大を引き起こす．また，伝送時間の測定に関して，行きと帰りの平均をとることから，全二重のネットワークでは誤ってルーティングテーブルを変更してしまう可能性がある．

4 提案手法: BntNetL

4.1 AntNet の欠点

従来の枠組みである AntNet の欠点として，Ant のスタック内にループが形成されてしまったときは，その Ant は死ぬかあるいはスタックのループを取り除くということを行うだけであり，罰報酬を与えるような設計がなされていない．また，経路が混雑していてもその経路を通る必要があるため，ノードのルーティングテーブルの更新が遅れる欠点を有する．

DCY-AntNet では何らかの作用によって Ant によるルーティングテーブルの更新が行われない場合にルーティングテーブルは隣接ノードをランダムに選択するように変更されていくが，DCY-AntNet の効果を確認するためにはパラメータの調整を慎重に行う必要がある．DCY-AntNet アルゴリズムが並行に実行されるために，Ant が移動しても強化値がほとんど与えられない状況になると，ランダムルーティングを行ってしまう可能性がある．NFB-AntNet では移動のコストがかかった経路に関して負の報酬を与えるので，実際に輻輳している経路を通る必要があり，負の報酬を与えるのにタイムラグが生じる．

著者らが発表した論文 [17] では，AntNet に上述の Forward Ant のメカニズムを導入し，両方向に対して情報の更新を行う作用を入れたが，一方で単方向の輻輳には偽りの情報をノードに与えてしまうことが欠点である．また，[10] では，パケットに前ノードの送信開始時間を付加する必要があるため，通信コストが余計にかかる．

AntNet では，ACS とは異なり，フェロモンに相当する量が確率として表されている．このことは，ルーティングテーブルは他の経路を選択した場合においても，その経路は強化され，他の経路を選択する確率を間接的に減らすことができるということを意味する．このことを用い，トラフィックが変動した場合に，素早く Ant の移動経路を変更できるようなアルゴリズムを考える．

双方向のルーティングテーブル更新　1 匹の Ant で両方向に関するルーティングテーブルを更新する．すなわち，AntNet で行われているような Backward Routing と Forward Routing を一緒に行う．Forward Routing に関しては，[10] のように，ある経路を通ったパケットに関しての所要時間を各ノードで記憶しておくのではなく，所要時間を推定し，それを基に情報更新を行う．

Ant の移動に関する制約　Ant が移動する際に，Ant がループを形成しないように移動先を決定する [18]．すなわち Traveling Salesman Problem や Vehicle Routing Problem での Tabu-list [19] に相当するアルゴリズムである．Ant のある出発点から目的地までの移動を，リアルタイムに辺の重みが変化する環境での最小コストの経路を見つける問題としてとらえる．Ant には AntNet での待ち行列に関するヒューリスティックが適用されるので，これにより目的とする地点にループを形成することなく早く向かえることになる．また，さらに，移動のコストがかかる経路に関しては強化値は少なくなるので，ループを形成しないような経路で最短経路を早く見つけることが可能になる．

4.2 双方向のルーティングテーブル更新

Forward Routing に関しては，所要時間を推定し，それに基づきルーティングテーブルを更新する．Backward Routing に関しては AntNet と同じアルゴリズムを用いる．

Ant が実際に移動していない経路に関して，どのようにその経路の推定所要時間を算出するかを述べる．この推定所要時間を元に，AntNet と同じアルゴリズムで情報更新を行う．また，Ant に対して推定所要時間を格納するスタックを持たせる．これにより，Ant のスタックサイズは 2 倍になる．

Forward Ant $F_{s\to d}$ が $s \to \cdots \to p \to k$ とノードを辿り，現在 c にいるとする．このとき，次の計算式で $k \to p$ 間の推定所要時間を算出する [3]．

$$T(k \to p) = \frac{Q_{k\to p} + Size(s \to d)}{Bandwidth_{k\to p}} + D_{k\to p} \qquad (10)$$

ここで，$Q_{k\to p}$ は ノード k における ノード p へのリンクの待ち行列長 [bit]，$Bandwidth_{k\to p}$ は ノード k における ノード p へのリンクの伝送速度 [bit/s]，$D_{k\to p}$ は ノード k における ノード p へのリンクの伝搬遅延 [s]，$Size(F_{s\to d})$ は $F_{s\to d}$ の大きさ [bit] である．

この $T(k \to p)$ を $F_{s\to d}$ の仮想所要時間のスタック $V_{d\to s}$ に積む．そして，$V_{d\to s}$ を用いて $k \to p, k \to$

[1] パケットの送信元で，送信先に行くための経路を与える方法

$\cdots, k \to s$ に関するルーティングテーブルを更新する．更新のアルゴリズムは AntNet と同じものを用いる．

一般的には，$Bandwidth_{k\to p}$ は与えられることが多いが，$D_{k\to p}$ が明示的に与えられることは少ない．しかし $D_{k\to p}$ は，$D_{k\to p} = D_{p\to k}$，すなわち伝搬遅延の可換性が仮定されるときには，次のようにして求めることが可能である．

1. $F_{s\to d}$ がノード p にいるとき，$T(p \to k)$ を計算し，$F_{s\to d}$ に格納する．
2. $F_{s\to d}$ が実際に $p \to k$ に移動したときに要した時間を $R(p \to k)$ とする．このとき，$R(p \to k) = T(p \to k) + D_{p\to k}$ であるから，条件から $D_{k\to p} = R(p \to k) - T(p \to k)$ として求められる．

このことは，伝搬遅延の可換性が仮定されれば，1 匹の Ant で推定所要時間を用いて逆方向のルーティングテーブルの更新を行ってもよいことを意味しており，[10] よりも少ないコストで Forward Routing が可能になる．

4.3 Ant の移動に関する制約

$F_{s\to d}$ が $s \to \cdots \to p \to k$ とノードを辿り，現在 k にいるとする．このとき，$F_{s\to d}$ は次の行き先を決定するときに，次のアルゴリズムに従ってルーティングを行う．このアルゴリズムを Loop-Free アルゴリズムと呼ぶ．

1. $F_{s\to d}$ の経由したノードの集合を $Visited(F_{s\to d})$ とする．これは $S_{s\to d}$ から計算することができる．
2. $C = N(k) - Visited(F_{s\to d})$ とする．
3. $C = \phi$ ならば，ノード k において $F_{k\to d}$ なる Forward Ant を発生させる．次に，$F_{s\to d}$ で $d \leftarrow k$ とし，ルーティングテーブルや統計情報の更新を行う．さらに，$F_{s\to k}$ のスタックを継承した Backward Ant $B_{k\to s}$ を生成し，$F_{s\to k}$ を消去する．
4. $C \neq \phi$ ならば，$P = \sum_{n\in C} P^k(n,d)$ を計算する．
   - $P = 0$ ならば，C から $F_{s\to d}$ の次の行き先をランダムに決定する．
   - $P \neq 0$ ならば，$n \in C$ について $P''^k(n,d) = \frac{P^k(n,d)}{P}$ を計算し，$P''^k(n,d)$ を (2) 式での $P^k(n,d)$ として扱い，$F_{s\to d}$ の次の行き先を決定する．

このアルゴリズムにより，ノードやリンクが故障した場合にその情報が隣接ノードに伝われば，その情報をその他のノードに伝達することができる．

4.4 BntNetL で期待される効果

BntNetL で期待される効果は，次の 3 点である．これらについて解説を行う．

1. 状況伝搬の高密度化
2. 輻輳経路の高速な伝搬
3. スタブの検出

状況伝搬の高密度化　Ant の生起間隔を AntNet と同じように設定すると，情報更新の間隔が平均 1/2 になる．これにより，トラフィックの変化に素早く対応することが可能になる．

単方向輻輳路の高速な伝搬　まず，従来の AntNet や DCY-AntNet，NFB-AntNet では，実際に Ant が移動した時間を元にノードのルーティングテーブルを更新する．ここで，ネットワーク上の $d \to s$ という経路が輻輳したとする．これらのアルゴリズムでは，Ant は $d \to s$ を避けて別の経路を経由するルートを発見しようとする．しかし，別の経路のコストが輻輳前の $d \to s$ のコストよりも大きい場合には，ループパケットの発生の原因となる．また，ノードがルーティングテーブルを更新する際の強化値が低くなり，ルーティングテーブルの切り替えが遅くなるという欠点を有する．また，$d \to s$ という経路を選択した場合には，実際に Ant が輻輳した経路を通ることになり，これによってもルーティングテーブルの更新が遅れる．一方で，BntNetL では，$d \to s$ が輻輳しても，その逆方向 $s \to d$ を辿ってきた Ant によって，$d \to s$ が輻輳していることを伝搬でき，Ant 内の所要時間に関する履歴を現在のトラフィックに適合させることを助ける．この概念を図 1 に示す．

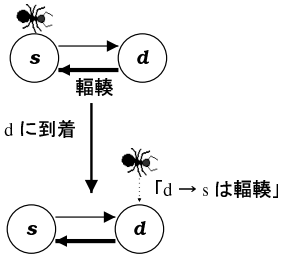

図 1: 輻輳伝搬の概念

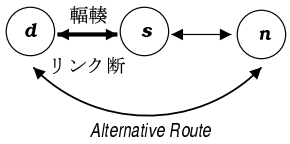

図 2: スタブの概念

スタブの検出　スタブと呼ばれる現象を図 2 に示す．ここで，s の隣接ノードは d と n だけであるとする．この現象は，リンク $s \leftrightarrow d$ が輻輳あるいは故障した場合，ノード n は早くその現象を検知して，別のルートを介して d にパケットを送信しなくてはならないが，$P^n(s,d) = 1$ である場合を考える [12]．すなわち n に到着した d に向かうパケットを誤って s に送信してしまう状態を意味する．s に向かったパケットは再び n に戻らざるを得ず，結果としてループパケットが発生することになり，これはパケットの伝送時間が増加する原因になる．

スタブが起きた場合に重要なことは，ノード n に $s \leftrightarrow d$ の経路が輻輳している，あるいはリンクが断たれていることを素早く伝えることである．しかしながら，従来の AntNet の枠組みでは，ループの発生を抑制できない．なぜなら，s で生成された Ant が d に向かうために，s から d に行くための経路が混んでいるので，高い確率で n に向かおうとする．しかし $P^n(s,d) = 1$ の場合には，高い確率で Ant が s に戻りループを形成する．この状態を解消するためには，n から s に向かう経路の待ち行列長が n から “Alternative Route” を経由して行く経路よりも十分長くなくてはならない．これは $s \sim n$ の間にループパケットが多く生成されている

ことを意味し，スループットが減少し，伝送時間を増加させる原因となる．一方で，BntNetL では Loop-Free アルゴリズムの導入により Ant のループを抑制することができる．また AntNet の枠組みにより， s で生成された Ant は n に行く確率は高くなり，Loop-Free アルゴリズムにより仮に $P^n(s,d)=1$ であったとしても再び s に戻って Ant が死んでしまうようなことは起こらない．従って，別の経路を検出し，強化することが可能である．また，BntNetL の持つ双方向のルーティングテーブル更新作用により，d から n 経由で s に向かった Ant によって s は d に行くために n にパケットを送信する確率を増加させることができる．従って，スタブが生じたときの対応が AntNet と比較して早くなると考えられる．

5 実験: 環境設定

5.1 実験目的

様々なトラフィック環境下での各アルゴリズムのパフォーマンスを比較し，提案手法である BntNetL がどのような状況で有用であるかを示すとともに，考察を行う．まず，ベンチマーク問題を行い，BntNetL の有用な状況に探り，さらに実際に用いられていたネットワーク上で BntNetL の振る舞いを調べる．

5.2 環境設定

第 6 章と第 7 章で示す実験について，全実験で共通に用いられるパラメータを表 2 〜表 4 に示す．通常のパケットに関する最大パケット長は 1500[Bytes] とし，1500 [Bytes] 以上のパケットの送信要求が来た場合には，1500[Bytes] のパケットを複数と残りのパケット，というようにパケットを分割する [20]．フラグメントにともない，IP パケットのフラグメントヘッダを考慮しなくてはならない [20] が，パケットの平均サイズの分布が崩れるため，単純に分割するのみとする．また，前学習時間では，最初の時刻にのみパケットを送信し，それ以降はシミュレーション開始までは通常のパケットは送信しない．また，パケット生存時間に関して，パケット送信時に生存時間を過ぎていた場合に消去するものとする．ただし，Ant に関しては生存時間を無限大とする．DCY-AntNet, NFB-AntNet の設定に関しては，[12] と同じ値を用いる．

ノードのパケット処理に関する内部構造を図 3 に示す．隣接ノードおよび自ノードで生成されたパケットは入力バッファに入る．入力バッファでは FIFO に従った処理を行う．入力バッファから取り出したパケットの行き先がこのノードならば，そのノードに処理をまかせ，そうでなければルーティングテーブルを用い，次にどの隣接ノードにパケットを送信するかを決定する．決定後は，出力バッファにパケットを並ばせ，FIFO に従って順に配送する．ただし，Backward Ant は Forward Ant や通常のデータパケットよりも優先的に処理される．

BntNetL と AntNet, DCY-AntNet と NFB-AntNet では，Ant のパケットサイズ及びノードでの処理時間に関してパラメータ設定が異なる．ノードの処理時間の内訳としては，Ant がスタックにリンクの移動時間を積んだり，ルーティングテーブルの更新にかかる時間である．Ant のパケットサイズに関しては，BntNetL ではスタックを 2 つ持つために，AntNet における Ant の大きさの 2 倍に設定する．また，逆方向による更新も行うため，ノードでの処理時間を 2 倍にしている．

表 2: シミュレーションパラメータ (共通)

Forward Ant 送信間隔	300[ms]
入力バッファ容量	∞
出力バッファ容量	各 1[Gbit]
通常パケット生存時間	15[s]
単位シミュレーション時間	0.1[ms]
最大パケット長	1500[Bytes]
ルーティングテーブルの初期確率	均等
実験時間	1000[s]
前学習時間	500[s]
$(\|\mathcal{W}\|_{max}, \alpha, c_1, c_2, z, a, \eta)$	$(300, 5, 0.7, 0.3, 1.70, 5, 0.15)$

表 3: シミュレーションパラメータ (BntNetL)

ノード処理時間	6[ms]
Forward Ant のスタックサイズ	48 + 16 × (ホップ数)
Backward Ant のスタックサイズ	48 + 16 × (全ホップ数)

表 4: シミュレーションパラメータ (AntNet, DCY-AntNet, NFB-AntNet)

ノード処理時間	3[ms]
Forward Ant のスタックサイズ	24 + 8 × (ホップ数)
Backward Ant のスタックサイズ	24 + 8 × (全ホップ数)
DCY-AntNet: (v, τ)	$(0.00015, 10)$
NFB-AntNet: $(c', \alpha', \delta, v')$	$(2, 0.9, 1, 0.2)$

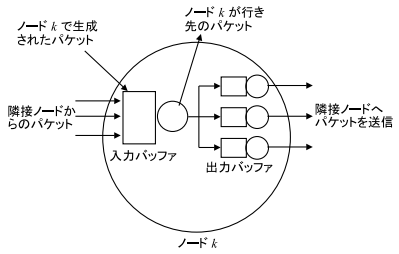

図 3: ノードの構造

以下の各実験においては，各アルゴリズムについて 10 回の実験を行う．測定項目は，単位時間のパケット受信量 (以下スループット)[bit/s]，パケット伝送時間分布，パケット伝送時間に関する平均 [ms] および分散 [ms^2]，パケット最長伝送時間 [ms](以下 最長伝送時間)，パケット損失率 [%] である．以下の実験結果においては，10 回の実験の平均をとったものを測定結果とする．ただし，パケット伝送時間に関する平均，分散，パケット最長伝送時間に関しては，10 回の実験をすべて加えあわせた全分布に関する値をとるものとする．各統計量に関しては，それぞれの中で最小値を取るものに関して太字で強調する．

6 実験: ベンチマーク問題

各アルゴリズムのベンチマークをとる目的で，図 4 で示す SimpleNet [2] トポロジでの実験を行う．トポロジの各リンクは全二重，帯域幅は 10[Mbps]，伝搬遅延は 1[ms] であり，その他のパラメータは表 2 と同じものを用いる．また，前学習時間で与えるトラフィックは，各ノードが 1 および 6 に向けてパケットを一つ送信するものとする．これには各ノードが 1 や 6 に向かう経路を学習させる目的がある．

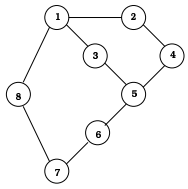

図 4: SimpleNet トポロジ

表 5: F-CBR::パケット伝送時間に関する統計量

	平均	分散	最長時間	損失率
BntNetL	**31.8**	**705.0**	**737.2**	**0.003**
AntNet	59.4	4373.2	1332.9	0.006
DCY-AntNet	81.5	9718.3	1750.0	0.006
NFB-AntNet	68.3	5564.2	2108.9	0.008

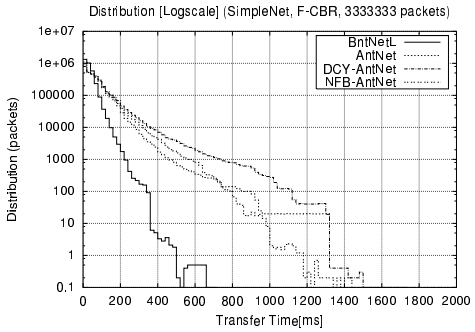

図 6: F-CBR::パケット伝送時間分布 (対数)

6.1 F-CBR

[2] で行われていた実験を行う．トラフィックパターンは，ノード 1 から ノード 6 に向けて 0.3[ms] 間隔で 512[Bytes] 長のパケットを送信する，というものを用いる．前学習時間では，時刻 0.1[ms] において各ノードはノード 1 およびノード 6 に向けてパケットを送信し，それ以外のトラフィックはないものとする．実験結果を表 5，図 5，図 6 に示す．図 5 では，横軸に時間の変化，縦軸にスループットを取る．以下，スループットに関するグラフはこの見方に従うものとする．また，図 6 では横軸にパケットの伝送時間を，縦軸にその時間に到着したパケット数の分布を表す．以下，題名に “(対数)” と書かれているグラフは，この見方に従うものとする．

図 5 をみてわかるのは，他のアルゴリズムと比較して，BntNetL ではスループットのぶれが小さく抑えられていることである．これは Ant の双方向の情報更新作用により，6 から 1 に向かった Ant によっても 1 から 6 に向かった Ant によってもルーティングテーブルが更新される効果である．スループットのぶれがおさえられている結果から，パケットの待ち行列での詰まりも少なくなっていると考えられる，図 6 で示されるパケットの伝送時間分布に関して他のアルゴリズムよりも良くなっていることが言える．またこの状況では，前学習時間で学習したルーティングテーブルをそのまま生かせる静的なトラフィックのため，Loop-Free の効果はないと考えられる．

6.2 F-CBR2

BntNetL の双方向性のルーティングテーブル更新の効果を確認するために，上述の F-CBR を両方向のトラフィックが発生するように拡張する．トラフィックパターンは，ノード 1 から ノード 6 および ノード 6 からノード 1 に向けて 0.3[ms] 間隔で 512[Bytes] 長のパケットを送信する．その他の設定は F-CBR と同じものを用いる．実験結果を表 6，図 7，図 8 に示す．

表 6: F-CBR2::パケット伝送時間に関する統計量

	平均	分散	最長時間	損失率
BntNetL	**29.6**	**651.0**	**352.4**	**0.003**
AntNet	77.8	9380.5	2208.0	0.008
DCY-AntNet	76.1	8724.4	1897.4	0.007
NFB-AntNet	67.6	5805.0	2503.9	0.007

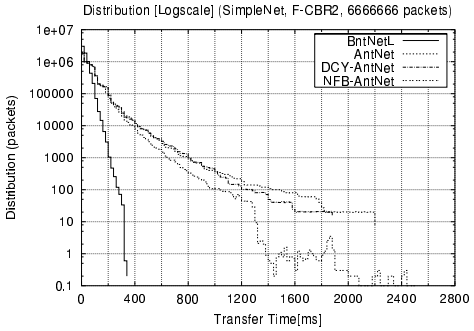

図 8: F-CBR2::パケット伝送時間分布 (対数)

AntNet に関してはパケットの平均伝送時間が F-CBR の場合と比較すると伸びていることがわかる．これは，ノード 1 およびノード 6 を除く途中のノードについて，Ant の行き先がそれまでのトラフィックに依存するために，ルーティングテーブルの更新頻度が減るためと思われる．DCY-AntNet ではむしろ F-CBR2 で性能が上がっており，NFB-AntNet ではそれほど性能に変化がないことがわかる．この中でも最良の性能を誇っているのは BntNetL である．F-CBR で示されている結果よりも良い結果が出ていることから，ノードに双方向の情報を伝達する作用により，両方向のトラフィックにおいて，特に早くパケットを早く伝達できる性能を有すると言える．F-CBR で BntNetL の性能が落ちているのは，6 → 1 方向に向かった Ant が，仮に逆方向の 1 → 6 方向が混んでいるリンクを経由した場合でも，その経路を強化してしまうためであると考えられ，これは両方向の情報更新による副作用であると言える．

6.3 HotSpot1

6.3.1 アルゴリズム能力比較

スタブ発生時の各ルーティングアルゴリズムの輻輳による適応能力を確認するために，学習した経路を塞ぐように特定ノード間についてトラフィックを大量に発生させ，環境が突然変化した場合の適応能力を見る．トラフィックパターンを以下に示す．

- 0[s] ～ 1000[s] の間，1 ↔ 6 の間に 0.5[ms] 間隔で 512[Bytes] のパケットを発生させる．
- 350[s] ～ 450[s] の間，1 ↔ 8 の間に 0.3[ms] 間隔で 512[Bytes] のパケットを発生させる．これは単純に 1 ↔ 8 間で直接転送しているだけでは伝送遅延が増加していく．
- 450[s] ～ 550[s] の間，1 ↔ 3 の間に 0.3[ms] 間隔で 512[Bytes] のパケットを発生させる．

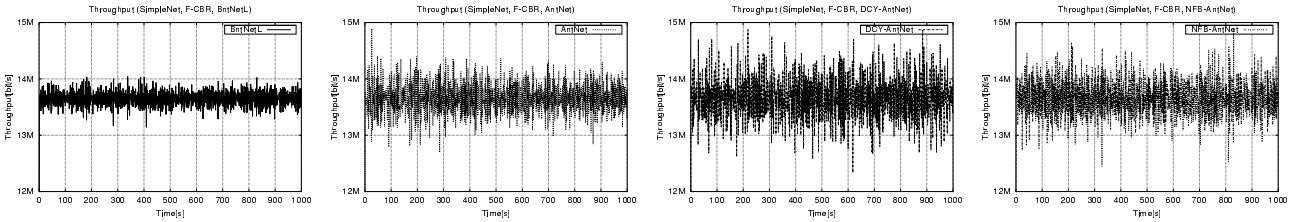

図 5: F-CBR::スループット (左から BntNetL, AntNet, DCY-AntNet, NFB-AntNet)

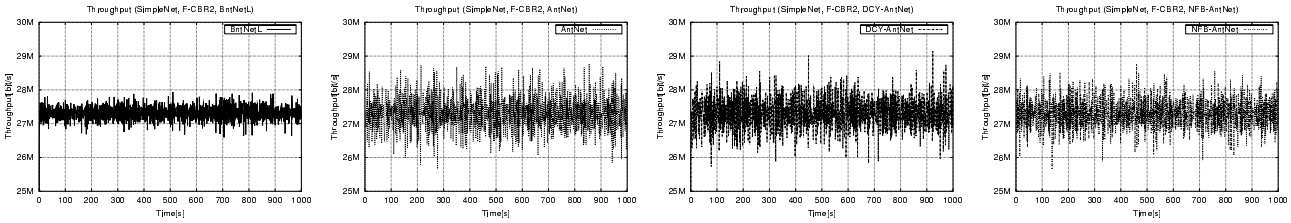

図 7: F-CBR2::スループット (左から BntNetL, AntNet, DCY-AntNet, NFB-AntNet)

表 7: HotSpot1::パケット伝送時間に関する統計量

	平均	分散	最長時間	損失率
BntNetL	**1015.8**	**6164291.1**	15001.3	**0.008**
BntNetV	1261.6	12823958.7	15001.3	1.045
AntNet	2508.1	24102414.9	15001.3	1.822
DCY-AntNet	1837.2	15456732.1	15001.3	0.336
NFB-AntNet	1776.6	14514782.2	15001.3	0.400

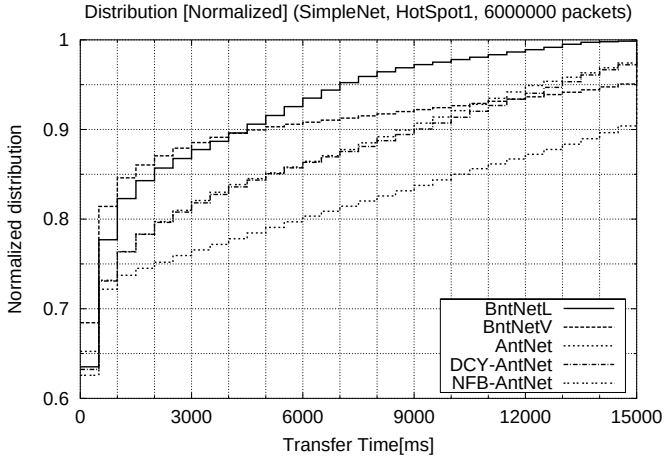

図 9: HotSpot1::パケット伝送時間分布 (正規化)

- 550[s] ～ 650[s] の間，1 ↔ 2 の間に 0.3[ms] 間隔で 512[Bytes] のパケットを発生させる．

この実験については，実験対象アルゴリズムとして，AntNet，DCY-AntNet，NFB-AntNet，BntNetL，BntNetL から Loop-Free を取り除いたアルゴリズム (以下 BntNetV) の 5 つのアルゴリズムを用いて比較を行う．前学習時間では F-CBR2 と同じようにパケットを送信する．実験結果を表 7，図 9，図 10 に示す．図 9 では，横軸にパケットの伝送時間を，縦軸には全送信パケットのうち，横軸の時間までに到着した割合を表す．以下，題名に “(正規化)” と記されているグラフは，この見方に従うものとする．

450[s] を越えた時点で一気にスループットが上がるのは，ネットワークの配送容量を越えてパケットを生成しているからである．1 と 8 の間でパケットを 0.3[ms/packet]，1 と 6 との間で 0.5[ms/packet] でパケットを生成し，パケット当り 512[Bytes] の大きさを有するから，必要とする回線容量は 43.69[Mbit/s] となるが，6 と 5 の間は共有され，1 から 6 に向かうパケット以外に回せる余剰帯域幅は片道約 2[Mbit/s] である．しかし実際には入力として 8 から 1 の間には約 13.6[Mbit/s] のトラフィックが生じる．このために，仮に経路制御がうまくいってもパケットの詰まりが生じてしまう．

表 7 で見てわかるのは，BntNetL がパケット到着の平均時間および分散，パケットロスト率を小さくできていることである．この理由について，以下の 3 点が生じた場合について考察する．

まず， 350[s] ～ 450[s] における 1 ↔ 8 で輻輳が起きた場合について考察する．まず，1 ↔ 8 でできるだけ速くパケットを送信するためには，1 と 8 の間で直接送信するだけではなく，1 ↔ 3 ↔ 5 ↔ 6 ↔ 7 ↔ 8 という経路も利用しなければならない．ここで，8 から 1 に向かう Ant について考えてみる．他の AntNet アルゴリズムでは，8 から 7 に向かい，7 から 1 に向かうときに， 8 および 6 に関して待ち行列がない場合に，以前の 1 ↔ 6 に関する経路を学習しているために， 8 に向かう可能性が高くなり，ループが検出されて Ant は死んでしまう．すなわち，スタブが発生している状態になる．

BntNetL では，Loop-Free ヒューリスティックにより，8 から 1 に向かうために 7 に向かった Ant は 6 に向かうことができると考えられる．BntNetV では，双方向の情報更新を行うものの，1 で生成された Ant が 2 あるいは 3 に向かっても，再び 1 に戻る可能性は大きいと考えられる．これによりループが発生し，死んでしまう Ant も多くなり，これが BntNetV と BntNetL のスループットの差として現れているものと言える．これと同じことは 1 で生成された Ant についても同じようなことが言える，すなわち，BntNetV では 8 に辿り着けずに死んでしまう Ant が多くなるために，スループットの上昇が AntNet とさほど変わらないと言える．

450[s] ～ 550[s] における 1 ↔ 3 で輻輳が起きた場合について考察する．1 ↔ 3 でできるだけ速くパケットを送信するためには，1 と 3 の間で直接送信するだけ

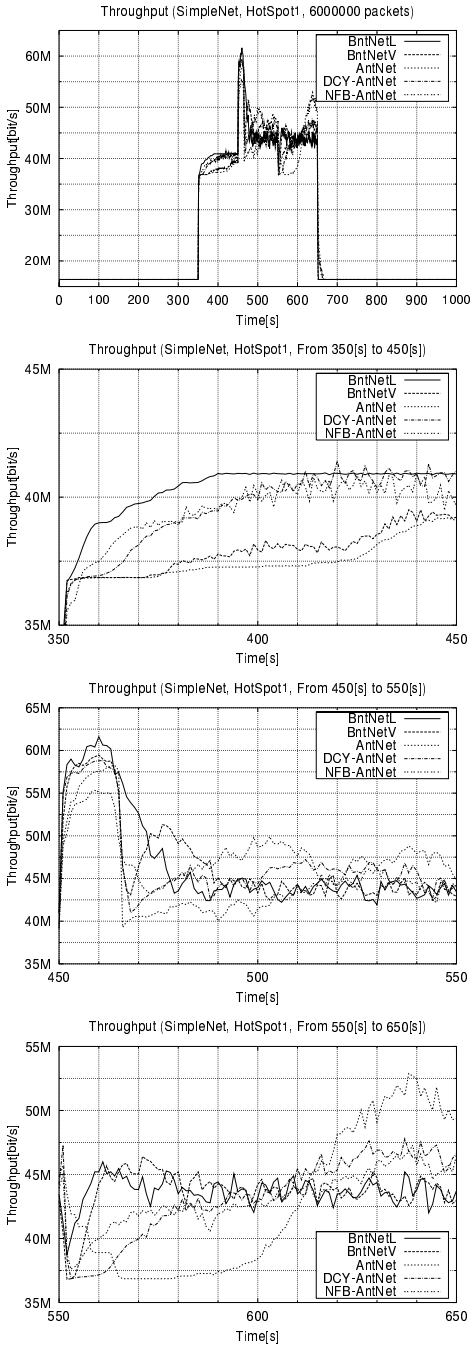

図 10: HotSpot1::スループット拡大図 (上から 0[s]〜1000[s], 350[s]〜450[s], 450[s]〜550[s], 550[s] 〜 650[s])

ではなく，1 ↔ 2 ↔ 4 ↔ 5 ↔ 3 という経路も利用しなければならない．1 と 6 の間を流れるパケットは，1 と 8 の間のパケットの詰まりが解消された後 (565[s]) 以降は，1 ↔ 8 ↔ 7 ↔ 6 という経路を辿れば良いが，その詰まりが解消されるまでは 5 を経由するのが最短経路となる． 550[s] を越えてすぐは，1 で生成され 3 に向かう Ant は，ヒューリスティックから 2 に向かう可能性が高くなる．BntNetL では，そのまま 4，5，3 と辿る，あるいは 5 の後に 6 へと向かうことが考えられる．Ant が 5 から 6 に向かった場合は，8 まで辿り着いた後に Ant の移動先がなくなるために，再び 3 に向かう Ant が生成される．このとき，また同様に 1 まで戻ったとしても，Loop-Free アルゴリズムにより 3 に行き着くことができる．その他のアルゴリズムでは 1 に戻りループにより死ぬ可能性がある．

表 8: HotSpot1::Ant の生起間隔を変更した場合のパケット伝送時間に関する統計量

生起間隔 [ms]	平均	分散	最長時間	損失率
300	1015.8	6164291.1	15001.3	**0.008**
600	**927.7**	**4906013.5**	15001.3	0.009
900	1057.4	6514414.3	15001.3	0.010
1200	1119.3	7423866.0	15001.3	0.029
3000	1644.2	13451153.3	15001.3	0.351

550[s] 〜 650[s] における 1 ↔ 2 で輻輳が起きた場合について考察する．1 ↔ 2 の輻輳が起きた場合にできるだけ速くパケットを送信するためには，1 と 2 の間で直接送信するだけではなく，1 ↔ 3 ↔ 5 ↔ 4 ↔ 2 という経路も利用しなければならない．また， 1 ↔ 6 間を流れるパケットは 1 ↔ 8 ↔ 7 ↔ 6 を通ればいいことになる．

1 と 2 の間で輻輳が起きる前に 1 から 3 の輻輳が起きていて，直接配送する経路と 1 ↔ 2 ↔ 4 ↔ 5 ↔ 3 の経路を分散して配送するのがもっとも速いことになる．ここで，2 と 3 の間を 5 経由で転送するような経路の学習ができていることになると考えられる．そして，1 と 2 の間に輻輳が起きたときに，直接転送する経路は待ち行列長のヒューリスティックにより Ant が避ける場合が多くなる．そこで，1 で生成された Ant は 8 ないしは 3 に向かうが，3 に向かった Ant は 3 から 2 へ向かう経路は学習済であるため，1 から 3 経由で 2 に向かう経路を素早く強化できる．また，その Ant が 2 へ到着したときにも，2 から 4 経由で 1 に向かう経路を， 4 から 5 経由で 1 に向かう経路を素早く学習できる．

BntNetL では 2 から 4 に向かった Ant が 4 から再び 2 に戻ることは起こらないが，BntNetV では 再び 2 に戻る可能性が高い．1 から 2 に向かう Ant に関しては BntNetV も BntNetL でも同じ振る舞いを示すが，2 から向かう Ant に関してはこのような振る舞いの差が生じ，これからスループットの立ち上がりに微妙な差が生じているものと思われる．

すなわち，図 2 で s が 2 つ以上存在するようなスタブが生じる場合，Loop-Free アルゴリズムの性質によって適応能力を上げることができると考えられる．これについては，後述する節で追加実験を行う．

6.3.2 Ant の生起間隔を変化させた場合の比較

BntNetL において Ant の生起間隔のみを変化させた場合についてパフォーマンスの比較を行い，どの程度の生起間隔が適切であるかを実験的に求める．生起間隔以外のパラメータに関してはすべて同じものを用いている．トポロジとして SimpleNet，トラフィックパターンとして HotSpot1 を用いる．実験結果を表 8，図 11，図 12 に示す．

平均や分散で最良のパフォーマンスを誇っているのは，Ant の生起間隔が 600[ms] の場合であるが，損失率が少ないのは Ant の生起間隔が 300[ms] の場合である．しかしこの 2 者の間の差はわずかであり，どちらがいいかとは一概に言えないが，ネットワークのトラフィックに対して損失を少なくする観点で行けば，Ant の生起間隔が 300[ms] の時が一番いいと言える．また，表 7 と表 8 を比較すると，BntNetL では Ant の生起間隔を 3000[ms] にしても，BntNetL，BntNetV 以外のア

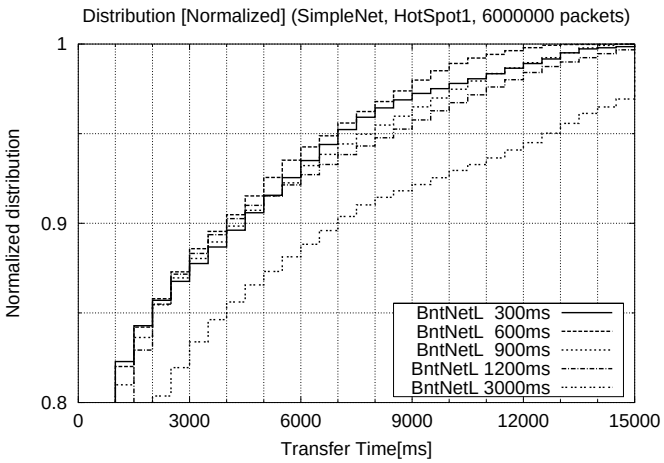

図 11: HotSpot1::Ant の生起間隔を変更した場合のパケット伝送時間分布 (正規化)

表 9: HotSpot2::パケット伝送時間に関する統計量

	平均	分散	最長時間	損失率
BntNetL	**1358.0**	**12050909.3**	15001.3	**0.266**
BntNetV	1386.5	12675403.2	15001.3	0.299
AntNet	2617.8	25268836.2	15001.3	3.144
DCY-AntNet	2147.4	20410575.1	15001.3	1.228
NFB-AntNet	1976.0	18473826.5	15001.3	1.122

ルゴリズムで最良であった NFB-AntNet よりも平均伝送時間で短いことがわかる．これは 350[s] ～ 450[s] でのスループットの立ち上がりにおける差が原因であると考えられ，これは Loop-Free と双方向の更新による効果である．また，スループットにおける「ぶれ」は，Ant の生起間隔が小さい方が安定している．これはルーティングテーブルをトラフィックに応じて適応的に切り替えられることによると考えられる．

6.4 HotSpot2

図 2 で s が 1 つしか存在しないようなトラフィックを生成することにより BntNetL の Loop-Free の効果は s が 1 つしかない場合にはそれほど有効ではないことを実験的に示す．入力として与えるトラフィックパターンは次の通りである．

- 0[s] ～ 1000[s] の間，$1 \leftrightarrow 6$ の間に 0.5[ms] 間隔で 512[Bytes] のパケットを発生させる．
- 350[s] ～ 450[s] の間，$7 \leftrightarrow 8$ の間に 0.3[ms] 間隔で 512[Bytes] のパケットを発生させる．これは単純に $7 \leftrightarrow 8$ 間で直接転送しているだけでは伝送遅延が増加していく．
- 450[s] ～ 550[s] の間，$3 \leftrightarrow 5$ の間に 0.3[ms] 間隔で 512[Bytes] のパケットを発生させる．
- 550[s] ～ 650[s] の間，$4 \leftrightarrow 5$ の間に 0.3[ms] 間隔で 512[Bytes] のパケットを発生させる．

実験結果を表 9，図 13，図 14 に示す．

表 7 と表 9 を比較すると，全体的に平均伝送時間が増加している．BntNetL について考えてみると，$1 \leftrightarrow 6$ の間で伝送しているパケットについて，スループットが減少したことに起因すると考えられる．HotSpot1 では 1 から 6 に向かう Ant は (2) 式が適用されて，輻輳

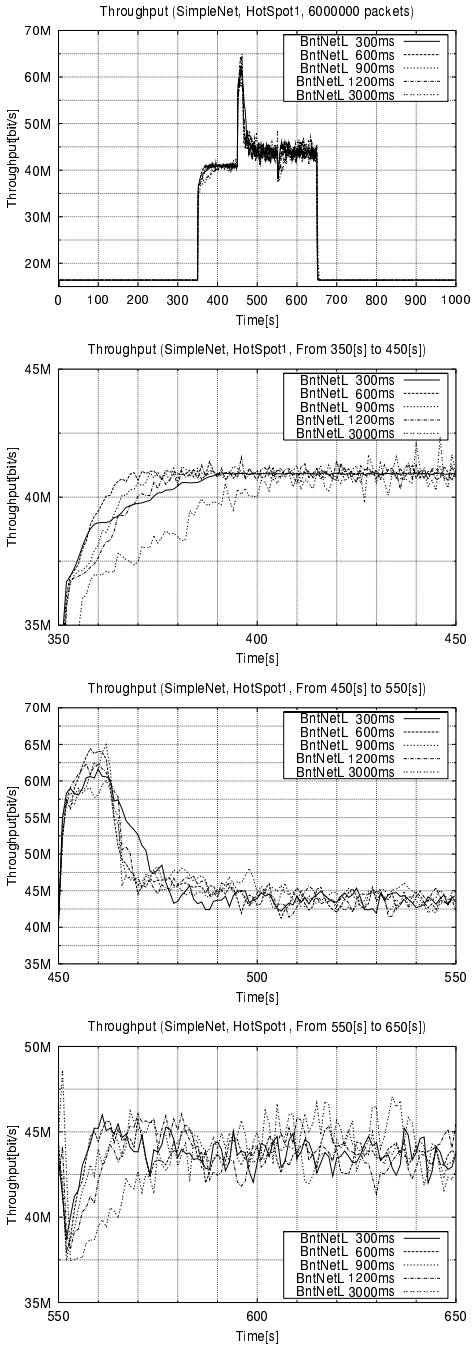

図 12: HotSpot1::Ant の生起間隔を変更した場合のスループット (上から 0[s] ～ 1000[s], 350[s] ～ 450[s], 450[s] ～ 550[s], 550[s] ～ 650[s])

している経路を避けて 6 まで辿りつけるが，HotSpot2 では 1 で Ant が進む時点では $7 \leftrightarrow 8$ が輻輳していることは検知できない．このため，1 から 8 に Ant が向かってしまう．このためにルーティングテーブルの更新が遅れてしまい，輻輳している経路にパケットを送信してしまうことが考えられる．また，一方で表 9 では BntNetL と BntNetV との平均伝送時間の差が小さくなっていることがわかる．

図 14 での 350[s] ～ 450[s] および 450[s] ～ 550[s] では，BntNetL および BntNetV のどちらもスループットの上昇の傾向は同じであることがわかる．特に，450[s] ～ 550[s] においては，$1 \rightarrow 3 \rightarrow 5$ は 1 から 5 へ向かうための最短パスであることから，450[s] の直前では $P^1(3,5)$ は高い確率になっている．このときスタブが発

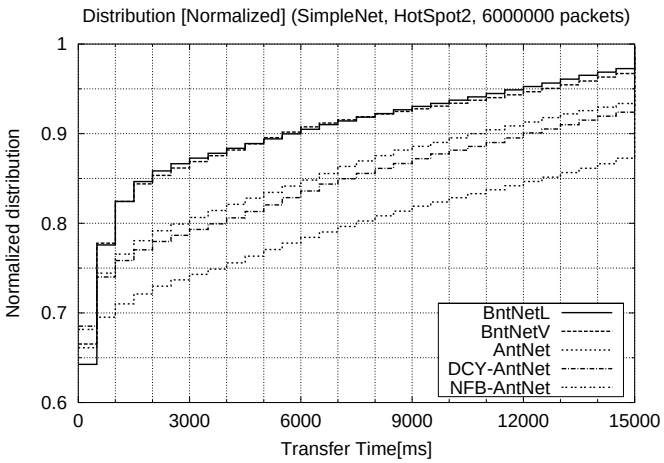

図 13: HotSpot2::パケット伝送時間分布 (正規化)

表 10: シミュレーションパラメータ (NSFNET)

トポロジ	NSFNET
平均パケット長	512 [Byte]
リンク伝送速度	1.5[Mbit/s]
最大パケット長	1500[Bytes]

生していることになるが，スループットの振る舞いに大きな差は見られないことがわかる．さて，図 14 での 550[s] ～ 650[s] に関するグラフを見ると，BntNetV ではスループットは一旦 37.5[Mbit/s] まで落ち込み，徐々に上昇していく振る舞いが観測されている．BntNetL では一旦 40[Mbit/s] まで落ち込むものの，すぐに復帰している．これは，Loop-Free の作用により，4 で生成された Ant が 2 に向かったときに，BntNetV では Ant はまた 4 に戻ってしまう可能性が高いと考えられる．$3 \leftrightarrow 5$ の HotSpot が生成されたときに，$3 \leftrightarrow 1 \leftrightarrow 2 \leftrightarrow 4 \leftrightarrow 5$ という経路を学習するため， 2 から 5 に向かうためには 4 に行く，ということを学習しているためと考えられる．一方で，BntNetL は Ant は 1 に向かうことができる．この差によりスループットの上昇の早さが異なると考えられる．すなわち，HotSpot1 と HotSpot2 の実験を通して，図 2 において，s に相当するノードが 2 つ以上ある場合に Loop-Free アルゴリズムが有効であることが言える．

7 実験: NSFNET

シミュレーション実行環境として，図 15 に示す NSFNET [21] を用いる．各リンクは全二重，それぞれの伝送速度は 1.5 [Mbit/s] であり，伝搬遅延として 図 15 の各辺の上に記している値 (単位は [ms]) を用いる．NSFNET におけるシミュレーションパラメータとして表 10 を用いる．また，前学習時間で与えるトラフィックは，各ノードが全部のノードに向けて 512[Bytes] のパケットを 1 つパケットを送信するものとする．パケット長は負の指数分布に従い，パケット到着間隔はポアソン分布に従う．パケットの行き先は一様分布に従うものとし [12]，その他のパラメータは第 5 章で示したものに従う．

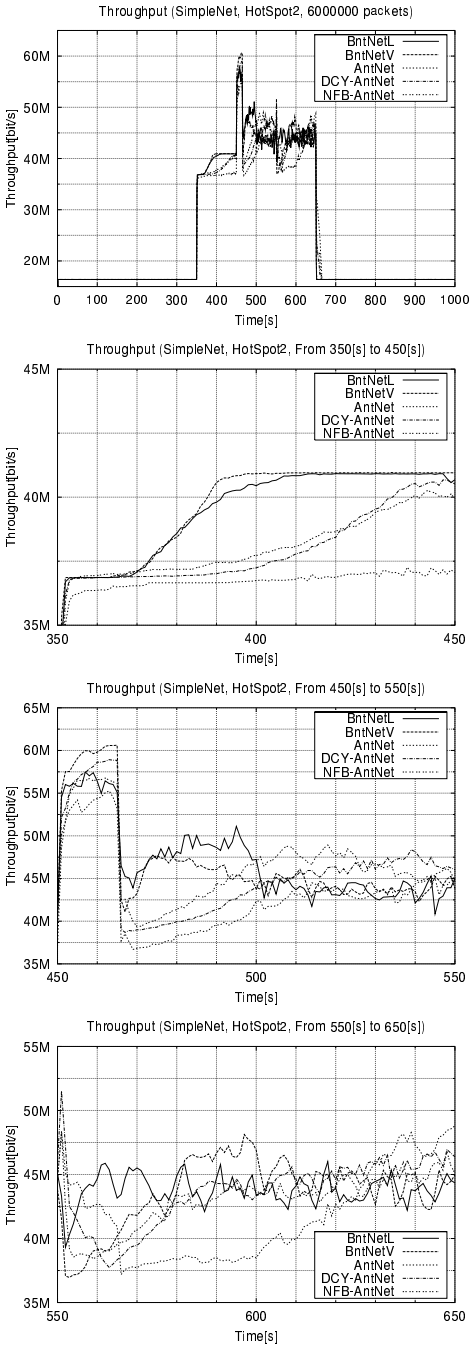

図 14: HotSpot2::スループット拡大図 (上から 0[s] ～ 1000[s], 350[s] ～ 450[s], 450[s] ～ 550[s], 550[s] ～ 650[s])

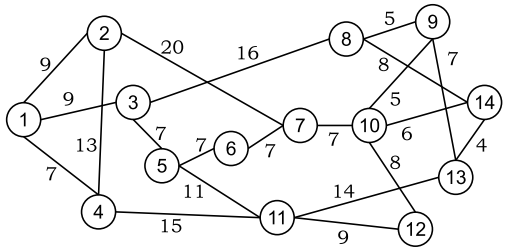

図 15: NSFNET と伝搬遅延 [ms]

7.1 実験目的

NSFNET [21] はアメリカ合衆国において存在した実際のネットワークである．このトポロジに各アルゴリズムを適用し，ネットワークのある一部に負荷が集中

表 11: 定常トラフィック時における統計量

生起間隔	項目	BntNetL	AntNet	DCY-AntNet	NFB-AntNet
10ms	平均	29.7	**28.9**	29.1	29.1
	分散	242.5	**213.6**	221.6	222.3
	最長時間	267.0	**196.4**	468.8	601.1
	損失率	0.003	0.003	0.003	0.003
5ms	平均	34.7	**32.5**	32.8	33.4
	分散	405.0	**309.0**	316.7	359.3
	最長時間	487.7	488.3	**474.2**	567.6
	損失率	0.003	0.003	0.003	0.003
3ms	平均	75.4	**71.7**	75.8	3788.6
	分散 ($\times 10^3$)	**4.344**	15.934	14.729	17226.163
	最長時間	**1432.0**	6872.5	10937.3	15027.6
	損失率	0.007	0.007	0.007	16.032
2.5ms	平均	1243.1	**611.8**	6044.0	6640.4
	分散 ($\times 10^6$)	1.425	**1.168**	17.919	16.924
	最長時間	**15002.0**	15027.4	15027.0	15027.4
	損失率	0.142	0.061	27.164	27.993

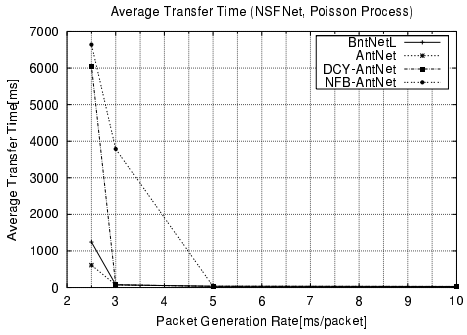

図 16: ポアソン過程下での平均伝送時間

してしまうような場合に，各アルゴリズムがどのように適応できるかを観測することを目的とする．

7.2 定常トラフィック

各ノードが独立に平均 10[ms]，5[ms]，3[ms]，2.5[ms] 間隔でトラフィックを生成した場合の平均伝送時間に関する平均および分散，最長伝送時間を表 11 に示す．

全体を通して，ネットワーク全体が輻輳した場合にも平均伝送時間を短くできているのは AntNet であることがわかる．BntNetL は AntNet よりは性能は劣るが，DCY-AntNet や NFB-AntNet よりは平均伝送時間を短くできている．BntNetL において平均伝送時間が AntNet よりも悪くなるのは，混んでいるルートを誤って強化してしまう可能性があることによると考えられる．これは，F-CBR と F-CBR2 の比較において，片方向性のトラフィックは双方向性のトラフィックより性能が落ちるのと同じ理由である．F-CBR などよりも大きなトラフィックが生じているために，性能の減少の度合いが F-CBR よりも大きく現れる例と言える．

NFB-AntNet において パケットの生起間隔が 3[ms] 以上の場合に性能が悪いのは，前学習時間で無トラフィック状態で学習を行わせ，学習終了後にトラフィックが急増する現象が発生し，どのリンクを選んでも負のフィードバックを与えられる現象に陥ってしまったためと考えられる．

DCY-AntNet において パケットの生起間隔が 2.5[ms] の場合にスループットが落ち込む現象は，学習終了後にトラフィックが与えられると，Ant がネットワークを巡回してルーティングテーブルを更新するとき，学習前の履歴が残っていること，およびトラフィック入力が非常に大きく Ant の移動に関して前学習時間よりも時間

表 12: ノード 2 → ノード 7 のトラフィック密度が増加した場合のパケット伝送時間に関する統計量

	平均	分散	最長時間	損失率
BntNetL	**35.1**	**1000.8**	**1011.2**	0.003
AntNet	47.5	23015.6	3334.8	0.003
DCY-AntNet	43.7	15184.5	3390.3	0.003
NFB-AntNet	44.2	14943.3	3237.8	0.007

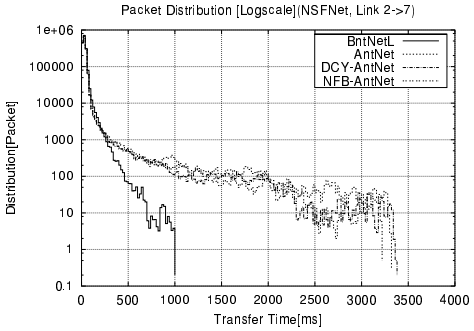

図 17: ノード 2 → ノード 7 のトラフィック密度が増加した場合のパケット伝送時間分布 (対数)

がかかることが考えられる．従って，実際には Ant が移動した経路に関してはルーティングテーブルはほとんど強化されない．一方で，一定時間毎に均等にルーティングテーブルの確率を均等に近づけていくアルゴリズムが実行される．このため，ネットワーク全体でランダムにルーティングを行ってしまい，スループットが落ち込むと考えられる．

7.3 局所ノード間の輻輳

ノード間の輻輳に対する各アルゴリズムの適応能力を見る目的で，時刻 0[s] ～ 1000[s] では各ノードは平均トラフィック密度が 10[ms/packet] の間隔でパケットを生成する．時刻 350[s] ～ 650[s] の間において，ノード 2 については，それとは独立にノード 7 に向けて 2.5[ms/packet] の間隔でパケットを生成する．これは実際のトラフィックではノード間で大きなファイルを転送するようなトラフィックが生じることを意味する．実験結果を表 12 ，図 17，図 18 に示す．

図 18 のスループットの全体像を見る限りでは，どのアルゴリズムも長いレンジで見た場合にトラフィックの変化に追従できていることがいえるが，追従の速さの観点で図 18 での拡大図を見ると，BntNetL がトラフィック密度が増加する時刻 350[s] ～ 400[s] において，スループットの立ち上がりが他のアルゴリズムと比較して早いことが言える．すなわち，それだけ早く 2 から 7 へパケットを分散して配送できることを意味し，輻輳に対する適応能力が高いことが言える．

図 17 および表 12 を見ると，DCY-AntNet，NFB-AntNet において AntNet よりも平均，分散に関して良くなっているが，BntNetL が平均，分散，最長到達時間のどの値を見てももっとも良くなっていることがわかる．他のアルゴリズムと比較すると，平均で約 10[ms]，最長到達時間で約 2[s] 良くなっている．これから，特定ノード間のトラフィックが増加した場合に BntNetL は有効であるといえる．

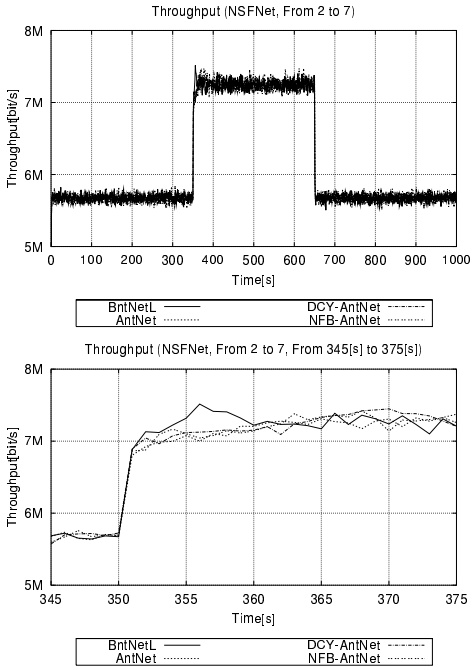

図 18: ノード 2 → ノード 7 のトラフィック密度が増加した場合のスループット (上から，全体像，立ち上がり拡大)

表 13: ノード 11 からのトラフィック密度が増加した場合のパケット伝送時間に関する統計量

	平均	分散	最長時間	損失率
BntNetL	**186.3**	**144127.3**	**8246.7**	0.002
AntNet	346.2	672247.3	14276.0	0.002
DCY-AntNet	355.2	643737.7	14583.5	0.002
NFB-AntNet	795.0	3514675.0	15012.7	0.022

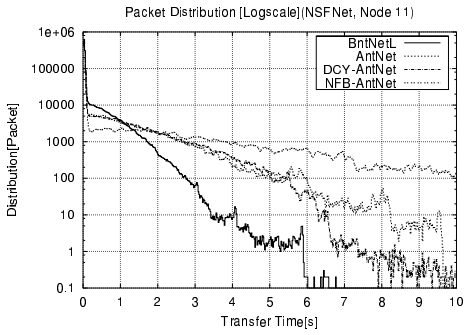

図 19: ノード 11 からのトラフィック密度が増加した場合のパケット伝送時間分布 (対数)

7.4 ノードの輻輳

ノードの輻輳に対する各アルゴリズムの適応能力を見る目的で，時刻 0[s] ～ 1000[s] では各ノードは平均トラフィック密度が 10[ms/packet] の間隔でパケットを生成し，時刻 350[s] ～ 650[s] の間において，ノード 11 に関してはランダムに選ばれたノードに向けて 0.6[ms/packet] 間隔でパケットを生成する．実際のトラフィックでは，ある http サーバや anonymous ftp サーバに負荷が集中した場合に相当する．実験結果を表 13，図 19，図 20 に示す．

図 20 での全体像を見ると，どのアルゴリズムも全体的なスループットには大きな差はないが，図 20 での立ち

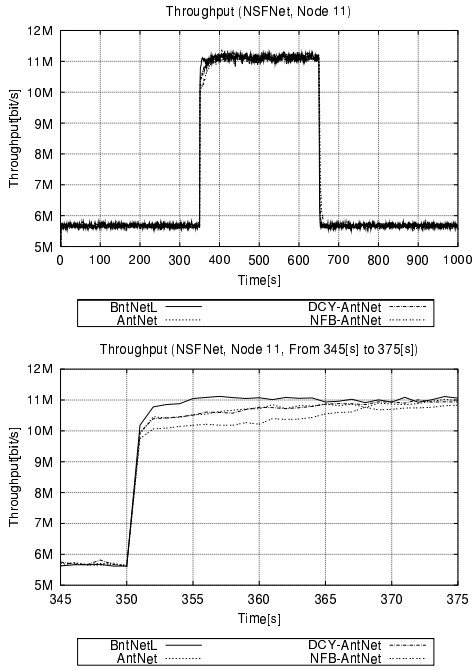

図 20: ノード 11 からのトラフィック密度が増加した場合のスループット (上から，全体像，立ち上がり拡大)

表 14: 全体のトラフィック密度が増加した場合のパケット伝送時間に関する統計量

	平均	分散	最長時間	損失率
BntNetL	55.7	**2981.4**	**1174.0**	0.002
AntNet	**53.7**	12665.3	3940.5	0.002
DCY-AntNet	57.5	17162.2	9461.8	0.002
NFB-AntNet	3009.2	17832485.8	15026.9	15.688

上がり時のスループットの振る舞いを見ると，BntNetL はスループットの立ち上がりが早く，ネットワークへのパケットの入力量が突然多くなっても，それに素早く追従できる能力を有することがわかる．このことは表 13 でも示されており，BntNetL は他のアルゴリズムと比較して，平均で約 150[ms]，最長到達時間で約 6[s] 短く，性能が一番いいことがわかる．

7.5 全体のトラフィック密度の変化

ネットワーク全体のトラフィック密度の変化に対して各アルゴリズムがどのように振る舞うかを見ることを目的で，時刻 0[s] ～ 350[s] および 650[s] ～ 1000[s] では各ノードは平均トラフィック密度が 10[ms/packet] の間隔でパケットを生成し，時刻 350[s] ～ 650[s] の間 (以下 HotSpot と表記) において，各ノードの平均トラフィック密度を 3[ms/packet] と増加させた場合の振る舞いを見る．実験結果を表 14，図 21，図 22 に示す．

図 22 での全体像から NFB-AntNet では HotSpot においてスループットが一時は上昇するが，その後徐々に減少していることがわかる．これは，ネットワーク全体のトラフィック密度が急増したことにより，Ant がどのリンクを選んでも負のフィードバックが与えられてしまう現象が生じ，これにより一部のパケットが別の経路を利用して配送されることになるが，これによって別のパケットの送信を阻害してしまっていると考えられる．このことは伝送時間の分布にも表れている．すなわち NFB-AntNet はネットワーク全体の急激なトラ

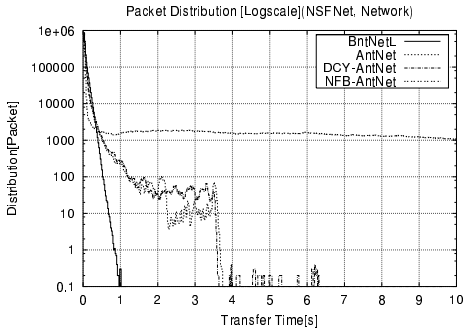

図 21: 全体のトラフィック密度が増加した場合のパケット伝送時間分布 (対数)

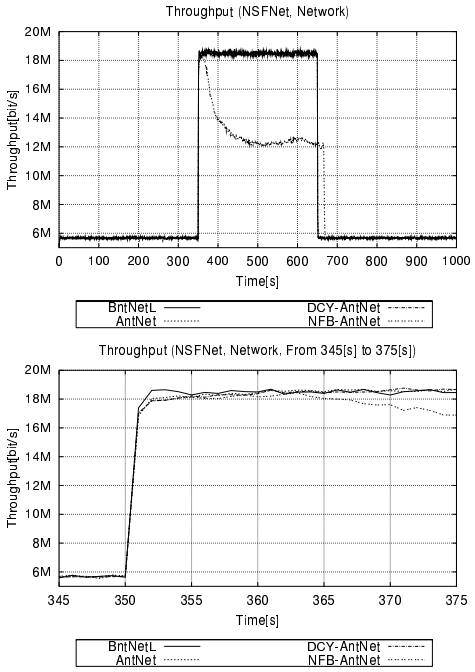

図 22: 全体のトラフィック密度が増加した場合のスループット (上から，全体像，立ち上がり拡大)

フィック密度の変化に追従する能力を有さないことが言え，これは全体的にトラフィック密度が増加するのを，局所的にトラフィック密度が増加するものとして負の報酬を与えることに起因する．

その他のアルゴリズムにおいては，図 22 から，スループットにおいてはどのアルゴリズムもほとんど差は見られないが，表 14 および 図 21 から，平均以外の項目では BntNetL ではその他のアルゴリズムよりも良くなっていることが言える．このことから，BntNetL はネットワーク全体のトラフィック密度の変化に対して均質なサービスを提供できる能力を有することが言える．

8 今後の課題

本論文では，ベストエフォート型のアルゴリズムについて比較実験を行ってきた．今後の課題として，次のようなことがあげられる．まず，BntNetL における Ant の双方向性と単方向性の切り替えを行うことが考えられる．NSFNET での実験で，平均パケット生起間隔が一定の場合には BntNetL はむしろパフォーマンスが劣ってしまうことが示されているため，このための枠組みを作ることが考えられる．また，Ant の生成間隔を動的に切り替える試みが考えられるが，[22] では，ネットワーク全体の Ant の最大数を決定するというアプローチのため，ネットワーク全体を見渡せるような構造が必要となり，AntNet の利点となる，分散制御環境でも適用可能という利点がなくなってしまう．これから， Ant によるルーティングテーブルの更新間隔を，各ノードの Ant によるルーティングテーブル更新間隔に依存して生成するようなアルゴリズムが考えられる．

これからのネットワークの発展により，ネットワーク層においてもパケットの到着順序が保証されるようなプロトコル，例えば動画や音声の実時間転送に関して資源予約を行う RSVP(Resource ReSerVetion Protocol)[23] などとの協調もとる必要があると考えられる．[23] では，資源予約を行う段階で各ルータに対して，帯域幅等の情報が伝達されることから，ルータに帯域毎にグルーピングされた待ち行列を設ける [24] などの作業により，Ant やベストエフォートで問題ないパケットに関しては，リアルタイム転送を要求されるパケットよりも優先度を低くすることにより，実現は比較的容易であろう．これと関連して，QoS 制御のエージェントベースのルーティングアルゴリズムも提案されている [25].

将来的には，Ant のサイズも無視できない問題になる．これは TCP/IP ではフラグメントが避けられないためであり，AntNet を大規模ネットワークの実問題に適用することを考えた場合に生じると考えられる．このフラグメントによって再び各ノードで Ant をデフラグメントするために必要なコストに対する考察も必要である．上記とも関連するが，IPv6 [26] 上では，IP アドレス長が 128[bit] になり，現在の IP(IPv4) プロトコル (32[Bit]) の 4 倍となる．これにともない，Ant のフラグメントの問題や Ant のオーバーヘッドが大きくなることが懸念されるが，Ant のオーバーヘッドに関しては，通信インフラストラクチャの整備により解決できる問題であると思われる．

9 おわりに

BntNetL や AntNet などに代表されるようなマルチエージェント系による経路制御アルゴリズムの利点として挙げられることの一つとして，RIP や SPF などと比較して通信量コストが少なくて済むことである [16]．本論文では，AntNet に双方向性を設けることとループを形成しないように Ant を動かすという制約を設ける，という手法を用いた BntNetL を提案し，AntNet とその発展型アルゴリズムである DCY-AntNet，NFB-AntNet との比較実験を行った．以上の実験から，BntNetL は以下の特徴を有することが言える．

1. ビットレートが固定の静的トラフィックでは AntNet よりも優れる．
2. 局所的に輻輳が起きる場合に，AntNet よりも優れる．
3. Loop-Free の効果は，図 2 における s が 2 つ以上の時に大きくなる．
4. パケットの平均生起間隔が一定のポアソン過程では AntNet よりもパケットの平均伝送時間で劣る．

BntNetL の利点をまとめると， AntNet などのアルゴリズムに比較して入力トラフィックの変動に対しても

安定してその増加に追従できることである，といえる．

口頭発表

土居 茂雄，山村 雅幸: BntNet によるネットワーク経路制御の提案，計測自動制御学会 システム/情報合同シンポジウム 講演論文集，pp.215-220, 1999, 高知．

謝辞

本研究に対して，多大なるご指導をいただいた山村雅幸助教授および山村研究室の皆様に感謝申し上げます．また，論文についてわからないところを伺ったところ，直接会う機会を設けて下さり， AntNet についてほとんど前提知識のない小生に対して，懇切丁寧にご教授いただいた AntNet アルゴリズムの開発者である ATR 人間情報通信研究所 (当時 IRIDIA, Université Libre de Bluxelles) Gianni Di Caro 氏に感謝の意を表します．最後に，学生生活を支えてくれた両親ならびに友人たちに感謝します．

参考文献

[1] Gianni D. Caro and Marco Dorigo. An adaptive multi-agent routing algorithm inspired by ants behaviour. *Proceedings of PART'98 - fifth Annual Australasian Conference on Parallel and Real-Time Systems*, 1998.

[2] Gianni D. Caro and Marco Dorigo. Antnet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research (JAIR)*, Vol. 9, pp. 317–365, 1998.

[3] Gianni D. Caro and Marco Dorigo. Two ant colony algorithms for best-effort routing in datagram networks. *Proceedings of PDCS'98 - 10th International Conference on Parallel and Distributed Computing and Systems*, 1998.

[4] Marco Dorigo and Luca M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp. 53–66, 1997.

[5] Marco Dorigo and Gianni Di Caro. Ant colony optimization: A new meta-heuristic. *Proceedings of the 1999 Congress on Evolutionary Computation*, Vol. 2, pp. 1470–1477, 1999.

[6] Marco Dorigo, Gianni Di Caro, and Luca M. Gambardella. Ant algorithms for discrete optimization. *Artificial Life*, Vol. 5, No. 2, 1999.

[7] John Moy. *RFC 2178, OSPF Specification Version 2.* Network Working Group, July 1997.

[8] Justin A. Boyan and Michael L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. *In Advances in Neural Information Processing Systems*, pp. 671–678, 1994.

[9] Samuel P. M. Choi and Dit-Yan Yeung. Predictive q-routing: A memory-based reinforcement learning approach to adaptive traffic control. *In Advances in Neural Information Processing Systems 8 (NIPS8)*, pp. 945–951, 1996.

[10] Martin Heusse, Dominique Snyers, Sylvain Guérin, and Pascale Kuntz. Adaptive agent-driven routing and load balancing in communication networks. *ENST de Bretagne Technical Report (accepted for publication in the Journal of Complex Systems)*, 1998.

[11] Gianni Di Caro and Marco Dorigo. Antnet: A mobile agents approach to adaptive routing. *Technical Report IRIDIA 97-12*, 1997.

[12] 種田和正, 片岡明. ロックフリー antnet とその適応能力の評価. 電子情報通信学会論文誌, Vol. J82-B No.7, pp. 1309–1319, 1999.

[13] Ruud Schoonderwoerd, Owen Holland, Janet Bruten, and Leon Rothkrantz. Ant-based load balancing in telecommunications networks. *Adaptive Behavior*, pp. 169–207, 1996.

[14] Devika Subramanian, Peter Druschel, and Johnny Chen. Ants and reinforcement learning: A case study in routing in dynamic networks. *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 832–838, 1997.

[15] Eric Bonabeau, Florian Henaux, Sylvain Guérin, Dominique Snyers, Pascale Kuntz, and Guy Theraulaz. Routing in telecommunications networks with ant-like agents. *Proceedings of Intelligent Agent for Telecommunications Applications*, pp. 60–71, 1998.

[16] Masaharu Munetomo, Yoshiaki Takai, and Yoshiharu Sato. An adaptive network routing algorithm employing path genetic operators. *Proceedings of International Conference on Genetic Algorithms*, 1997.

[17] 土居茂雄, 山村雅幸. Bntnet によるネットワーク経路制御の提案. 計測自動制御学会 システム/情報合同シンポジウム 講演論文集, pp. 215–220, 1999.

[18] 箱田多美, 銭飛. 強化学習を取り入れた新しいループフリールーティングアルゴリズム. 平成 11 年 電気学会 電子・情報・システム部門大会講演論文集, pp. 709–710, 1999.

[19] Bernd Bullnheimer, Richard F. Hartl, and Christine Strauss. An improved ant system algorithm for the vehicle routing problem. *Sixth Viennese workshop on Optimal Control, Dynamic Games, Nonliner Dynamics and Adaptive Systems*, 1997.

[20] Douglas Comer, 村井純, 楠本博之. 第 3 版 TCP/IP によるネットワーク構築 - 原理，プロトコル，アーキテクチャ -. 共立出版, September 1999.

[21] Kimberly C. Claffy, George C. Polyzos, and Hans-Werner Braun. Traffic characteristics of the t1 nsfnet backbone. *Proceedings of INFOCOM 1993*, 1993.

[22] Marcos Gallego-Schmid. Modified antnet: Software application in the evaluation and management of a telecommunication network. *Proceedings of the 1999 Genetic and Evolutionary Computation Conference Workshop Program*, pp. 353–354, 1999.

[23] Bob Braden, Lixia Zhang, Steve Berson, Shai Herzog, and Sugih Jamin. *RFC 2205, Resource ReSerVation Protocol Version 1 Functional Specification.* Network Working Group, September 1997.

[24] 中里秀則, 石田寛史, 門洋一. Ip ネットワークの通信品質制御. 沖電気研究開発 181, Vol. 66, pp. 63–66, 10 1999.

[25] Kazumasa Oida and Masatoshi Sekido. An agent-based routing system for qos guarantees. *IEEE International Conference on Systems, Man, and Cybernetics Abstracts*, p. 273, 10 1999.

[26] ネットテクノロジーラボ. 最新技術概説 入門 IPv6. 技術評論社, 1999.