トポロジを考慮しソース選択を行うデータ転送スケジュラー

高橋 慧[†] 田浦 健次朗[†] 近山 隆[†]

本稿では，トポロジ情報を用いて転送時のリンクの共有を回避することで，各ノードに到着するデータのスループットを向上させるデータ転送スケジュラーを提案する．自然言語処理などのデータ・インテンシブなアプリケーションでは，全体の実行時間に占めるデータ転送時間が大きいので，効率的なデータ転送が重要になる．例えば N 本のデータ転送が 1 本のリンクを共有すると，それぞれの転送の速度は最悪で 1/N と大幅に低下してしまう．もしデータのソースが複数あり，トポロジ情報を用いてこれらを適切に選択すれば，このリンクの共有による転送速度の低下を回避出来る．しかし既存のデータ転送スケジュラーは，トポロジを考慮せずノード間で一定なバンド幅を仮定しており，このリンクの共有を検知することができない．

そこで我々はトポロジ情報を用い，あるノードが必要とするデータが複数のソースにある場合は，最も大きなバンド幅で転送できるソースを選択する．複数のノードが同一のソースを用いる場合には，それらのノードを一列に並べてパイプライン転送を行う．また，あるリンクを複数の転送が共有しているとき，多くのノードが必要としているデータの転送は重要であると考えられるので，より多くのバンド幅を与えるような最適化を線形計画法を用いて行う．これらの工夫によって，全体のファイル転送のスループットが向上する．この転送計画アルゴリズムを実装し，実験を行った．

A Data Transfer Scheduler Selecting Sources by Using Topology Information

Kei Takahashi,[†] Kenjiro Taura[†] and Takashi Chikayama[†]

We propose a data transfer scheduler avoiding link sharing among transfers and trying to maximize throughput arriving at each node. Data transfer scheduling is especially important in the execution of data intensive applications, such as natural language processing, since data transfer takes much of their execution time. For instance, when N data transfers share one link, each transfer speed decays down to 1/N. If data have multiple replicas and if each node selects data source from them by using network topology information, this decay may be avoided. However, existing data transfer schedulers do not consider network topology, and only assume static bandwidth among nodes. Thus, the decay of transfer speed cannot be detected.

We plan efficient data transfer scheduling by using network topology and bandwidth information. When data have multiple replicas, a replica reached with the maximum bandwidth is selected. When multiple nodes need data located at one source, the data are multicasted in a pipeline fashion. In addition, we actively control each transfer bandwidth: when multiple multicast transfers share a link, more bandwidth is given to a multicast transfer with more nodes. Consequently, the total throughput is improved. We implemented and evaluated this transfer algorithm.

1. はじめに

コンピュータの性能向上，特に記憶容量の飛躍的な増加によって，これまで望めなかった大規模な処理が可能になっている．例としては，Web 上のドキュメントに対する言語処理・遺伝子解析などが挙げられる．これらは処理自体は単純であっても，大規模なデータに対して処理を行うことで，これまで得ることが出来なかった成果を上げている．これらをデータ・インテンシブなアプリケーションと呼び，データサイズはテラバイトからペタバイトに達するので，並列処理が不可欠になっている．

ここで，データ・インテンシブなアプリケーションの実行においては，データの転送時間が全体の処理時間に対し無視できない割合を占める．分散環境ではネットワークが不均質であり，任意の 2 ノード間のバンド幅は大きく異なるので，転送計画によって必要な時間は大きく異なる．また複数のノードに大きなデータを転送する場合，1 本のリンクを多数の転送が使用する

† 東京大学
The University of Tokyo

ことで転送のバンド幅が大きく低下してしまう可能性がある．このような様々な要素を考慮して賢い転送スケジューリングを行うには，トポロジを使わないアプローチには限界がある．

一方で，近年大規模ネットワークの測定技術は進歩しており，多くのスイッチやノードから構成されたネットワーク・トポロジを end-to-end の測定で推定する技術が確立されている．そこで我々は，トポロジ情報を用いて複数の転送を競合を避けるように計画し，各転送先ノードに到着するデータの合計スループットを最大化するようなデータ転送計画アルゴリズムを提案する．対象とするのは，複数のデータに対し，複数の転送元にコピーが置かれており，さらに複数のノードがこれを必要としているような問題である．ここで，提案手法では転送にパイプラインを用い，1 種類のデータ転送は同じリンクを一度しか用いない．また，複数の転送が存在することによるリンクの競合を回避するよう，他の転送の存在を考慮した上での転送の再計画を行う．また，あるリンクを複数の転送が共有しているとき，多くのノードが必要としているデータの転送は重要であると考えられるので，より多くのバンド幅を与えるような最適化を線形計画法を用いて行う．これらの工夫によって，多数のノード間で同時に多数の転送を行う際に，スループットを大きくするような転送計画を短時間で立てることができる．この転送計画アルゴリズムを実装し，シミュレーションによる評価を行った．

本論文の構成は以下の通りである．2 章では背景や関連研究について説明した後，3 章で問題の定式化を行う．ここで示した問題は付録に示す通り NP 困難であるが，1 種類の転送について近似解を与えるアルゴリズムについて 4 章で説明し，さらに複数の転送に対する計画を行うアルゴリズムを 5 で与える．6 ではシミュレーションを用いた評価結果を示し，最後にまとめと今後の課題を示す．

2. 関 連 研 究

同一のデータを 1 つのノードから多数のノードに効率よくマルチキャストする研究はこれまでに多くされているが，多種類のデータのマルチキャストに注目したアルゴリズムは多くない．ここでは，1 種類のデータのマルチキャストについて、厳密なトポロジ情報を用いずに end-to-end で測定したバンド幅のみを用いる方法，トポロジを用いてリンクの共有を避ける方法，及び BitTorrent に代表される分散アルゴリズムを用いて動的に転送経路を構成する手法の 3 つを取り上げる．

2.1 ツリーを利用したマルチキャスト

最も簡単なマルチキャストの方法は，転送元ノードが全転送先ノードに対し直接データを転送する，flat-tree と呼ばれる手法である．この手法は実装が単純だが，転送元ノードの末端リンクがボトルネックとなる可能性が高い．転送時間はノード数を N, メッセージサイズを S とすると，$O(S \cdot N)$ となりノード数とサイズに比例するため，ノード数が増えると貧弱な性能となってしまう．

この転送元ノードのボトルネックは，転送先ノード間でツリーを構築して，転送の中継を行うことで解消できる．[1)]MPICH-G2 では，ノード間の距離の情報を用いて転送元から近い順にデータを中継していくようなツリーを構築している．この手法での大きなメッセージの転送時間は $O(S \cdot \log N)$ となり，flat-tree に比べ改善が図られた．しかし，この手法はリンクの共有による影響を考慮しておらず，実際に大きなメッセージを送ると性能が低下してしまうものと思われる．

2.2 Balanced Multicasting

[4)]Den Burger らは，マルチキャストツリーの構築に際して，Balanced Multicasting という手法を提案している．まずクラスタ間とクラスタ内のリンクを分けて考え，クラスタ間のネットワークトポロジとそのバンド幅を利用し，様々なマルチキャストツリーの候補を生成する．それらの候補の重ね合わせを考え，各ノードに到着するスループットを最大化するよう線形計画法を解き，各ツリーの係数を求める．これにより，マルチキャストを行うことができる．

このアルゴリズムはクラスタ間において厳密にリンク容量を考慮していることから，リンクの共有を考慮したマルチキャストの計画を行うことが出来る．一方で，考慮しているトポロジはクラスタ間のみであり，クラスタ内のリンクについては考慮されていない．これは，全てのリンクについて本アルゴリズムを適応すると，計算量が爆発してしまうためである．この制限は各ノードに 1 つしか転送が存在しない場合には問題が無いが，複数のデータが同時に転送される場合にはクラスタ内のリンクも考慮する必要がある．

2.3 トポロジを考慮したパイプライン・マルチキャスト

[3)] 白井らは，高速にトポロジを推定する手法と共に，トポロジ情報を用いたマルチキャスト・アルゴリズムを提案している．実用上多くの場合，ネットワークトポロジは図 1 のような閉路を持たないツリー構造で表される．このツリーはノードとスイッチで構成されて

おり，ノードは必ず一本のみのリンクを持つものとする．ここで，転送元 (source) から順に転送先のノードを一直線につないだパイプラインを構成すると，転送時間の見積もりはデータサイズを $size$, パイプライン中の最小のバンド幅のリンクを $min(links)$ とした場合，$(\frac{size}{min(links)} + \sum rtt)$ となる．データが十分に大きい場合，転送時間はノード数によらず，データサイズだけに比例する．また，パイプラインの転送速度は最も細いリンクに制約される．

ここで最大のバンド幅を持つパイプラインは，各ノード・スイッチを深さ優先でなぞってノードを並べることで求められる．ツリーの指定された全てのノードを接続する木は必ず全てのリンクを通るので，パイプラインの最大スループットは最小のバンド幅を持つリンクを超えることができない．ここで先ほど求められたパイプラインでは，各リンクが上り下り共に一回ずつ用いられるので，上り下りの転送が干渉しない場合，最小のバンド幅のリンクの値がそのままパイプラインのバンド幅になる．

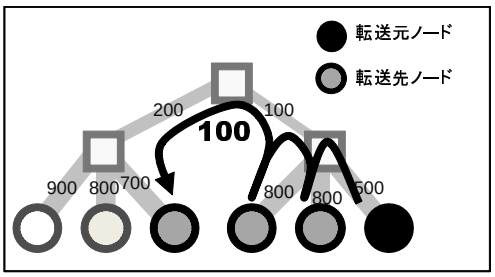

図 1 パイプラインマルチキャストの例
Fig. 1 An example of pipeline multicasing

2.4 BitTorrent 的アプローチ

[2)]BitTorrent に代表されるいくつかの P2P ソフトでは，ネットワーク状況を推定すること無しに，動的に転送元を選択して適応的なマルチキャストを行うことができる．ここでは代表例として BitTorrent のアルゴリズムを説明する．

まずデータは断片化して送受信し，各断片は異なったノードから受信できる．ノードの受信元は元々データを持っていたノードとは限らず，まだデータの一部しか受信していないようなノードも積極的に中継に関わる．転送に参加するノードは，現在転送に参加しているノードのリストを受け取り，その中からランダムに 5 つのノードを同時に選択し，それらに転送を依頼する．依頼されたノードは，依頼してきたノードのうち，他のノードに最も大きなスループットで転送しているノード 4 つを選択して転送を了承し，その他のノードからの依頼は拒否する．拒否されたノードは改めて別のノードに転送を依頼する．これによって，バンド幅が大きなノードが上位に，小さなノードが下位に位置するようなツリーが自動的に構築される．

BitTorrent の特徴は，予めトポロジやバンド幅などのネットワークの情報を収集する必要がなく，かつバンド幅が大きい順に並んだマルチキャストツリーが自然と構築されることである．複数の転送があっても，それぞれの転送は独立して計画できる．

しかし，クラスタ環境のようなある程度統制された環境下で，予め取得したトポロジを用いて最適なルートを計画できるのであれば，このような試行錯誤は単なるオーバーヘッドとなる．また BitTorrent の挙動は複雑で，ある転送群が与えられたときに終了時刻を予測することは大変難しい．

3. 問 題 設 定

3.1 与えられる情報

本研究では，予めネットワーク・トポロジ及び各リンクのバンド幅が与えられるものとする．ノード及びスイッチは有向リンクで接続され，それぞれにバンド幅が正の値として与えられている．このトポロジは実際の中継機器の数を厳密に反映している必要は無いが，ノードの分岐位置の情報は正しく保持しているものとする．これにより，2 つの転送がリンクを共有するかどうかを正しく予測することができる．

ここで転送計画を立てたいのは，次のような問題である．

(1) N 種類のデータ転送がある
(2) それぞれの転送に対し，データのコピーが n 個のノードに配置されている
(3) それぞれの転送に対し，m 個のノードがデータを必要としている

以下，データを持つノードを source, データを必要としているノードを destination と呼ぶ．各 destination はそれぞれ最適な source を 1 つ選択すれば完全なデータを得ることが出来る．単純化のため，1 つの destination が複数の source から同時にデータを受け取ることは出来ないものとする．

3.2 目標: 合計スループット最大化

転送計画の評価は様々な指標が考えられるが，本研究で設定した目標は，各転送先 (destination) ノードに一定時間当たり到着するデータのスループットを最大化することである．これを「合計スループット最大化問題」と名付ける．例えば図 2 の例では，合計スループットは 700 + 500 + 500 で 1700 となる．実現

可能な様々な転送計画のうち，合計スループットが最大になるような転送計画を立てるのが本研究の目標となる．合計スループットは，各 destination の末端リンクの合計値より大きくなることはできない．

合計スループット最大化問題は，最も簡単な転送数が 1 つのときでも NP 困難である．(証明は付録に示す) このため，厳密な最適解に到達するのは困難であり，近似アルゴリズムを用いることになる．

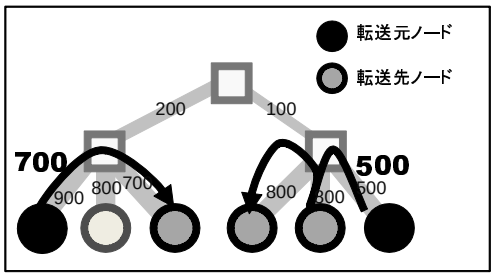

図 2 スループット最大化問題の例
Fig. 2 An example of the maximizing throughput problem

4. 複数ソースのマルチキャスト

本章では，1 転送のスループット最大化問題の近似解を与えるものとして，ボトルネックリンク最大化問題を解くアルゴリズムを示す．これにより，1 種類のデータが複数の source に存在し複数の destination がこれを必要としているとき，細いリンクを用いず，リンクの共有が発生しないパイプライン群を構築できる．

4.1 ボトルネックリンクの最大化

2.3 節に述べたパイプライン・マルチキャストを行う際，転送速度はパイプライン中の最も細いリンク (ボトルネックリンク) によって制約される．もし複数の転送元ノードが存在する場合，各転送先が大きなバンド幅で接続できる転送元ノードからデータを受信することにより，1 ノードからの転送に比べてパイプライン中のボトルネックリンクを向上できる．このように様々なパイプライン群が構築できる内で，ボトルネックリンクのバンド幅を最大化するような転送計画を求める問題を考える．これによりリンクの共有が発生せず，バンド幅が最大となるパイプライン転送を行うことが出来る．

ボトルネックリンクの最大化と合計スループット最大化は異なる目標関数であり，二つの最適解が等しくならない例も作ることが出来る．☆しかし，細いリンクの使用や転送の競合を回避できるという点では，ボトルネックリンク最大化によって計画された解はスループット最大化の候補となっていると考えられる．このボトルネックリンク最大化問題の解は，次に述べるように動的計画法で解くことが出来る．

☆ 例えばある末端ノードが極端に細いリンクでつながっている場合，スループットはこのノードを無視してパイプラインを構築した方が大きくなる．

4.2 用いるリンクの選択

グラフの最小木問題の解法であるプリムのアルゴリズム (Prim's Algorithm) を変形して用いると計画することができる．これを次に示す．

あるデータを多数の destination $\{d_i \in D\}$ が必要としており，複数の source $\{s_i \in S\}$ がそのデータのコピーを持っている．また，これらのノードとスイッチをつなぐ有向リンク群 $\{l_i \in L\}$ がある．また，A をソースから到達できるノード集合，L_a を追加されたリンク群とする．

リンク $l \in L$ をバンド幅が大きい順に調べ，もし l の起点が A に含まれていればそのリンクを L から取り除き，L_a に追加する．l の終点ノードが A に含まれていなければ A に追加し，もしこのノードが D に含まれていれば D から取り除く．このようなリンクの追加を続け，D が空になり，かつ全ての source から接続された destination を辿るようなパイプラインが構築可能になったら停止する．

以上のアルゴリズムを疑似コードで示すと以下のようになる．

origin(l):リンク l の始点
endpoint(l):リンク l の終点

$A = \emptyset$
$L_a = \emptyset$
$L =$ 全てのリンク

```
while True:
    if D = ∅ and pipeline_is_ready():
        break
    l = pick_link()
        L_a = L_a ∪ {l}
        A = A ∪ {endpoint(l)}
        D = D ∩ {endpoint(l)}


def pick_link():
    for descending order of bandwidth (l ∈ L) :
        if origin(l) ∈ A:
            L = L ∩ {l}
            return l
```

アルゴリズムのオーダーは，ノード数を N とするとき $O(N^2)$ となる．

4.3 パイプラインの構成

リンクを追加する仮定で，ある有向リンク群が与えられたとき，source から destination を一直線にたどるパイプラインが構成可能か判断する必要が生じるが，このアルゴリズムを説明する．

Source に接続されているノードはソースからのホップ数で半順序関係を付けることが出来る．あるノードからホップ数が大きい方向に移動してたどれるノードを「そのノード以下のノード」と呼ぶことにする．

各ノードに対し，次の 2 つの条件を考える．

A あるノードを始点として，そのノード以下の destination を全て一直線でたどって戻ることができる

B あるノードを始点として，そのノード以下の destination を全て一直線でたどることができる

ここで，パイプラインが構成可能な条件は，全ての source に対し B が成立することである．A は B よりも強い条件であり，A を満たせば B も満たす．また，あるノードが A, B を満たすかどうかは，そのノードを起点として次のように再帰的に判定していくことができる．

A を満たす条件 そのノードの子ノードのうち，Destination に至る子ノード全てが A を満たし，かつ双方向リンクでつながっている

B を満たす条件 そのノードの子ノードのうち，Destination に至る子ノードが以下を満たす

（1） たかだか一つは片方向リンクでつながっているか，B を満たす

（2） その他の子ノードは全て双方向リンクでつながっていて，A を満たす

このアルゴリズムによって，パイプラインが構成可能かどうかを判断することができると同時に，具体的にパイプラインを求めることが出来る．あるノードを通るパイプラインは，まず B を満たす子を全てたどって戻ってくる．その後 A のみを満たす子があれば，その子に向かう．source を起点としてこのなぞりを行うことにより，パイプラインを構築できる．

4.4 ボトルネックリンク最大の証明

こうして得られたパイプラインは，条件を満たすあらゆるパイプライン群の中で，最小のリンクのバンド幅が最も大きくなっている．これを以下に示す．

リンク集合 L_b が存在し，L_b 内の最小のバンド幅が

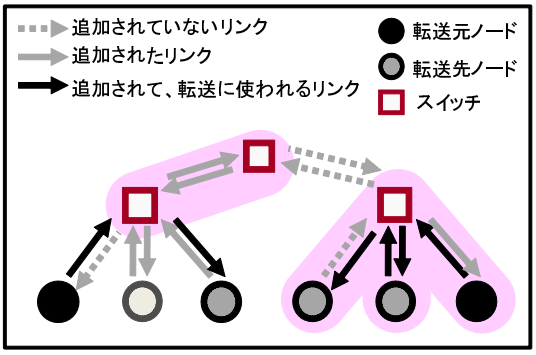

図 3 ボトルネックリンク最大化のアルゴリズム
Fig. 3 The algorithm to maximize the bottleneck

L_a 内の最小のバンド幅よりも大きかったとする．ここで L_a を構成する際，アルゴリズムが最後に追加した枝 l_i は，パイプラインに用いられているはずである．ここで l_i よりもバンド幅が大きく，かつ始点が source から辿れる枝は既に L_a に追加されているから，l_i を用いることなしにパイプラインが構成可能であるならば，その枝の始点は source から辿れないことになる．しかし，パイプラインに含まれる任意の枝は，定義より source からたどれるはずであるから，矛盾する．よってこのようなリンク集合 L_b は存在しない．

5. スループット最大化のアルゴリズム

本章では，複数の転送に対してスループットを最大化するような転送計画を立てるアルゴリズムを説明する．

まず基本的な考え方として，ある source から複数の destination にデータを転送する時，トポロジを幅優先に辿ったマルチキャストを行う (5.1 節)．1 転送の場合は，先のボトルネックリンク最大化問題を解くことでスループット最大化の良い近似解が得られるので，これを利用して各転送について独立にパイプラインを用いた転送計画を立てる．こうして全転送について，どの source からどの destination にデータを転送するかが求まったら，この計画を全て同時に実行する時に合計スループットを最大化するよう，線形計画法を用いて各転送に割り当てるバンド幅を決定する (5.2 節)．これによって初期解の転送計画が求まった．

こうして得られた初期解は，それぞれの転送を独立に計画しているので，ソースを変更することで合計スループットを改善できる可能性がある．そこで各転送について転送の再計画を行い，スループットが改善される場合はそのように変更する (5.3 節)．この再計画を順次行い，スループットが改善されなくなったら

停止し，最終的な転送計画として採用する．

以下，各フェーズについての詳細を説明する．

5.1 初期マルチキャストの計画

各転送について順番に 1 転送ずつ「ボトルネックリンク最大化」のアルゴリズムを用いて計画を立てる．用いるバンド幅は全て初めに与えられたもので，各転送は独立に計画される．これによって各転送についてリンクを上り下りたかだか 1 回しか使わず，かつ細いバンド幅のリンクを用いず複数の source が有効に使われるようなパイプラインが構築される．

5.2 線形計画法を用いた転送量の割り当て

データの転送計画が与えられたとき，各ノードに到着するスループットを最大化することを考える．例えばあるリンクを二つの転送が共有している際，片方は多数のノードが関わるマルチキャストで，片方が 1 ノードのみへの転送である場合，前者により多くのバンド幅を与えることで，全体のスループットは向上する．実際にはある転送のバンド幅は複数の制約によって定められているので，これを定式化した上で線形計画法を用いて最適化を行う．転送量の割り当ての例を図 4 に示す．

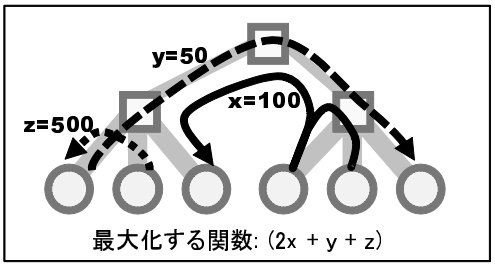

図 4 線形計画法によるバンド幅の割り当て
Fig. 4 Assignment of bandwidth by linear programming

整数 $\{1..l\}$ によって識別されるデータ転送が l 個あり，それぞれのバンド幅が $\{bw_i|1 \leq i \leq l\}$ であり，マルチキャストの到着ノードの数は $\{n_i|1 \leq i \leq l\}$ 個あるとする．このとき最適化したいのは，一定時間当たり各ノードに到着するデータ量であるから，$\sum_{k=0}^{l} bw_k * n_k$ と表すことができる．

ここで，バンド幅が c_i であるリンク L_i を通るデータ転送の集合を S_i とするとき，あるリンクを通過するデータの総量が c_i 以下であるという条件は，$\sum_{k \in S_i}^{l} bw_k \leq c_i$ と表すことができる．同様に全てのリンクについての条件を書き下し，線形計画法を用いると bw_l の値，あるいは関係が求まる．出力として値が定まらず等式が出力された場合は，これを満たすようなバンド幅を適当に定める．

線形計画法によるバンド幅最大化を行うと，特定のデータ転送にバンド幅が全く割り当てられず，そのタスクがずっと実行可能にならない，いわゆる飢餓問題が発生し得る．これを防ぐため，転送計画においては一定時間かかっても転送が開始されないファイルに対し一定のバンド幅を保証するような工夫が必要だと思われる．

5.3 パイプラインの再計画

こうして得られた転送計画は，各転送について合計スループット最大化の最適解を直接求めたわけではなく，しかも各データに対して独立に計画を立てているので，転送間でのリンクの共有による影響を考慮できていない．このため，各転送についてマルチキャストの再計画を行い，山登り法により最適解に近づけていく．

n 個のデータ転送が計画されているとして，i 番目のあるデータの転送を再計画する手順は以下の通りである．まず全体のバンド幅マップ (T_{all}) から，各種類のデータ転送計画において使われたバンド幅 $(T_i|0 \leq i < n)$ を全て差し引いたバンド幅マップ (T_{rest}) を作る．これは現在の計画で使われていない，各リンクのバンド幅を示している．これに現在そのデータ転送に割り当てられているバンド幅 T_i を加え，このバンド幅マップを元にボトルネックリンク最大化問題を解く．

各データ転送についてマルチキャストの再計画を行い，線形計画法で最適化を行う．これによって合計スループットが改善された場合には，その計画を採用する．これを全ての転送について順番に行い，どの転送を再計画しても合計スループットが改善されなくなったら停止する．

6. 実験と評価

本アルゴリズムを評価するために，実際の 400 ノードにおけるトポロジに対しと様々なバンド幅・転送計画をランダムに生成し，本データ転送アルゴリズムの評価を行った．バンド幅は 100 から 1000 の間で一様分布させ，source 及び destination の数を (5,10,50)，データの種類を (5, 10, 50, 100) と変化させて 10 題ずつ問題を生成した．評価したアルゴリズムは以下の四つである．

Random-flat tree 各 destination がランダムに source を選択し，source が全 destination に直接データを転送する

Random pipeline　各 destination は最も大きなバンド幅で接続できる source を独立に選択し，同じ source から転送される destination でパイプラインを構築する．パイプラインの順番はランダムに決定する

Topology-aware pipeline　各 destination は最も大きなバンド幅で接続できる source を独立に選択し，同じ source から転送される destination でパイプラインを構築する．パイプラインの順番はトポロジ情報を用いて，各リンクを一回ずつしか用いないようにする　(5.1 節で説明した計画に相当する)

Improved pipeline　本研究が提案するアルゴリズムである．Topology-aware pipeline の結果に対し，転送の再計画を行い，リンク共有を回避したものである．

各アルゴリズムで転送計画が定まったら，線形計画法を用いて合計スループットを最大化し，この値でもってアルゴリズムを評価した．Random-flat tree と Random pipeline は乱数を用いるので，10 回ずつ実行して平均を取った．この結果を図 6，8，10 に示す．横軸はパイプラインの数，縦軸は同じ問題に対し Random-flat tree を用いたときのスループットに対する倍率を表している．また実線・点線・破線は異なった問題に対する結果を，線の濃さは異なったアルゴリズムを示している．

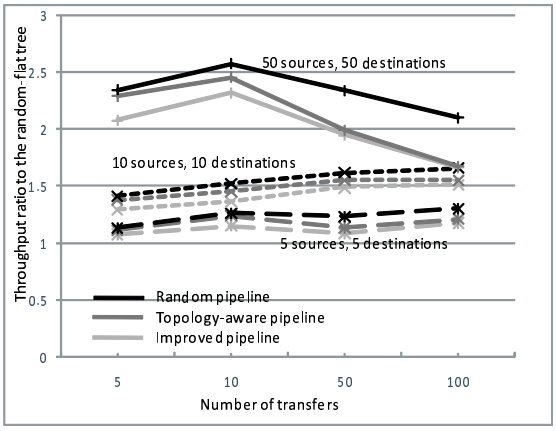

図 5　Source と destination が同数の時の結果
図 6　Results: when the number of sources equals to that of destinations

全部で 36 条件の転送中，2 つを除き全ての問題において，提案アルゴリズムが最も良いスループットを示した．Random-flat tree のアルゴリズムに対し最大 2.9 倍，平均 1.7 倍の性能を示した．End-to-end のバンド幅は用いるがトポロジは考慮しない Random-pipeline に対しても，最大 1.7 倍，平均 1.3 倍の性能を示した．唯一 Random-flat tree よりも悪い性能となったのは source が 5 つ，destination が 50 で転送数が 50 及び 100 の問題である．この原因は，パイプラインを用いるアルゴリズムが最も転送量が小さなリンクに制約されてしまったためだと思われる．

また同じトポロジを用いつつ，リンクのバンド幅を 500 から 1000 の間で一様分布させた場合，全て一定値 1000 とした場合もほぼ同様の結果となり，本手法がトポロジを用いない手法に比べて有効であるという結果が得られた．またアルゴリズムの実行時間は，400 ノード・50 転送で 10-20 秒と，実用上十分な速度で

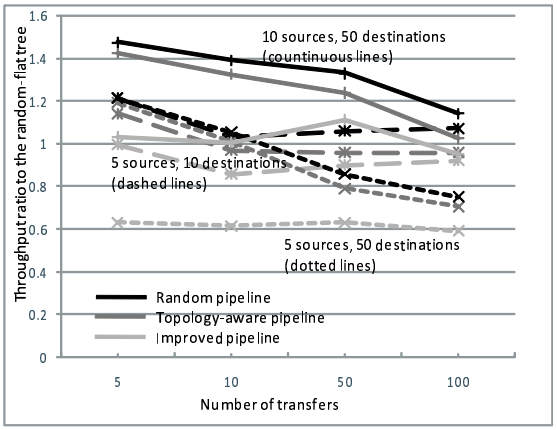

図 7　destination 数より source 数が多い時の結果
図 8　Results: when the number of sources is more than that of destinations

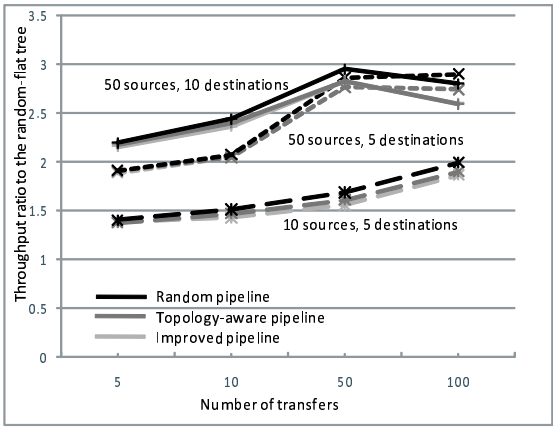

図 9　destination 数より source 数が少ない時の結果
図 10　Results: when the number of sources is less than that of destinations

あった．

7. おわりに

複数の source から複数の destination へのデータ転送が複数ある時に，トポロジ情報及びバンド幅マップを用いることにより，スループットを最大化するような転送計画アルゴリズムを提案した．複数の転送を同時に行う場合，トポロジを考慮しないアプローチではリンクの競合が発生し，転送速度が低下してしまう．

本稿ではまず 4 章において，トポロジを考慮した複数のソースを用いたパイプライン・マルチキャストのアルゴリズムを示した．この転送計画ではリンクの競合が発生せず，しかもパイプライン中に含まれるボトルネックリンクが最大になる．さらに 5 章において，複数の転送に対し合計スループットを向上させるようなアルゴリズムを示した．まず，先のパイプライン・マルチキャストを用いて各転送を独立に計画する．その後各転送を再計画し，複数のソースを活用することでリンクの共有を回避するよう転送計画を改善する．

このアルゴリズムの評価を 6 章において行った．シミュレーションの結果，全転送先がランダムに転送元を選択した場合に比べて最大 2.9 倍，平均 1.7 倍の性能を示した．ノード間のバンド幅は用いるがトポロジは考慮しないアルゴリズムに対しても，最大 1.7 倍，平均 1.3 倍の性能を示した．またアルゴリズムの実行時間は，400 ノード・50 転送で 10-20 秒と，実用上十分な速度であった．

今後は，既存の一般的な探索手法を組み合わせることで探索性能を向上させるとともに，本転送計画アルゴリズムを利用した，データ・インテンシブなアプリケーションを効率的に各ノードに割り当てできるようなタスクスケジュラーを構築していきたい．データ・インテンシブなタスクを分散環境の各ノードに割り当てる際には，タスクの割り当てによって必要なデータ転送も変化する．あるタスクスケジュールに対し，本アルゴリズムを用いて最適な転送を計画し，転送コストを小さくするようなスケジューリングを選択するようにしたい．

謝　辞

スループット問題の NP 困難の証明は柴田剛志さんに教えていただきました．ここに感謝いたします．

参考文献

1) N. T. Karonis, B. R. de Supinski, I. Foster, W. Gropp, E. Lusk and J. Bresnahan, Exploiting Hierarchy in Parallel Computer Networks to Optimize Collective Operation Performance, 14th International Parallel and Distributed Processing Symposium (IPDPS '00) (Cancun, Mexico), May 1-5 2000, pp. 377-384.
2) Dongyu Qiu and R. Srikant. Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks ACM SIGCOMM 04 (Portland, Oregon, USA.), Aug. 30-Sept. 3, 2004,
3) Tatsuya Shirai, Hideo Saito and Kenjiro Taura. A Fast Topology Inference — A building block for network-aware parallel computing. In Proceedings of the 16th IEEE International Symposium on High Performance Distributed Computing (HPDC 2007) (Monterey, USA.) June 2007, pp.11-21
4) Mathijs den Burger, Thilo Kielmann, and Henri E. Bal. Balanced Multicasting: High-throughput Communication for Grid Applications Supercomputing 2005 (SC05) (Seattle, USA.), November 12-18, 2005
5) Mathijs den Burger, and Thilo Kielmann. MOB: Zero-configuration, High-throughput Multicasting for Grid Applications IEEE International Symposium on High Performance Distributed Computing 2007 (HPDC'07) (Monterey, USA.) June 27-29, 2007

付　録

A.1 1 転送に関する合計スループット最大化問題が NP 困難であることの証明

ここでは，任意の 3SAT 問題について，対応する 1 種類のデータに対するネットワークスループット最大化問題が必ず存在し，スループット最大化の最適解を求めるアルゴリズムによって任意の 3SAT 問題を解けることを示す．これにより，スループット最大化問題が NP 困難であることを示すことが出来る．

A.1.1 末端リンク充足問題

あるネットワークグラフが与えられたとき，いかなる転送計画の合計スループットも，各 destination ノードの末端リンクのバンド幅の合計を超えることはできない．この上限値を与える転送計画が存在するがどうかを判定する問題を考える．スループット最大化問題の最適解を求めるアルゴリズムがあれば，この判定問題もその計算量以下で解くことが出来る．

A.1.2 SAT クロージャのネットワークによる表現

図 11 の左側に示すようなネットワークの部分グラフを考える．3 つの destination ノードである D_a, D_b, D_c がバンド幅 a, b, c で 1 つのスイッチにつながっていて，さらに上位のスイッチに source ノード S_d, S_e がバン

ド幅 d, e で，さらに上位のスイッチにリンク l_x がつながっている．ここで，d, e は以下の条件を満たすものとする．

$$max(a,b,c) < d < min(a+b, b+c, c+a) \quad (1)$$

$$max(a,b,c) < e < min(a+b, b+c, c+a) \quad (2)$$

つまり S_d と S_e は D_a, D_b, D_c のうち 2 つの末端リンクのバンド幅しか満たすことができない．よって，末端リンクが全て満たされる解が存在するのであれば，D_a, D_b, D_c のいずれかへの転送は上位のリンク l_x から降りてくるはずである．その場合，l_x を流れるデータは a 又は b 又は c であり，もし l_x の接続されているスイッチにちょうど a 又は b 又は c でデータを転送できるノードが存在すれば，この条件は満たされる．まとめると，もしこの部分グラフの destination 全てが最大スループットでデータを受け取れるならば，l_x を通じて実際に転送されるデータ量は a, b, c のいずれかである．これを $a \vee b \vee c$ と表すものとする．

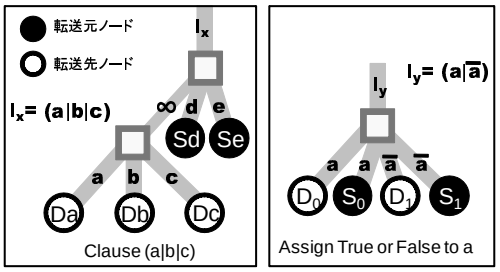

図 11 3-SAT クローズのスループット最大化問題への変換

A.1.3 論理変数のネットワークによる表現

次に，図 11 の右側に示すような部分グラフを考える．1 つのスイッチに 2 つの destination D_0, D_1 がバンド幅 $a, \bar{a}$ で，2 つの source S_0, S_1 がバンド幅 $a, \bar{a}$ でつながっており，さらに上位のスイッチに対してリンク l_y がつながっている．なお，$\bar{a}$ とは a と異なる値である．ここで，l_y のバンド幅は以下を満たすものとする．

$$max(a, \bar{a}) < ly < a + \bar{a} \quad (3)$$

つまり，l_y を通して転送できるのは a 又は $\bar{a}$ のどちらかである．ここで，もし l_y を通して転送されるデータ量が a とも $\bar{a}$ とも異なった値だとすると，D_0 や D_1 のスループットが充足されなくなってしまう．このため，l_y を通じてデータが転送されるのであれば，その値はちょうど a 又は $\bar{a}$ である．

A.1.4 3SAT 問題のネットワークによる表現

まず以下の要素を定義する．

- 論理変数 $X = \{x_i | 1 \leq i \leq n\}$
- クロージャ $\{C_i = a_{i1} \vee a_{i2} \vee a_{i3} | 1 \leq i \leq m\}$
- リテラル $a_{ij} = x$ または $\bar{x}$ (但し $x \in X$)

3SAT 問題とは，論理式 $C_1 \wedge C_2 \wedge \cdots \wedge C_n$ を充足するような論理変数 X への真偽の割り当てを求める問題である．ここで，任意のクロージャは A.1.2 で説明したような部分グラフで表現することができる．さらに，論理変数の割り当ては A.1.3 で示した部分グラフで表現できる．こうして，m 個のクロージャと n 個の変数に対応した部分グラフを生成し，それらを一つのスイッチに接続したトポロジを考える．各変数に相当するのは正の値のバンド幅で，異なる変数に相当するバンド幅の値はそれぞれ異なるものとする．これを図 12 に示す．

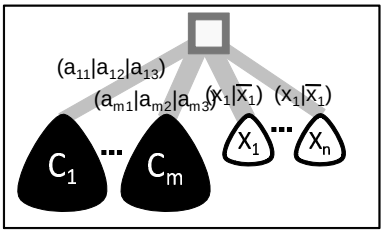

図 12 3SAT クロージャのスループット最大化問題による表現

このグラフに対し，末端リンクを充足する計画を立てられるかどうか，という判定問題を考える．この問題を解くことと，元の 3SAT 問題を解くことは同値である．実際に末端リンクを充足する計画が存在するならば，変数に対応した部分グラフのリンクに対し x_k あるいは $\bar{x_k}$ が選択されたことに対応するので，元の 3SAT 問題が解かれたことになる．これにより，任意の 3SAT 問題が 1 転送に対するスループット最大化問題に帰着することが示された．