2013 年 7 月 8 日　情報システム基盤学合同輪講発表資料

MOB: Zero configuration High-throughput Multicasting for Grid Applications

著者：　Mathijs den Burger，Thilo Kielmann

出典：　*Proc. of the 16th int'l symposium on High performance distributed computing pp.159-168, 2007.*

発表者：　高性能コンピューティング学講座 本多・近藤研究室　1353029　松下明史

1　はじめに

近年、インターネットを介して、地理的に離れた複数の計算資源 (ノード) を接続してクラスタを形成し、並列計算させることで、仮想的に 1 台の高性能コンピュータをつくるグリッドコンピューティングが注目されている。このような環境では、効率的にデータをマルチキャストする方法が重要である。

そこで本論文では、従来手法で行われていた帯域幅情報の取得や転送ツリーの構築などの事前の設定を行わないで、高いスループットを実現するマルチキャスト手法、MOB(Multicasting Optimizing Bandwidth) を提案する。

2　研究背景

2.1　マルチキャスト

マルチキャストとは、ネットワーク内の指定した複数の相手に対して同じデータを送ることである。もっとも単純な方法は、送信元 (ソースノード) が、相手ノードへ順々にデータを送る方法である。しかし、この方法では、ノード間の帯域幅を無視して配信しているため、ボトルネックになるノードに大きな負荷がかかり、送信が完了するまで時間がかかる。

2.2　Balanced Multicasting

Balanced Multicasting[1] は、ネットワーク監視ツールを用いてノード間の帯域幅情報を取得し、その情報から最適な転送ツリーを構築するマルチキャスト手法である。例として図 1(a) があった場合、このアルゴリズムは図 1(b) の 3 つの転送ツリーを構築し、送信データも転送ツリーの容量に合わせて分割し送信する。しかし、この Balanced Multicasting の問題として、帯域幅情報の更新間隔が長く、実際の帯域幅と誤差が生じることが挙げられる。

3　提案手法

以下に述べる BitTorrent[2] の考えをマルチキャストに適用した手法が、MOB である。

3.1　BitTorrent

BitTorrent は、大量のデータを配信することを目的として開発された Peer-to-Peer ファイル共有アプリケーションである。具体的には、.torrent ファイルより、自分がダウンロードしたいファイルを既に手に入れているユーザを探し出す。その後、そのユーザの中から、適当な数のユーザと接続し、取得しているデータを報告し合い、未取得データを所持しているノードに対し送信要求を送る。送信要求を受け取ったノードは、送信可能な状態なら送信を開始する (choking algorithm)。

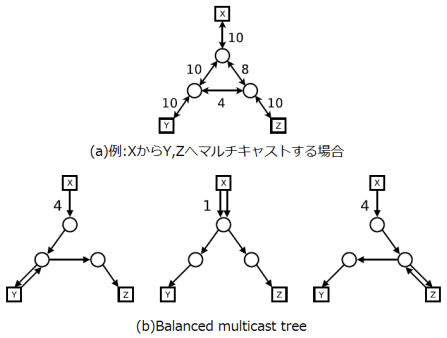


図 1: Balanced Multicasting の例

送信する際は 1 つ数百 k B のチャンクごとに送り、10 秒間に受信したチャンク数を測定する。1 つも受信できなかった場合、そのノード間の帯域幅が著しく小さいと判断し、通信を中断する。その後、他のノードへ再び送信要求を出す。また、1 つ以上受信している際は、そのまま通信を継続する。ただし、連続で 30 秒続いた場合は、通信を中断し、再び choking algorithm を行う (optimistic unchoke)。この動作を繰り返すことで、データをより早くダウンロードできる。

3.2　MOB

MOB は 4 つの段階に分けられる。

1 クラスタ内接続

各ノードは、クラスタ内の他の無作為に選んだ最大 5 つのノードと接続を行う。

2 クラスタ間接続

各クラスタから 1 つ代表ノードを選び出し、それらのノードで 1 と同様の接続を行う。

3 データ転送

ソースノードは送信するデータを、クラスタ内のノード数分に分割してそれぞれに配信する。配信

後、段階 1、2 で接続したノードに対して、choking algorithm、optimistic unchoke を行い、データ転送を行う。

4 最終同期

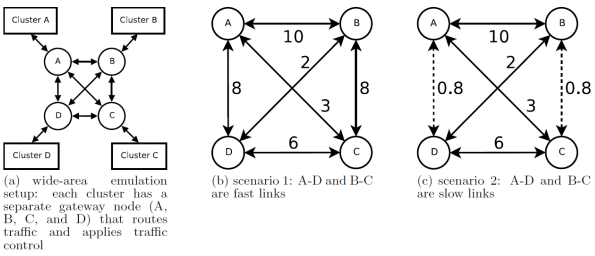
あるノードがデータ全体を取得した場合、全ノードに対して確認応答をとり、同期する。その後、他のノードが全データを取得するまで、送信要求に答え続ける。

4 実験

提案手法の性能を確認するため、様々な条件で 600MB のデータをマルチキャストした際のスループットを測定した。また、比較対象として、MOB の 1、2 の接続条件をなくしたマルチキャストである BitTorrent と、ネットワーク監視データを用いるマルチキャストの 1 つである Balanced Multicasting[2] に対しても同じ実験を行った。ただし、Balanced Multicasting に用いる帯域幅情報は、各条件での初期設定とし、実験中は更新しないものとする。

4.1 クラスタ間通信がボトルネックとなる場合

4 つのクラスタを図 2(a) となるようにネットワークを構成した。そして、クラスタ間のスループット条件の 5 通りと、各クラスタの規模が均一な場合 (各クラスタに 16 ノード) と、各クラスタの規模が不均一な場合 (クラスタ A が 4 ノード、クラスタ B が 12 ノード、クラスタ C が 16 ノード、クラスタ D32 ノード) の 2 通りの合計 10 通りの実験環境で実験を行った。



(a) wide-area emulation setup: each cluster has a separate gateway node (A, B, C, and D) that routes traffic and applies traffic control

(b) scenario 1: A-D and B-C are fast links

(c) scenario 2: A-D and B-C are slow links

図 2: 実験環境 (図中の数字の単位 : MB/s)

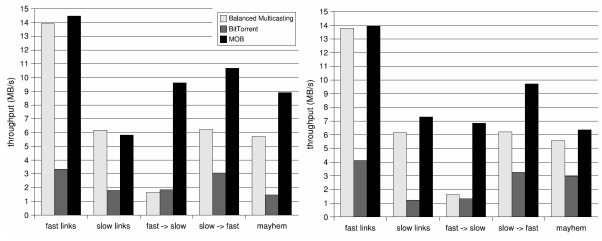


図 3: クラスタ間ボトルネックを発生させた際のスループット (左 : クラスタ規模均一　右 : クラスタ規模不均一)

図 3 より、MOB のスループットは、Balanced Multicasting のスループットと比べて全ての条件で同等か上回る結果を出した。特に、実験中にスループットが変化する条件においては、結果の差が顕著になった。これは、Balanced Multicasting に用いられたネットワーク監視データと実際のスループットに差が出たことが原因である。

4.2 クラスタ内通信がボトルネックとなる場合

4 つのクラスタ間のスループットを、10Gb/s とし、クラスタ内の帯域幅を 100Mbit とし、クラスタ内通信がボトルネックとなる環境で、クラスタ内ノード数を 1、2、4、8、16 ノードと変化させて実験を行った。

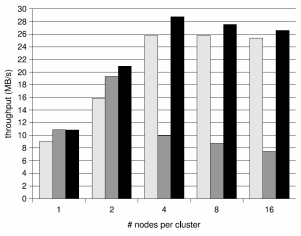


図 4: クラスタ内ボトルネックを発生させた際のスループット

図 4 において、クラスタ毎のノード数が 1 つ、2 つの際は、MOB と BitTorrent で大きな差は見られなかったが、クラスタ毎のノード数が 4 つになると BitTorrent のスループットは大幅に低下した。これは、BitTorrent において、クラスタとゲートウェイ間のリンクに負荷がかかったためと考えられる。

5 終わりに

本論文では、転送ツリー構築などを行わずに、高いスループットを実現するマルチキャスト手法として、MOB を提案し、実験を行った。

提案手法は、クラスタ間、クラスタ内にボトルネックが発生している環境でも、従来手法のスループットを上回る結果を出した。このことから、提案手法は、現実的なネットワーク環境においても適用できる手法と考えられる。

参考文献

[1] Mathijs den Burger, Thilo Kielmann, “Balanced Multicasting: High-throughput communication for Grid applications” Supercomputing 2005. Proceedings of the ACM/IEEE SC November 12-18 2005 Conference, p46

[2] B.cohen, “Incentives build robustness in BitTorrent” In Proceeding of Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA, USA, May 2003. http://www.bittorrent.org/bittorrentecon.pdf