TCPにおける再送タイムアウトに関する一考察

5J－8

釘本 健司† 天海 良治‡ 村上 健一郎†

NTTソフトウェア研究所† NTT基礎研究所‡

1 はじめに

長距離高速ネットワークにおいて、帯域を有効に使ってTCP(TransmissionControl Protocol)[1]のスループットを向上させるためには、伝送路で使用できる最大の帯域を使い切るだけのパケットを送り出せるように、送信バッファサイズおよび受信バッファサイズを十分に大きくし、確認応答(ACK: Acknowledgement)を待たずに送信できるデータ量の最大値、すなわち最大ウィンドウサイズを増やすことが必要である[2]。

ところが、単一ストリームの通信において、エンドノードが接続されているネットワークの帯域に比べて低い帯域のリンクを通る場合には、最大ウィンドウサイズを増やすと、スループットがネットワークの静的な帯域の30%以下になる現象を確認した。

本論文では、まずこの現象を解析し、次に再送タイムアウトの値について触れ、最後にネットワークの静的な帯域見積りに基づく送信レート制御を行なって重度の輻輳による転送停止が単一ストリーム時に起こらないようにする手法を提案する。

2 アノマリの発生

我々はまず、バッファサイズを広く変化させた時のスループットの変化を調べる実験を行なった。実験に使用した二台のワークステーションはルータのFDDI(Fiber Distribution Data Interface)リングにつながっており、二台のルータの間は、意図的に輻輳を起こすために45Mb/sのVC-3で結んである。この環境でバッファサイズを送信バッファ、受信バッファともに1536(bytes)から524288(bytes)まで変化させた。この実験の結果、バッファサイズが実際に使用可能な帯域に対して非常に大きな場合には、既存のTCPアルゴリズムが想定していない事態(アノマリ)が発生し、かえってスループットの低下をもたらすことが確認された。

3 アノマリの発生原因の解析

図1は、アノマリが起こっている時のシーケンス番号とACK番号のそれぞれの増加の様子を表している。実験から得られたデータを解析した結果、アノマリが起こっている時のTCPの動作は以下のようになる。

1. 送信側のバッファサイズは大きいので、実際に使用可能な帯域以上にパケットをネットワークに送り込んでしまうために輻輳が起こり、送信側のパケットの連続したシーケンス番号の複数のパケットの消失が起こる。

2. 送信側パケットの消失のため、受信側は期待しているものよりも大きなシーケンス番号を持つパケットを受信する。これらを受けとるたびに受信側は次に期待しているシーケンス番号をセットして直ちに送信側にACKを返す。これらはDuplicate ACKと呼ばれる。

3. 消失したパケットが送り出された直後の送信側のウィンドウは十分大きいため、Duplicate ACKにより輻輳が検知される前に、先行して多くのパケットが送られる。これらのパケットの数だけDuplicate ACKが返されることになる。

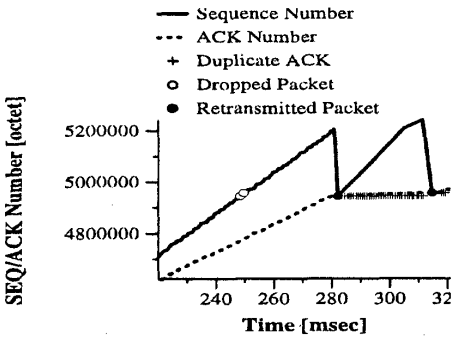

図1: アノマリ発生時の順序番号の増加

4. 送信側は、受信側から送られてきたDuplicate ACKの最初の3個を受けとると、消失したパケットのために当該パケットだけの再送を行なう(Fast Retransmit)。また、輻輳状態から回復した後に早くもとの転送速度に戻るように、輻輳ウィンドウをもっとも小さい値に戻すのではなく、3個のDuplicate ACKが返った時点での輻輳ウィンドウの半分の値にされる(Fast Recovery)。

5. 送信側は、すでに送られたものの直後のシーケンス番号のパケットから送信を再開しようとする。ところが、Duplicate ACKが三つ続いた直後は、輻輳ウィンドウが小さ過ぎてパケットの送信ができない。4個目以降のDuplicate ACKは、ネットワークの潜在的な帯域と見做されるため、送信側ではACKが返るたびに輻輳ウィンドウがt_maxsegずつ増やされる。こうして増加した輻輳ウィンドウがFast Retransmit直前の値を越えるとパケットの送信が再開され、いくつかのパケットが送られる。Fast Retransmitで再送したパケットに対するACKが返ると、大きくなり過ぎた輻輳ウィンドウを是正するために($ssthresh$+t_maxseg)の大きさに減じられる。そのために再びパケットの送信ができなくなる。

6. 送り側から再送したパケットの次のパケットも消失しているので、再びこのパケットに対するDuplicate ACKが続く。ここで再びFast Retransmit and Fast Recoveryアルゴリズムが働く。しかし、先行して送られたパケットが一度目に比べると少ないため、輻輳ウィンドウが十分に大きくならないうちにDuplicate ACKは尽きる。そのため送信側からのパケットの送信が全く行なわれなくなり、再送タイムアウトが起こるまでの間、すなわち、約1.2秒間転送は完全に止まる。このためにスループットが低下する。

Takeshi Kugimoto, Yoshiji Amagai, Ken-ichiro Murakami
NTT Software Laboratories, NTT Basic Research Laboratories

4 再送タイムアウトの短縮

我々が実験に使用したワークステーションでは、TCP の再送タイムアウトを判定する UNIX カーネル内関数である *tcp_slowtimo()* は、500ms ごとに呼ばれる。再送タイムアウトは最低でもこの関数が 3 回呼ばれてから起こる。したがって、再送タイムアウトの最小値は 1.0 秒から 1.5 秒の間の値である。この再送タイムアウトの値を小さくすれば、全体の転送時間に占める転送停止時間の割合が減るので、スループットは向上する。試しに *tcp_slowtimo()* を 1tick(10ms) 周期で呼ばれるように改造してみたところ、500ms 周期で呼ばれる場合に比べてスループットは 150% も向上した。ただし、再送タイムアウトを単に短くすればよいというわけではない。再送タイムアウトには (1) 不必要な再転送の防止、(2) 輻輳を回避して他のトラフィックにも転送の機会を与える、という役割もある。これを根拠もなくむやみに短くすることは、問題に対する適切な対処とはいえない。

5 使用可能帯域の見積りと送信レート制御

再送タイムアウトの値を変更することなくスループットの向上をはかるには、アルゴリズムの改造によって再送タイムアウトができるだけ起こらないようにすればよい。その実現には、(1) Fast Retransmit and Fast Recovery アルゴリズムを連続したパケット消失にも対応できるようにすること (2) 輻輳が起こらないように送り側の送信レートを制御することの二つの指針が考えられる。

このアノマリの根本的な原因は、パケットを輻輳が起こるほどネットワークに送り込んでしまうことである。したがって (1) の方法では、送信側が過度にパケットを送り込むことには変わりがないので、ネットワークに定常的に輻輳を起こすことになる。そこで、我々は (2) の輻輳が起こってしまうほどパケットを送り込まないように送信レート制御を行なう方法をとる。

(2) の方法をとるには、パケットの送信時点で使用可能な帯域を適切に知ることができ、その帯域を越えないように単位時間あたりの送信パケット数を制限する仕組みが必要である。しかし、これまでホストがネットワークの使用可能帯域を知る方法は知られていなかった。そこで、我々は使用可能な帯域を知る方法として、送信側に返る ACK のスピードから使用可能帯域を知る方法を提案する。

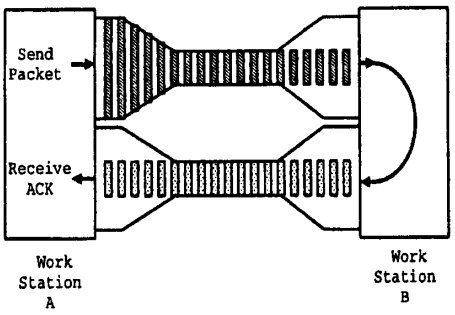

図 2: 使用可能帯域の見積り

図 2 に示すように、送信側から送られたパケットがネットワークを通過して受信側に届く時には、転送レートはその経路上のもっとも細いリンクの帯域まで落ちていると考えられる。受信側に到着したパケットに対して ACK が送信側に送り返されるので、ACK から計算できる単位時間あたりのパケットの量は、ネットワークを通して受信側に到着したパケットの転送レートを反映していると考えられる。これを利用して、使用可能帯域を越えないように送信レートを制限する。

図 3 は、図 1 に示したグラフの近辺での送信側でのパケットの送信レートと、返ってきた ACK を元に計算された転送レートを 30ms ごとにそれぞれ計算してグラフにしたものである。

この実験環境でのネットワークの使用可能帯域は 45Mb/s であり、ACK を元に計算された転送レートはこの値以内に収まっていることがわかる。それに対して送信レートの最大値は使用可能帯域を越えていることがわかる。

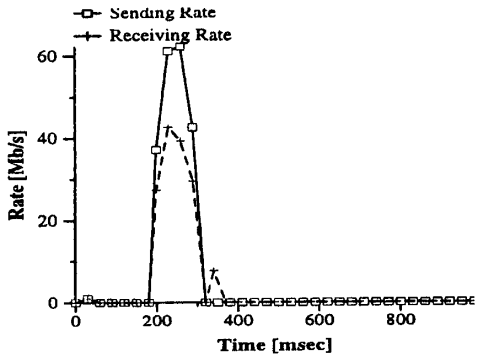

図 3: 送信レートと受信レート

我々は、こうして得られるネットワークの静的な最大使用可能帯域をもとに、過度のパケット送信が行なわれないようにするため、送信レート制御を行なう以下のような手法を提案する。

1. TCP のコネクションが確立した直後は通常のアルゴリズムで通信を開始する。
2. 受信側から返ってくる ACK を元に現在のネットワークの転送レートを定期的に計算する。転送レートの変動が少なくなって一定時間が経過するか、輻輳が起こって Fast retransmission and Fast Recovery で解決できないような複数パケットの消失が発生したら、その時点の転送レートが現在使用可能なネットワークの帯域であるとして記録する。
3. 得られたネットワークの帯域をもとに送信レートを定期的にチェックし、帯域以上に送られないように制限する。
4. 定期的に送信レートをあげ、RTT 後に転送レートが大きくなるようならその送信レートを新しい転送レートとする。転送レートに変動がなければ送信レートを直ちに元に戻す。

6 おわりに

長距離高速インターネットにおいて、TCP のスループットを大きくするための過度のバッファサイズの増大は、既存アルゴリズムが想定していないアノマリを発生させることを示した。また、このアノマリを回避するために、ACK によるネットワークの静的な帯域見積りに基づいた転送レート制御法を提案した。

参考文献

[1] J. Postel: Transmission Control Protocol, RFC793, 1981

[2] V. Jacobson, R. Braden: TCP Extensions for Long-Delay Paths, RFC1072, 1988.

[3] Gary R. Wright, W. Richard Stevens: TCP/IP Illustrated, Volume 2, 1995