Vol. 36 No. 7　　情　報　処　理　　July 1995

特別論説

情報処理最前線

ギガビットネットワークの壁†

後藤滋樹††　村上健一郎††

1. はじめに—問題の所在

コンピュータネットワークの利用が拡大するとともに高速化への要望が高まっている．実際に2.4 Gbps（毎秒ギガビット）の高速バックボーンを用いてコンピュータ間の通信の実験を行う事例も出現している．ただし2.4 Gbpsという数字は光ファイバの伝送速度を示しているのであり，実際に長距離区間で測定してみると，光ファイバの速度に比べてコンピュータ通信が実際に達成できる速度は著しく低い．

低速の要因はひととおりではないが，最も深刻な原因は光の伝播に遅延が存在するという本質的な制限である．この現象は単純であり容易に理解できるが，従来はあまり注目されることがなかった．そのために超高速のネットワークを議論する際に，しばしば誤解の原因になっていたように思われる．

本稿は，このような観点からネットワークの遅延時間に注目すべき理由を述べ，具体的に遅延時間がコンピュータ通信に及ぼす影響を考察する．さらにコンピュータネットワークの高速化のために考慮すべき技術についても紹介する．

2. 遅延時間が支配的である理由

コンピュータ通信，あるいはより広くデータ通信と呼ばれる分野には次の二つの送受信の方法がある．この二種類は通信の途中でエラーが生じるときの対処法により区別される．エラーは通信回線の障害などによって発生する場合もあるが，それればかりではなく，コンピュータの処理速度が通信速度に間に合わない場合やデータを保持すべきメモリ容量が不足する場合にも発生する．高速の通信においては，下の二つの送受信の方法を明確に区別することが大切である．

（1） エラーを補償する方法：受信側はデータを正しく受け取ると受信確認(ACK) を送信側に返送する．送信側では，ある一定時間内にACKが戻らないデータに関しては途中でエラーが生じたものと見なして，そのデータを再送する．（あるいは受信側が積極的にNACKというACKの否定信号を返す方法もあり得るが，ここでは以下に述べる実例に即して説明する．）

（2） エラーを補償しない方法：受信側は何らデータの確認をしない．したがって送信側でも再送の処理を行わない．

一般に電子メールやファイル転送などのコンピュータ通信においては，内容が正しく送受信されることが大切であるから(1)の方法をとる．これに対して音声や画像の信号をリアルタイムで通信するような場合には(2)をとることが多い．ただし，これは利用傾向の概略であり，(1)も(2)もそれぞれに一長一短があるから，特質を理解して使い分けることが肝要である．

さて，本稿で問題にするのは上記の(1)の方法，すなわち受信確認（ACK）を必要とする場合である．なお，以下の説明は典型的な(1)の方法として広く使用されているTCPのプロトコルに即している．TCPはインターネットで利用されているTCP/IPのトランスポート層のプロトコルである[1]．

ACKを必要とする場合の送信側の動作は次のようになる（**図-1** 参照）．なおデータを送受信するためにひとまとめにしたものをパケット(packet) と呼ぶ．

（i） データをパケットの形で送信する．このときに送信済みのパケットのコピーを再送する場

† The Speed Limit of Computer Networks by Shigeki GOTO and Kenichiro MURAKAMI (Global Computing Laboratory, NTT Software Laboratories).

†† NTTソフトウェア研究所広域コンピューティング研究部

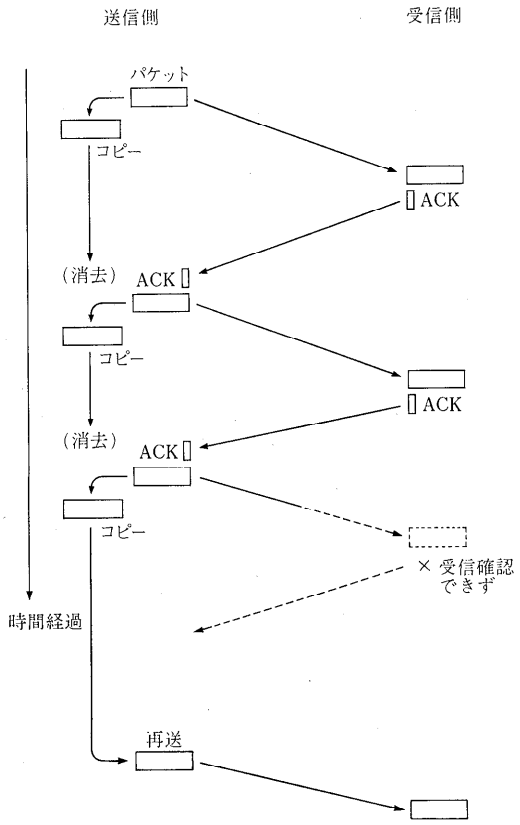


図-1　ACK を必要とする送信

合に備えて保持しておく．

（ii）　受信側から確認（ACK）が届くのを待つ．

（iii）　ACK が無事に届いた場合には，再送の必要がないため保持していたパケットを消去する．そして次のパケットを送信するために（i）に行く．

（iv）　もし一定時間以内に受信側からの ACK が届かなかった場合には，保持していたパケットを再送するために（i）に行く．

ここで(iv)の「一定時間」をどのように定めるかによって，エラーが発生した場合の動作が変化する．TCP の場合に多く用いられている方法は，まずパケットが送信側と受信側を往復するのに要する時間を推定する．その推定往復時間を経過しても ACK が届かない場合には，何らかのエラーが生じたとして再送すべきであるが，実際の往復時間は種々の原因で多少は変動する．厳密に推定往復時間で再送を行うと再送の機会が相当に多くなる．そこで通常の処理では推定往復時間よりも長く待つ．たとえば，推定往復時間の 2 倍を経過しても ACK が届かない場合に初めて再送する．この 2 倍という数字は実装が簡単であるという理由で多くの実例で選ばれている．なお往復時間の推定のためには，ACK がうまく返る場合の送信時刻から ACK 受信時刻までの時間を測定すればよい．

いずれにしても，この方法を用いると送信側の動作のほとんどは（ii）で説明した待ち時間となってしまう．これは図-1 からも明らかであろう．もっとも待ち時間が無視できなくなるのはパケットの往復時間が意味を持つ場合である．今日の通信は光ファイバを用いているので，光速で往復すれば瞬時であるかのように思われるかもしれないが，実は光速はそれほど速いわけではない．真空中では 1 ms（ミリ秒）に光は約 300 km（キロメートル）進む．ファイバの中では真空よりも遅くなり 1 ms に約 180 km といわれているから，中距離以上の通信では往復時間が数 ms のオーダになり，無視できなくなる．

さて上述の送信側の動作にはいかにも無駄がある．ここで述べたように ACK をひたすら待つ方法をストップアンドウェイト（stop and wait）と呼ぶが，これは実用的ではない．実用に供されているプロトコルでは無駄が生じないように改良されている．その改良法と限界を次章で解説する．

3. インターネットプロトコル（TCP/IP）の問題点と改良

図-1 に示したストップアンドウェイトの方法は，あたかも電車の運行方法に似ている．つまり送信側からお客を乗せて電車が出発し，受信側からは電車が回送されるという具合である．この回送電車を待たなければ，次のデータ（お客）を送信することができない．データ通信の場合には，同じパケットが回送されてくるわけではないけれども，往復時間が支配的になるという様子はまったく同じことである．

電車の運行の場合に，このように一編成しか車両を用意しないのは現実的でない．普通は多数の編成の電車を準備して，最初の回送電車を待たなくても次々に電車を発車することができる．これと同様な工夫がデータ通信の場合にも行われてい

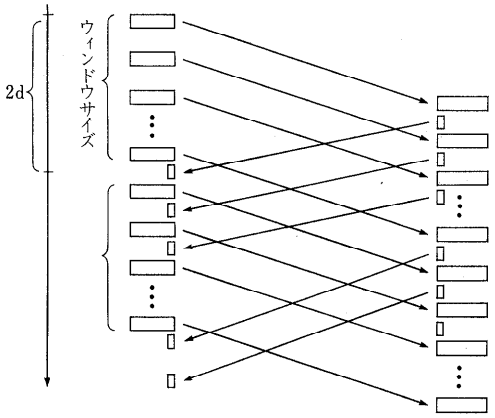


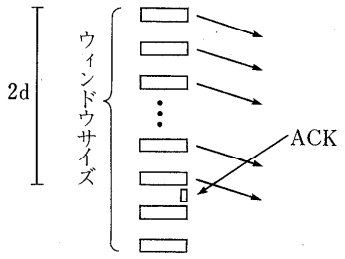
図-2 ウィンドウ制御

る．データ通信の場合には，この工夫をウィンドウ制御という．

図-2 に示すようにウィンドウ制御，つまり電車の場合でいえば複数編成の電車を持つ方式，においては最初に送信したパケットに対するACKが届く前に次のデータを送信できる．ただし前章で述べたように送信済みのパケットをACKが届くまで保持しておく必要があるから，ウィンドウのサイズ（同時運行可能な電車の数に相当する）をやみくもに大きくするわけにはいかない．その分のメモリ容量が必要になるからである．TCPプロトコルの場合のウィンドウサイズは標準では64 kB（キロバイト）までであるが，実際のワークステーションなどの実装ではメモリの容量を勘案して4 kB，あるいは8 kB程度に抑えている例もある．

さてウィンドウ制御を採用した場合に，実効速度はどのようになるか．これは図-2から概略の様子が読み取れるのであるが，少し解説したほうが理解しやすいと思う．今，送信側から出たパケットが受信側に到着するまでに d 時間（dはdelay, 遅延の意味）かかるとする．d が片道の遅延時間であるから，往復時間は $2d$ である．つまり送信側から最初に出たパケットが無事に受信側に到着してACKが送り返されるまでには $2d$ 時間が経過する．この $2d$ 時間の間にウィンドウサイズで決められた分量のパケット（電車でいえば手持ちの車両の数に相当）をちょうど送信する場合が一番無駄がない．

もし $2d$ 時間の間にウィンドウサイズよりも少ないパケットしか送信しないとすると，明らかに



ウィンドウサイズが大きくても $2d$ 時間以内に送信が完了しない場合．

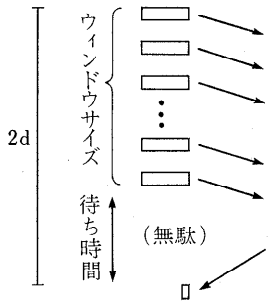


図-3 ウィンドウサイズと $2d$ の関係

輸送力（実効速度）に余力がある．また逆にウィンドウサイズを送信し終っても $2d$ 時間が経過していなければ，無駄な時間が生じているがウィンドウサイズ以上には送信できないから，これ以上は改善できない（**図-3**）．結局遅延時間が d，ウィンドウサイズが W の場合の実効速度の上限は下式で表わされる．

$$W/2d$$

これがウィンドウ制御を行った場合の実効的な通信の速度である．実際に具体的な数字を当てはめてみると次のようになる．

【例1】

距離：450 km（関東から関西までを想定した数字）

ウィンドウサイズ：4 kB（あるワークステーションの実例）

距離450 kmをファイバ中の光速180 kmで割ると遅延時間（d）は2.5 ms（ミリ秒）となる．往復時間は $2d$ であるから5 msである．ウィンドウサイズの4 kBをビットで表現するには8倍して32 kb（キロビット）となる．32 kb/5 msを計算すると6.4 Mbps（メガ毎秒）である．

【例2】

距離：450 km（上と同じ）

ウィンドウサイズ：8 kB（あるワークステーションの実例）

上と同様の計算により，上の2倍の12.8 Mbpsが得られる．

【例3】

距離：450 km（上と同じ）

ウィンドウサイズ：64 kB（あるワークステーションの実例，TCPの最大値）

上と同様の計算により，例1の16倍の102.4 Mbpsが得られる．

実際に計測された例でも上の各例に相当する結果が得られている．なお厳密にいうとパケットを構成するときには純然たるデータ部分の他に管理用の情報も送信される．上の計算値はそこまで勘案したものではないから，実測値と多少の誤差が生じる．いずれにしても通常のワークステーションを用いた場合はもちろん，TCPの最大値のウィンドウサイズを用いたような実装でも，往復5 msの遅延時間があると到底Gbps（ギガビット毎秒）には達しないのである．

この問題を解決するためには，上限値である$W/2d$の値を大きくする必要があるが，遅延時間は物理的な制限であり，二点間の距離を変えなければ改善できない．残るのはウィンドウサイズである．実際にTCPプロトコルの規格ではウィンドウサイズを拡大する方法がスケーリングと呼ばれるオプションになっている[2]．

このオプションを実装したワークステーションあるいはスーパコンピュータなどを用いて実測をしてみると，確かに実効速度を改善することができる．筆者らは約900 kmの区間に155 Mbpsの通信回線を用意して，TCPプロトコルでも92 Mbpsの速度を出せることを実験によって示した[3),4)]．900 kmという数字を上述の例3に当てはめて計算すると51.2 Mbpsとなるから，92 Mbpsを達成するためには標準のTCPのウィンドウサイズを約1.8倍に拡大したことに相当する．

ただしウィンドウサイズを拡大するためには，先に述べたようにメモリの容量の拡大が必要になる．これが実際の応用における制限となる．

4. 高速ネットワークにおける種々の問題と解決法

これまでに述べたように高速のコンピュータネットワークにおいては遅延時間が無視できない．ただし実際にネットワークを設計する際には遅延時間以外の要因もおおいに考慮しなければならない．以下ではいくつかの要因も含めて考察する．

4.1 エラーからの回復が不必要な場合

本稿では，高速ネットワークにおいて遅延時間の影響が深刻になるという問題を，エラーが起きたら再送をするような通信方式の場合に考察した．その解決法はウィンドウサイズを大きくすることである．

ところで，データ通信でもエラーを気にせずに再送をしない場合がある．この場合にはACKを待つという動作はなくなり，話は大幅に単純化される．たとえば音声や画像を伝送する際に再送の処理をすればデータは間違いなく届くようになるが，再送をすると時間遅れが生じてリアルタイム性が損なわれる．このため画像や音声を通信する場合には，エラーが生じても構わないと割り切る場合が多い．またネットワークの管理用のプロトコルなどで定期的に情報交換をする場合には，必ずしも毎回の通信を正確に行わなくても，いずれ後の交信が成功すればよい．このような目的のためにインターネットではUDPというプロトコルが多用されている．UDPはエラーが起きても再送をしない．

このように，エラーが起きても再送をしないプロトコルを採用するのは，遅延時間の影響を軽減する解決法の一つといえる．ACKを待つ必要がないから往復の時間を気にしなくてもよくなる．

4.2 プロトコルの選択

先に示した例題では，東京と関西の間で標準的なワークステーションを用いてTCP/IPで通信をすると，実効速度がそれほど上がらないという問題があることを数値の実例で示した．

この数値がただちにTCPの限界を示したものではない．この点は誤解されやすいので注意が必要である．すなわち，例題の直後に説明したようにTCPのウィンドウサイズを大きくすれば数値を改善することができる．

もちろんTCPではない新規のプロトコルによ

って解決しようという提案もある．TCPがいつまでも使われるということでもないだろうから，ここで示したような解決策を最初から取り入れた新プロトコルを考案することも可能である．ただし本稿で述べたような本質的な制限はどのようなプロトコルに対しても（エラーの回復をする限り）適用されることに注意していただきたい．（なお後述の4.3の末尾を参照されたい．）

4.3 コンピュータの処理時間

先に示した実例の計算ではもっぱらファイバの中の光速による伝播時間を問題にしたが，実際のシステムでは送信側も受信側もコンピュータの中の通信処理のプログラムが走行して処理をする．この処理のための時間も無視できない．実際にはCPUの能力，メモリへのアクセスのための時間，OSのオーバヘッドなどを考慮すべきである．

CPUの処理能力についてはClark達[5]の研究がある．これによるとBSD UNIXのTCP/IP処理のプログラムを解析をした結果，1パケットのヘッダ部の処理は80386（インテル）の命令数で数えてTCP処理とIP処理の合計で約400命令であるという．なお再送の制御はTCPプロトコルの機能である．またIPプロトコルの重要な機能は経路制御（ルーティング）である．インターネットに接続するためにはIPアドレスが必要であるが，経路制御はIPアドレスを用いて行う．

さて400命令を処理する時間は10 MIPSのチップを用いれば40μs（マイクロ秒）である．逆に1秒間には25,000パケットを処理できる．もし1パケットが4 kB（キロバイト）であるとすれば，100 MB/secつまり800 Mbps（毎秒メガビット）となる．これは100 Mbps程度のLANを使うためには十分な能力である．

メモリへのアクセス時間は，マシンのアーキテクチャによっては深刻な要因となる．たとえば通信インタフェースにIOメモリがあり，パケットをIOメモリから主記憶に移動させなければならないとすると，そのコピーのための時間が問題になる．またオペレーティングシステムの種類によっては，パケットをユーザ空間とシステム空間の間でコピーする必要がある．さらにパケットのエラーの検査のためにTCP/IPプロトコルではチェックサムの計算を行うが，これもメモリを参照する演算である．実際にKay達[6]は，DECstation 5000の処理時間の内訳を分析し，メモリのコピーの時間とチェックサムの計算が最も大きな部分を占めていることを明らかにしている．

このような問題に対して各種の改善法が提案されている．たとえばIOメモリをシステム空間の一部にするとか，コピーと等価な動作を仮想空間を切り替えることにより実現するなどである．またチェックサムの計算をスーパコンピュータのベクトル演算で並列化したり，メモリのコピー動作とチェックサムの計算を同時に行うことでオーバヘッドを見かけ上はなくす工夫もある．さらにヘッダ部の処理の最適化が研究されている．このような研究の例としてJacobson[7]がある．

各種の工夫を凝らすことにより処理が最適化されると，残るのはオペレーティングシステムのオーバヘッドになる．このオーバヘッドはTCP/IP以外のプロトコルにも同様に作用するから，たとえば最適化されたTCP/IPと，高速化を前提としたプロトコルであるXTP（eXpress Transfer Protocol）をCRAY Y-MPの実装例で比較したところ，実効速度の上で差がほとんどなかったという報告もある[8]．

4.4 順序番号が不足する問題

TCP/IPに固有の問題として，やや細かい注意をしておこう．TCPのパケットの中には順序番号（sequential number, 通し番号ともいう）の欄がある．順序番号はTCPによって運ばれるストリーム中のデータを順番に数えた数字である．TCPのパケットは可変長であり，順序番号はTCPのデータ部に含まれる最初のデータの番号を示している．

この欄は32ビット長であるから相当に大きな数であると思われるかもしれないが，超高速のネットワークで次々にTCPのパケットを生成していくと，ある時間が経過すると順序番号が一巡してしまう．**図-4**にTCPパケットのヘッダを掲げる．ヘッダとはデータの本体の前に付加される管理用の情報のことである．

ここで問題となるのは，パケットはいつまで存在するか，ということである．つまり同じ番号を持ったパケットが同時に存在する可能性があるか，という問題である．実はTCPのパケットは，一段下位のプロトコルであるIPパケットの中に納められて転送される．そこでTCPパケッ

トの寿命はIPパケットの寿命に依存する．図-5にTCPとIPとの関係を示す．

さてIPパケットには，まさに寿命（TTL：time to live）という欄がある．図-6にIPパケットのヘッダを示す．TTLは最大255の値をとれる．TTLの単位は標準規格上は秒であるが，実際にはネットワークの中でルータ（中継装置）を通過するときに1ずつ減算される．いずれにしてもTTLの値が0になると，そのIPパケットは破棄されてしまう．そこで問題はIPパケットのTTLが0になるまでの間に順序番号が一巡するか否か，である．

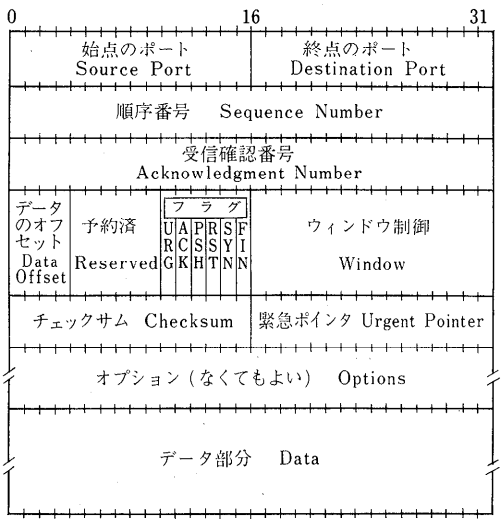


図-4　TCPパケットのヘッダ

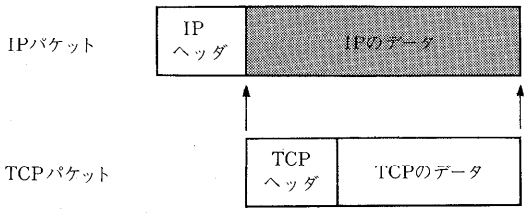


図-5　TCPとIPの関係

大体の見当をつけるためにTTLの値が15であるとし，実際に15秒が寿命であると仮定して計算をしてみる．32ビットの順序番号の単位はバイトであるから，32ビットで区別されるデータは4GB（ギガバイト）ある．つまり32Gb（ギガビット）である．これが15秒の間に一巡するのは，32 Gb/15 sec＝2.13 Gbps（毎秒ギガビット）の場合である．TTLの最大値である255の場合にも，これが実際に255秒であるとすると，32 Gb/255 sec＝0.125 Gbpsとなる．この数字は，今後の高速ネットワーク時代には，あまり十分な数字とはいえない．順序番号の欄の桁数を増やす必要がある．

もっともTTLの実際の計算は実時間の1秒ではなく，ルータを通過する際に1ずつ減算されることが多い．たとえば隣接するルータの間が10 msかかるとする．この場合にTTLが15ならばTTLが0になるまでに150 msかかり，TTLが255ならば2550 msである．上と同様の計算をすると，それぞれ213 Gbps, 12.5 Gbpsとなる．この数字ならば今日のコンピュータネットワークから見て十分な数字かもしれない．

上のTCPの順序番号の他にIPパケットのID（フラグメント識別子）の欄も16ビットしかない．この部分にも一応注意が必要であるが，話が細部にわたるため本稿では割愛する．

5．今後の研究課題

前章で述べたように，超高速のコンピュータネ

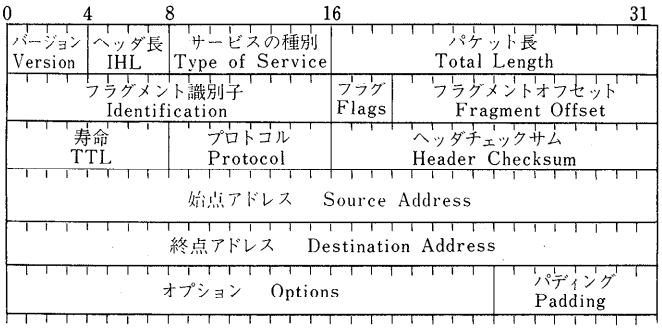


図-6　IPパケットのヘッダ

ットワークを実現して行くためには，プロトコルの標準規格を高速にたえるものにし，そのコンピュータによる処理の最適化をはかる必要がある．

ただし，次のような点にはぜひ注目しておきたい．本稿で述べてきた高速ネットワークの問題点ならびに解決法は，通信が送信側と受信側で構成されるような単純なモデルである．その1対1の通信において光ファイバの速度が使い切れないからといって，ファイバの帯域が無駄になるわけではない．なぜならコンピュータネットワークは同時に多数のユーザが利用できる．やや低速の利用者でも多数が集まれば，結局大きなトラフィックとなる．

この考えを進めれば，一人の利用者，あるいは一つのプロセスでも，通信のチャネルを多数同時に使うことができる．実際に通信の速度を劇的に改善する工夫は，多くの場合に何らかの並列性を利用している．そこで今後は，従来の通信のモデルに拘束されずに，新しい通信のモデルを構築する方向を考えるべきである．

本稿の中で用いた電車の例を再び使うと，コンピュータネットワークにおいては仕事やデータを複数の乗客に託してもよいのである．その処理はコンピュータにとって比較的簡単である．

6. おわりに

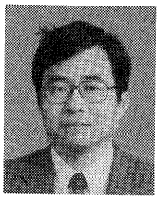
これまで高速のネットワークにおいては，その速度の数字のみが尺度として用いられてきた感がある．本稿では，遅延時間が無視できない場合には，実効速度が遅延時間の数値によって大きく支配されることを説明した．

さらに高速ネットワークを実現するために注意すべきいくつかの要因についても述べた．本稿が，高速ネットワークの実現のために多少とも役に立てば幸いである．

参考文献

1) Davidson, J. M.（後藤，村上，野島共訳）：はやわかり TCP/IP, 共立出版，p. 114 (1991).
2) Malamud, C.（後藤，村上，野島共訳）：インターネット縦横無尽，共立出版，p. 260 (1994).
3) Kugimoto, T., Murakami, K., Amagai, Y., Oka, A., Itoh, M., Goto, S. and Itoh, M.: An Experience with a Long Fat Pipe, The 9th International Conference on Information Networking (ICOIN-9), pp. 535-540 (1994).
4) 天海，村上，釘本，岡，伊藤，後藤，伊藤：長距離超高速インターネットの特性解析，[An analysis of Behavior in a High-speed Long-distance Internet]，情報処理学会マルチメディア通信と分散処理ワークショップ，pp. 149-156 (1994).
5) Clark, D., Jacobson, V., Romkey, J. and Salwen, H.: An Analysis of TCP Processing Overhead, IEEE Communications Magazine, pp. 23-29 (June 1989).
6) Kay, J. and Pasquale, J.: Measurement, Analysis and Improvement of UDP/IP Throughput for the DECstation 500, USENIX Conference 1993 Winter, pp. 249-258 (1993).
7) Jacobson, V.: Some Design Issues for High-speed Networks, Networkshop'93 (1993).
8) Chang, Y.: The Experience of Porting the XTP Protocol onto a Cray Y-MP, VISTAnet document.
9) 村上健一郎：超高速インターネット，第16回学際研究会議「ネットワークの世界」Session 3, pp.1-19 (1995).

（平成7年4月3日受付）



後藤 滋樹（正会員）

1948年生．1971年東京大学理学部数学科卒業．1973年同大学院修士課程修了．現在，NTTソフトウェア研究所広域コンピューティング研究部長．工学博士．人工知能ならびにコンピュータネットワークの研究に従事．著書「PROLOG入門」（サイエンス社1984），「記号処理プログラミング」（岩波書店1988）．訳書「はやわかり TCP/IP」（共立出版1991），「インターネット縦横無尽」（共立出版1994）．電子情報通信学会，人工知能学会，ソフトウェア科学会，応用数理学会，IEEE, ACM, Internet Society 各会員．

村上健一郎（正会員）

1955年生．1979年九州大学工学部情報工学科卒業．1981年同大学院修士課程修了．現在，NTTソフトウェア研究所主幹研究員．インターネットパラダイム，記号処理計算機などの研究に従事．著書「インターネット」（岩波書店1994）．訳書「はやわかり TCP/IP」（共立出版1991），「インターネット縦横無尽」（共立出版1994）．電子情報通信学会，ソフトウェア科学会，ACM, Internet Society 各会員．