CN4－2
(1976. 1. 28)

コンピュータ・ネットワークにおける NCP実現上の問題点

鍛治勝三
(財)日本情報処理開発センター

1. はじめに

1960年後半にコンピュータと遠隔地にある端末装置を結ぶコミュニケーション・システムの概念が出現して以来，数多くのタイム・シェアリング・システムやオンライン・システムが開発された。

1968年に開発を開始したARPAネットワークが成功したのをきっかけに各国にかなり多くのコンピュータ・ネットワークが出現した。これらのコンピュータ・ネットワークの多くは、データ伝送と交換能力を持つサブネットを採用したが、急激に発達したため、その国の電電公社が構築するデータ交換システムがまに合わなかった。その結果として，これらのネットワークの設計者自身が自らの交換システムを構築している。

商用のパケット交換ネットワークとしては、1971年にスペインの電電公社CTNE(COMPANIA TELEFONICA NATIONAL DE ESPA)が最初のサービスを開始した。Trans－Canada Telephone SystemがCCITTの勧告に準拠した方式のDatapacシステムを1976年にサービスを開始することを発表している。

ARPAネットワークが構築されてから6年経った。この間にパケット交換ネットワークに関してはかなり研究も進み、実用上問題がなくなってきている。しかし、ネットワークに接続するためにHostに存在するNetwork Control Program(NCP)に関しては，まだ初期段階を出ていない。

以下では、Hostとサブネット間結合とNCPの検討及び測定について述べる。ここでの調査対象のコンピュータ・ネットワークあるいはパケット交換ネットワークとしては、表1に示した代表的なコンピュータ・ネットワークあるいはその一部であるパケット交換ネットワークの7つである。

2. HostとPSPとの接続方法

まず最初に、Hostを既存のPacket Switching Processor (PSP)にどのようにして接続するかを検討する。接続媒体としては、実験用ネットワークではPSPがHostと同一構内にあるのでチャネル間結合を多く用いられ、一方商用ネットワークではPSPが遠く隔れた電話局内に設置されているので回線結合が用いられる傾向が強い。

ここではチャネル間結合について考えてみる。Hostの標準システムとしてチャネル間結合がサポートされていれば問題はあまりないが、サポートされていない場合には特別なI／Oルーチンを新たに作成しなければならない。I／Oルーチンのプログラミングは非常に複雑である。チャネル間結合アダプタ(CCA, Channel-to-Channel Adaptor)が標準品でない(特に他メーカー)場合，障害時の障害原因の切分けなどの保守作業が困難である(図1(1)参照)。

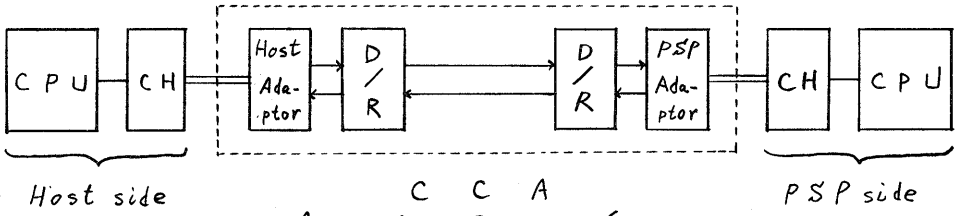
一方、Hostを直接PSPに接続するのではなく、間にミニコンピュータ(N

表1　HOST／パケット交換プロセッサ（PSP）

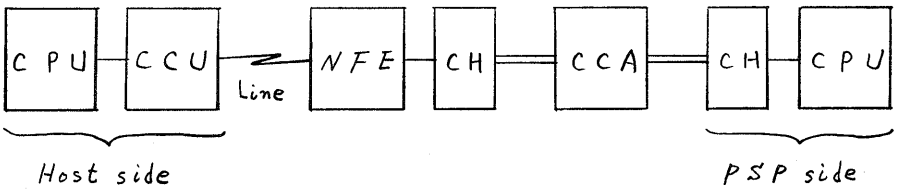
ネットワーク名 / 比較項目	JIPNET	ARPANET	
		Local/Distant Host	Very Distant Host
稼動開始年	1974	1969	
地域	JIPDEC構内	米国全土およびヨー	ロッパの一部
開発主体	JIPDEC	米国国防省　ARP	A
目的	JIPDECの3Host を結ぶ実験的研究用ネットワーク	全米の主要大学，研ネットワーク，ハワ	究機関を結ぶ研究用イとヨーロッパへ拡張
パケット交換プロセッサ（PSP）	NEAC 3200／50	DDP 316／516 新プロセッサ開発中	
ノード数	3	58（内TIP 24台）	
Host 数	3	88	
PSP間伝送速度（kbit/sec）	48	50	
接続媒体	チャネル	チャネル	回線
通信方式	半2重（コンテンション方式）	全2重	全2重
伝送速度（Kbit/sec）	1,000	100	1.5～230.4（通常：9.6）
伝送の最小単位（bit）	8	1（IMP padding convention：16）	16
伝送方式	8ビット並列	ビット直列	ビット直列
同期方式	ブロック	非同期（bit-by-bit）	同期
誤り制御	無	無	CRC（24bit） Positive ACK
データ・リンク確立コマンド	Unlock Program Switch	Ready Indicator On	Line Passive State
データ・リンク切断コマンド	Lock Program Switch	Ready Indicator Off	—
伝送制御手順	—	—	Particular Binary Synchronous Communication
最大メッセージ長（bit）	9,216	8,095*1	8,096（1,008分割）
ヘッダ長（bit）	32	72*1	72
備考	PSP：Packet Switching Processor	*1：先頭の32ビットはleaderであり，先頭の72ビットはHost-to-Hostメッセージのヘッダである。	HostとIMPはパケット単位で送受を行なう。第1パケットは16ビットのControl Wordと32ビットのleaderから成る。第2パケット以降は，先頭に16ビットのControl Wordがあり，次に最大1008ビットのセグメントから成る。メッセージは最大9パケットに分割されて授受される。

インタフェース比較一覧表

CYCLADES (CCIGALE)	CTNE	DATAPAC	
		HDLC	BSC
1974	1971	1976（予定）	
フランス	スペイン	カナダ	
IRIA	CTNE(電電公社)	Trans-Canada Telephone System	
フランス政府，大学研究機関等を結ぶ実験的研究用ネット	商用パケット交換ネットワーク	商用パケット交換ネットワーク	
MITRA-15	UNIVAC418 Ⅲ, IBM 3968	—	
12（含Concentrator 6台）	2	—	
20	8		
48, 19.2, 9.6, 4.8	4.8	—	
回線	回線	回線	回線
全2重	全2重，半2重	全2重	半2重
4.8, 48	4.8	—	—
8	8	—	8
ビット直列	ビット直列	ビット直列	ビット直列
同期	同期	同期	同期
CRC (16-bit)	CRC (16-bit) または LRC+VRC Alternate ACK	CRC (16-bit) Control Field (8-bit)	CRC (16-bit) Alternate ACK
—	Open Command	Request Connect	ENQ bid (Contention basis)
—	Close Command	Request Disc.	EOT
Transparent Binary Synchronous Mode Procedure, CCITT V24	Transparent Procedure or Non-transparent Procedure	HDLC (High-level Data Link Control)	BSC (Binary Synchronous Communication)
2040	2040	2000*1, 2040*2	2000*1, 2040*2
96	88	80*1, 40*2	80*1, 40*2
		*1：Virtual Callの場合 *2：Datagramの場合	*1：Virtual Callの場合 *2：Datagramの場合



(1) Channel-to-Channel Adaptor



(2) Network Front-End Processor

Fig.1 The Interconnection of a Host and a PSP

FE, Network Front-End processor)を介して接続する方法がある。既存のコンピュータ・システムでは既に通信制御装置(CCU, Communication Control Unit)が設置され、そのI/Oルーチンもサポートされているので、CCU-NFE間は回線接続、そしてNFE-PSP間はCCA接続とする(図1(2)参照)。NFEとして5000ドル位の非常に安いマイクロ・コンピュータを使えば、CCAサポートの特別なI/Oルーチンを作るよりも安くなり、ソフトウェアの保守も不必要となり、信頼性もあがり、同時にハードウェア保守も簡単になる。

またCCA接続で注意しなければならないことは、ネットワーク・プロセス間で授受されるメッセージが数十バイト以上でも、Host-PSP間ではメッセージを分割してセグメント単位にやり取りする場合には最終セグメントは1バイトになる可能性があり、Host側のチャネルI/Oスーパバイザは1バイトでも転送できなければならない。もしできないならばCCAによるpaddingやHost-PSPプロトコル及びHost-Hostプロトコルでこの対策を考慮されていなければならない。

3. HostとPSPとの接続事例

JIPNETのIMP[*1]とFACOM230-75 HostはCCAで接続されている。CCAの仕様を以下に示す。

通信方式:半二重;転送方式:8ビット並列・ブロック転送;転送速度:1Mビット/秒;転送制御用コマンド:Write Data with Interrupt(WDI), Write Data(WD), Read Data(RD), Sense, No Operation(NOP), Lock Program Switch(LK), Unlock Program Switch(ULK), その他

*1: Interface Message Processorの略で、PSPに相当する。

ＣＣＡは半二重であるが、両系に対し特に制御局／従属局という関係をもうけず、コンテンション方式を採用している。コンテンション方式であるので、双方が同時にＷＤＩ／ＷＤコマンドを発信した場合、コマンドの衝突が起る。この場合ＪＩＰＮＥＴのＨｏｓｔ－ＩＭＰプロトコルでは、ＩＭＰ側が送信権を持ち、再度ＷＤＩコマンドを発し、一方Ｈｏｓｔ側は受信状態に入り、ＩＭＰ側よりのＡｔｔｅｎｔｉｏｎを受け付けてからＲＤコマンドを発信し、実際のデータ転送が行なわれる。ここで問題になるのは、ＩＭＰ側のリトライが非常に早く行なわれるので、ＮＣＰがＡｔｔｅｎｔｉｏｎ受付用のＥＣＢ（Ｅｖｅｎｔ Ｃｏｎｔｒｏｌ Ｂｌｏｃｋ）を再度登録する[*1]のがまにあわず、ＩＭＰ側のＡｔｔｅｎｔｉｏｎ割込みはＩ／Ｏスーパバイザによって捨てられてしまい、ＮＣＰにはそのＡｔｔｅｎｔｉｏｎがｐｏｓｔされないので両系はデッドロックの状態に入ってしまう。ＩＭＰ側のＡｔｔｅｎｔｉｏｎとＮＣＰ側のＷＤＩコマンドがモニタ内ですれ違うと、すでにＡｔｔｅｎｔｉｏｎがｐｏｓｔされている状態でコマンドの不整合となり、その時既にｐｏｓｔされていたＡｔｔｅｎｔｉｏｎと次のＩＭＰからのリトライによるＡｔｔｅｎｔｉｏｎとが続けてくることになり、Ａｔｔｅｎｔｉｏｎが１つ余分に入って来たことになってしまう。コマンド不整合後のこれらの対策としては、ＩＭＰ側のリトライを数ミリ秒遅らせるようにしている。コマンドの衝突はモニタ内部，チャネルーアダプタ間，アダプタ内部，アダプターＩＭＰ間などの各所で起るが、ＪＩＰＮＥＴの第一バージョンではコマンドの衝突時のステータスが充分にチェックされていなかった。またコマンド衝突の場合の大部分において、Ｉ／Ｏスーパバイザはハード障害とみなし、エラー・ロギングを収集しており、衝突が頻繁に起るとエラー原因をディスクへ書き出すのが間に合わなくなり、コアー・バッファが無くなり、システムをダウンさせていた。これらの障害に対しては、第二バージョンでは対策がとられている。

４．ＮＣＰの常駐場所

ＮＣＰをＨｏｓｔ内に常駐させると既存ＯＳとのインタフェースが非常に複雑となるので、ほとんどすべてのネットワークに関連するホスト・ソフトウェアをＨｏｓｔコンピュータそれ自身よりもむしろＮＦＥに常駐させた方がよい。またＮＣＰをＨｏｓｔに常駐させることによってＨｏｓｔ側に約１００Ｋバイト位のメモリ容量と処理時間のオーバヘッドが生じる。ＪＩＰＮＥＴの測定によるとＮＣＰがフル稼動している場合はＣＰＵバウンドの状態になり、ＮＣＰのＣＰＵ優先順位が高いのでネットワークに関係のないジョブに対してＣＰＵリソースがディスパッチされない。

ＮＣＰをＮＦＥに常駐させ、ネットワークに加入しているすべてのＨｏｓｔが同じ（標準）ＮＦＥを採用すれば、ネットワーク・ソフトウェアは一通になり、開発コストが安くなる。

Ｈｏｓｔに常駐しているＮＣＰがユーザ・プログラムではなく制御プログラムとして作成されている場合は、ＮＣＰの障害が即ＯＳの障害になることが多く、作成，デバック，そして保守・改良が困難である。一方ＮＣＰがＮＦＥに常駐していれば作成も容易で、保守や改良もＨｏｓｔシステムのローカル運用に対し何ら影

---

*1：Ａｔｔｅｎｔｉｏｎ受付け用ＥＣＢをＳＥＴＥＣＢマクロ命令により登録し、Ａｔｔｅｎｔｉｏｎが入るとそのＥＣＢにｐｏｓｔされる。Ｐｏｓｔされた ＥＣＢは再度登録しないと次のＡｔｔｅｎｔｉｏｎが受け取れない。

響を与えずに行うことができる。

ネットワーク・ソフトウエアが一通りになると、ネットワーク・プロトコルの改定もずっとやりやすくなる。加入しているHostがたくさんになり、NCPがすべてのHostも内に常駐している状態では、ネットワーク・プロトコルの改定は不可能となり、複雑で不効率なプロトコルに甘んじなければならなくなってくる。

NCPをNFEに常駐させた場合は、新たにHost－NFEプロトコルが必要となってくるが、これはHost－PSPプロトコルやHost－Hostプロトコルに比べてずっと簡単である。しかしながらプロトコルの重複を増加させる欠点が生じる。

現在JIPNETで作成されているNCPはすべてHost内に常駐している。したがって、ここでは検討をより具体的にするため、NCPはHost内に常駐していると仮定する。

5. 既存OSの検討事項

NCPの作成においては、特に既存OSとのインタフェースや機能を充分に検討し、NCPの肥大化によるHostの過度の負担を防ぎ、かつネットワークに関係せぬローカル・ジョブへの影響を最少限におさえなければならない。

NCPも1つのプロセスであるので、既存OSに各々独立したジョブステップの間での通信機能が備っているか否か、また備っているとしたら、それはユーザプロセスで使うことができるか否かによって、NCPの位置づけは、次の3通りが考えられる。

① 制御プログラムの一部とする
② ユーザ・プログラムとする
③ ①と②の中間的なサブモニタとする

既存OSにジョブステップ間通信機能が無い場合には、新たにこの機能をサポートするプログラムを作成しなければならない。ロール・イン／ロール・アウト機能やバーチャル・メモリを採用したOSでは、受信側ジョブステップが実主記憶上に存在しない場合があるので、その処理が非常に複雑であり、この機能をユーザ独自で開発するのは不可能に近いと考えられる。

これらの内、①はある意味では理想であるが、OSの設計時にNCPの組み込みが考慮されていない場合はあまり良いものはできないと考えられる。ネットワーク結合のために増加した制御プログラムが、そのHostのみによるローカル・ジョブのオーバヘッドの増加をもたらす可能性がある。また既存OS部分に改造を加えねばならぬ可能性も強く、いわゆる標準OSではなくなるため、その後のメーカーのOSメンテナンス・サポートが期待できぬ恐れがある。

これに対し、②のユーザ・プログラムとして作成する形はユーザとしてはもっとも作成が容易であるが、独立したプロセスの起動・停止や制御・監視ができない可能性があり、ネットワーク・サービスに関連する既存OS内の制御表を参照するなど、かなり無理が生じる可能性が強い。

以上のような点を総合すると、結果的には③のサブモニタの形でNCPを作成することがもっとも望ましい。サブモニタ形式を採用したOSでは、あるサブモニタの障害が別のサブモニタへ影響を与えず、完全に自立している。

いずれにしろこれらの選択は既存OSの機能および性格によって決定づけられ

てしまい選択の余地がない可能性がある。今後新たに開発されるOSは他のコンピュータとのコミュニケーションを前提として設計されなければならない。

ユーザ・プロセスは直接Host－Hostプロトコルのコマンドを使うのではなく、マクロなコマンド（NCPサービス・マクロと呼ばれる）を通じてNCPにサービスを依頼するのが普通である。NCPサービス・マクロを既存OSのマクロ命令と同じようにスーパバイザ・コール（SVC）で実現できることが望ましい。そのためには既存OSの設計時にSVCコードの何番から何番までをユーザに解放するかを完全に決めておき、SVCルーチン作成の詳細マニュアルを準備しておかなければならない。OS/360[1]ではSVCコード255～200まではユーザのために予約されている。

ネットワーク・プロセスがコネクションを切断せずに異常終了した場合、既存OSのターミネータがそのことをNCPに通知してくれれば、このことをリモートNCPに通知し、コネクションを自動切断することができる。ターミネータが通知してくれず、しかもタイム・アウト処理がなされていない場合にはリモートプロセスは永久に待つことになる。

以上をまとめると、最低限以下の機能が既存OSに備っていることが望ましい。

① NCPとプロセスとの間で非同期に情報の受け渡しが可能なこと
② ビット・ストリング・データのトランスペアレントな送受が可能なこと
③ NCPがプロセスの発生，監視および強制終了をおこなえること
④ プロセスの実行に必要なファイルを動的に割付け・返却が可能なこと
⑤ 通信制御プログラム中の端末制御プログラムが伝送制御手順毎に完全に独立したモジュールであること
⑥ アカウンティングのためにリソース使用情報をNCPが取り出せること
⑦ ローカル・リソース全体の使用状況がNCPから参照できること

5．パフォーマンスの測定

(1) ファイル転送のパフォーマンス測定

1ブロックが576バイトのファイルを1hop*1離れた2つのHost間で転送した場合の測定によると、JIPNETのFACOM230－75からHITAC8450へのスループットは25.7Kビット／秒，round trip time*2（RTT）は179ミリ秒である。F－H間では別に2hopの迂回ルートもあるので、それも併用するとスループットは30.1Kビット／秒に増加し、RTTは153ミリ秒に減少し、adaptive routingの効果が現われている。

同様な実験[2]が1970年ごろのARPANETを使ってSRIのXDS940からUtah大学のPDP－10で行なわれており、スループットは26.3Kビット／秒，RTTが175ミリ秒が測定されている。その後IMPが改善されており、現在測定すればスループットは30.3Kビット／秒，RTTは152ミリ秒になることが推定される[3]。

2つのネットワークの測定結果を比較するとほぼ等しくなっている。これは両ネットワークとも同じ規模の回線，IMPコンピュータそしてHostもコンピュー

---

*1：通過する回線の個数

*2：ここでは、プロセスがメッセージ送信要求を出してから、それに対するacknowledgeがプロセスに返されるまでに経過した時間と定義する。

タを使っているためで、プロトコルの差異は影響していないことがわかる。

(2) ネットワーク・パフォーマンスの測定

ＪＩＰＮＥＴを使って図２のような迂回ルートのあるトポロジで２つのＨｏｓｔ間で２つの異なった大メッセージ長に対してコネクション数を変化させて総合パフォーマンスを測定した（図３参照）。

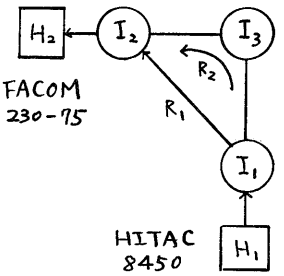


Fig.2 Network Topology

最初のケースは、パケットのオーバヘッドが最小になるようにメッセージ長を最大パケット長（full packet）の１１５２ビットにした。そのようなメッセージに対する最大スループットはルートR1のみ使用した場合は約４１Ｋビット／秒であり、active connectionが４個でほぼ上限に達している。

これはサブネットのフロー・コントロールが原因であり、発信地ＩＭＰと目的地ＩＭＰ間にパイプという論理的なパスを張り、そのパイプの容量を制限して輻輳を防止している。サブネット内ではメッセージを単一パケット・メッセージ（１１５２ビット以下）と複数パケット・メッセージ（９２１６ビット以下）とに分けて、単一パケット用のパイプ（Ｓ－パイプ）を４本と複数パケット用のパイプ（Ｌ－パイプ）を１本設けている。各パイプはコンカレントにメッセージを流すことができ、複数パケット・メッセージは目的地IMPにバッファを確保した後Ｌ－パイプを通し、一時に１個だけ流すことができる。

Ｓ－パイプは４本であるので、４本以上のコネクションを張ってコンカレントにメッセージを送信しても、Ｓ－パイプのバウンドで抑えられて、スループットは増大しない。

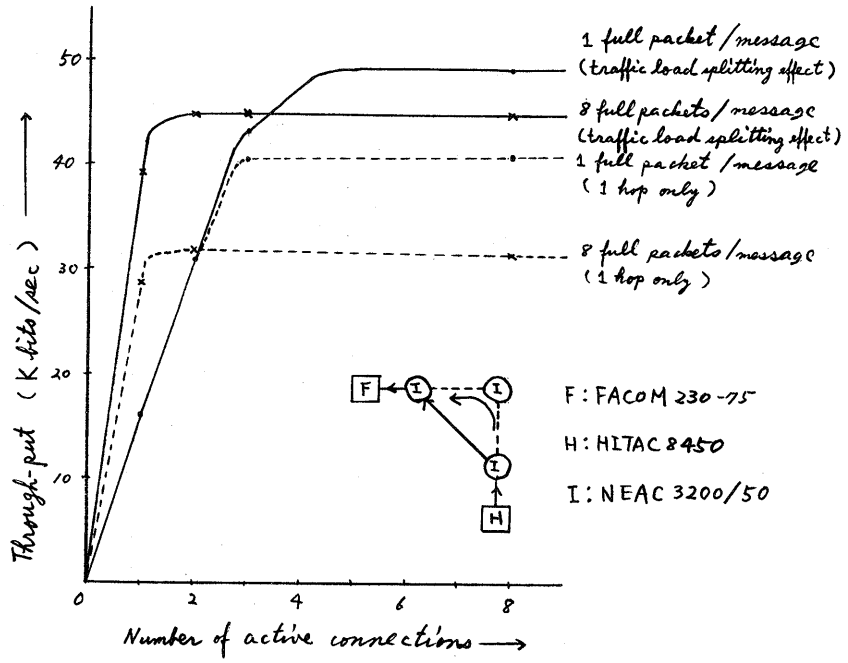


Fig.3 Maximum Through-put Measurements

ルート R_1 と迂回ルート R_2 を併用した場合の最大スループットは約50Kビット／秒に増加する。

これはサブネットのadaptive routingの効果である。Adaptive routingとしてはShortest Queue＋Biasを採用しており、最短ルートの出力回線の送信待ちキューが長くなると迂回ルートの出力回線が選択される。

第2のケースは、メッセージを最大メッセージ長（8-full packet）の9216ビットにした。そのようなメッセージに対する最大スループットはルート R_1 のみを使用した場合は約32Kビット／秒であり、非常に効率が悪い。これはレーパイプが1本しかないため、別のコネクションのメッセージはレーパイプが空くまで送信が待たされるためである。ルート R_1 と迂回ルート R_2 を併用した場合の最大スループットは約45Kビット／秒に増加する。これはメッセージの一部（パケット）が迂回ルートを通り、全体としてのRTTを減少させており、レーパイプが早く空き状態になるためである。

ロング・メッセージの最大スループットは1970年ごろのARPANETの約70Kビット／秒[2]に比べると非常に悪くなっている。このころのARPANETではロング・メッセージを特別扱いせず、リンク毎にフロー・コントロールを行っていた。しかし特定のHostへのactiveリンクが5本を越えるとreassemble lockupが起りうることが判明して、その後フロー・コントロールが強化されたためたがんもっと低い値（31.1Kビット／秒）[3]になると推定される。

次に単一コネクションでメッセージ長を変化させてRTTを測定した（図4参照）。

RTTは、1フル・パケット・メッセージの場合1hopでは68ミリ秒，2hopでは100ミリ秒である。サブネットの測定から推定すると、両Hostの送受信処理に約32ミリ秒がかかっている。

8フル・パケット・メッセージの場合1hopでは298ミリ秒，2hopでは337ミリ秒である。1hopの場合、2hopの迂回ルートを併用すると217ミリ秒に減少する。この217ミリ秒は1hopルートに5パケットを流した場合に相当し、このことから1hopルートへ5パケット，2hopルートへ3パケットを振り分けたことが推定される。

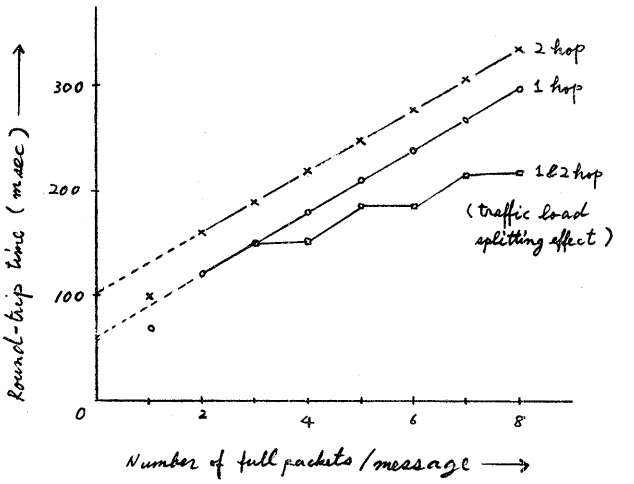


Fig.4 Minimum Round-Trip Time Measurements

図4からRTTの式 $Tr(h,p)$ を求めると、

＜1hopの場合＞ $Tr(1,p)=60+30p$ (1)

＜2hopの場合＞ $Tr(2,p)=100+30p$ (2)

＜1&2hopの場合＞ $Tr(1\&2,p)=88+16\times\{p+mod(p,2)\}$ (3)

である。ここでhはhop数，pはメッセージに含まれるパケット数である。各々

の式の定数項は第1パケットとRFA[*1]などのフロー伝送制御パケットが目的地IMPへ到着するのに必要な時間の和であり、hop数の関数である。pの係数30は1フル・パケットが1hopだけ前進するのに必要な時間であり、30pは最終パケットが先頭パケットよりもどの位遅れて目的地IMPへ到着するかを表わしている。

以上から次のRTTの近似式Tr(h,p)が導き出される。

$$Tr(h,p) \fallingdotseq 60h + 30p \quad (4)$$

p=7（8064ビット）の場合は、

$$Tr(h,7) = 60h + 210 \quad (5)$$

であり、ARPANETのフル・メッセージ（8063ビット）の場合の式[4]

$$Tr^{A}(h,8) = 60h + 200 \quad (6)$$

とほとんど一致する。

式(3)からadaptive routingによりp=2nのRTTがp=2n-1のRTTに等しくなることが解かる。

6. おわりに

以上でNCPの実現上の問題点，検討および測定について述べたが、今後はコンピュータ・ネットワークをコンピュータ・ユーティリティ（バーチャル・ネットワーク）へと発展させるための共通ネットワーク制御言語（Network Job Control Language，NJCL），各Hostに散在している共用データ・ベースへのアクセス（Distributed Data Base Access，DDBA），複数Hostを併用して実行されるアプリケーション（Distributed Application）などの開発を容易にするためのサポート・プログラムが必要となってくるであろう。

本論文作成のために、JIPNETの測定に協力してくれた開発第一課の各位に深く感謝の意を表わします。

最後に、紙面の関係で本文で直接引用した文献のみを示し、参考にさせて頂いた文献の一覧表は発表当日に配布する予定である。

参考文献

[1] IBM，"IBM System/360 Operating System：System Programmer's Guide，" Form C28-6550-6

[2] G. D. Cole，"Performance measurements on the ARPA computer network，" Proc. Second Symposium on Problems in the Optimization of Data Communications Systems，pp. 39-45，Oct. 1971

[3] J. M. McQuillan，et al，"Improvements in the design and performance of the ARPA network，" FJCC '72，pp. 741-755

[4] D. C. Wood and R. K. Treham，"An assessement of the performance of packet switching networks，" EUROCOMP '75，pp. 1-11

---

*1：Request For Allocationの略で、発信地IMPが目的地IMPにRFAパケット（80ビット）を送り、目的地IMPにバッファを確保する。