コンピュータ　ネットワーク

—Dr. Robert E. Kahn の講義より—

昭和 49 年 3 月

財団法人 日本情報処理開発センター

はじめに

昭和49年1月中旬，(財)日本情報処理開発センターの招きにより来日された米国国防省ARPA(Advanced Research Projcet Agency)のDirector, Dr. Robert E. Kahnにより「コンピュータ・ネットワーク」に関する講義がおこなわれた。

本資料は，その15時間にわたる講義の概要を日本文に翻訳したものである。

講師Dr. Robert E. Kahnは，データ・コミュニケーションの専門家であり，BTLから一時MITにおいて教鞭をとっていたこともあるが，1966年よりBBNにおいてARPAネットワークの設計および Distributed Computation の研究を行ない，ARPAネットワークの建設に貢献した有力メンバーの一人であることはよく知られている。1972年国防省のARPAに移り，現在Director of ARPA network の要職にあるとともに国際ネットワークの委員会の会長でもある。

この講義はARPAネットワークにおける幾多の改良の経緯，現状，新しい計画等の詳細にわたる技術的問題を中心におこなわれたが，一般的なコンピュータネットワークの新技術についても言及され，我が国における今後のネットワーク技術にも大きな示唆を与えられた。

本資料は講義の順序にしたがい，そのまま記録したものであるが，内容により，いくつかの見出しを付し，また質問応答は最後にまとめた。

目　　次

Introduction

ＡＲＰＡネットワークの開発は 1960年代の中頃に，部分的な技術に関してはそれ以前に始まったものである。ＡＲＰＡネットワークは全米に多くのコンピューターリソースを置き，それらのリソースを共有出来るように結びつけることによって大変多くの利点を生みだした。即ちデータベース，プログラムならびにハードウエア自体も共有することができるような最良の方法を見出すことができたのである。またこのネットワークの開発を進めて行く上でも各々のマシンをネットワークに結合することにより，さらに良い利用率とサービスが得られるということも見い出した。このことは以前から推察はしていたのだが，実際の確認ができなかったことである。

ネットワークの構成

この講義では，スライドを使用しネットワークがどのように発展してきたのかという過程と現状を説明していきたいと思う。まず，リソース共有として次の３つを挙げることができる。マシンの共有，ソフトウエアの共有，そしてデーターベースの共有である。マシンに関しては，私達は全米にある約２ダース以上の主なマシンを同時に結びつけている。しかしこれらのマシンを結びつけるコモンキャリアーのサービスを見つけることが非常に難しいものであった。実際には1.ダイヤルアップ回線（Dial-up Circuits）2. リース回線（Leased Circuits） 3.マニュアルスイッチ（Manually Switched）および4.ストア・アンド・フォアワード（Store and Forword)のネットワークの４つのみの選択がゆるされていただけであった。最終的には４番目のストア・アンド・フォアワード・ネットワークを採り入れ，我々用のリソースネットワークを開発した。その理由は，まず他の３つを次のように判断したからである。ダイヤルアップ回線はリソース共有という目的で使用するには古すぎ，帯域も狭く，しかも単一コンピューターシステムに非常の多くの連結が必要になってくる。リース回線は連結する場所毎に別個の連結が必要になり費用が高くつく。また，マニュアルスイッチはＡＴ＆Ｔがマニュアルスイッチを４都市で50kb/秒のサービスを提供するというので興味があり，しばらくの間，実施してみたが，速度が遅く，費用も高くつくため途中でやめることとした。結局，最後に，連結に対する必要条件に見合う，ストア・アンド・フォアワード・ネットワークを採用することを決めた。そしてその方法は幾つかのリソースを連結し，Fig１に示すようにその各々のリソースの近くに小型コンピュータを設置することとした。全てのリソースを直接連結するよりもコンピュータを介して連結し，またすべてのペアを結ぶという方式をとらなかったため費用も安くついた。

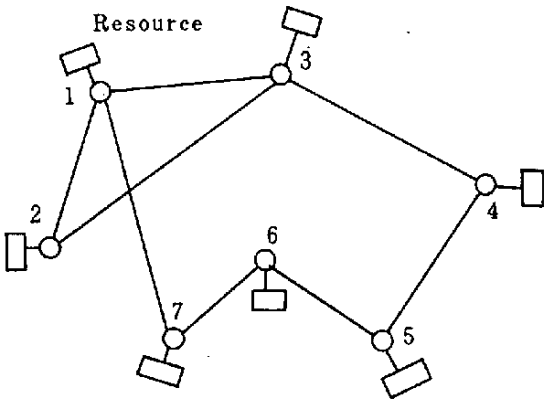


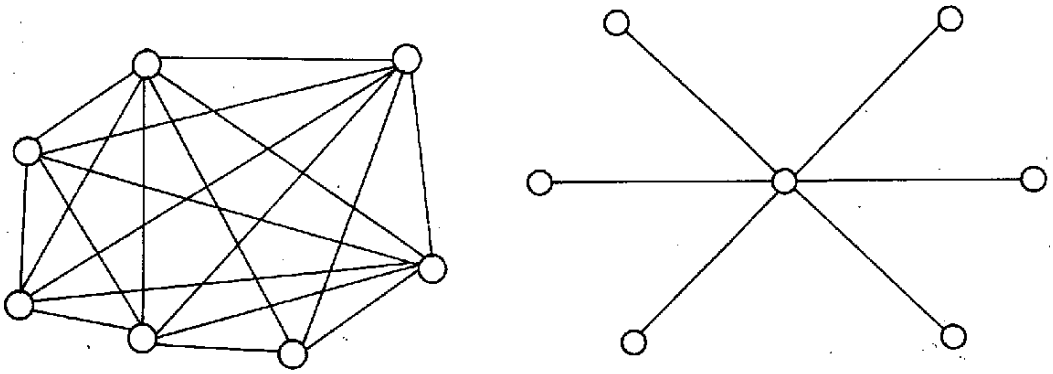
Fig 1

Chart１にコストと遅延（Delay）の関係を示している。この中央下の全結合（fully connected）の2.4 kb/s 回線のネットワークはFig 2の左側に示すようなものであるが，1000

内至10000bitにつき，最低遅延速度が1秒弱，最高が10秒ぐらいである。費用の方は2.4kb/s 回線で20のNode の場合，1Mbitにつき20セントから40セントぐらいである。

ARPAネットで多く扱っているグラフィック画面の情報を送るにはこれでは遅延が長すぎ，その場合1秒に1Mbits ぐらいを送る必要が出てくる。

Chart 1のその左側のスターネットと呼ばれる50kb/s回線のものはFig 2の右側に示すような中央に集中する形をとる。これは$\frac{2}{10}$～$\frac{7}{10}$ 秒の広帯域伝送の低い部分を使って伝送ができるわけである。費用も1メガビットで20～40 セントぐらいであるが，2.4kb/sのネットワークより性能は良く，伝送も確実で信頼できる。中央ノードが作用しなくなれば，他のノードも切れてしまうため，誰も通信することができなくなり，間違いは全く起こらないからである！



全 結 合
(2.4kb)

スターネット
(50kb)

Fig 2

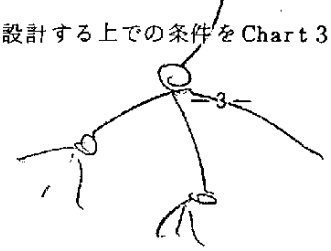
それから全結合の場合で19kbの AT&Tのファシリティーによるネットワークがあり，伝送時間は$\frac{2}{10}$秒から$\frac{6}{10}$ 秒であるが，これにかかる費用はかなり高い。他にもDDDダイヤルアップネットワークがあり，費用も1メガビット4～5ドルかかる。これは2kbのdirect distance styleのものである。またAT&Tが以前4都市に提供していたマニュアルスイッチの50kbサービスもあった。このサービスはダイヤルアップと同じ伝送で，費用はダイヤルアップより少し安いものであった。

さて，我々が開発したネットワークは，$\frac{2}{10}$～$\frac{3}{10}$秒の伝送速度で，費用もメカビットあたり15～18セントぐらいのものである。今まで説明してきた幾つかのネットワークよりも伝送時間も早く，費用も一番安い。

Chart 2に転送量（Block size）に対する実効速度の関係を示す。これによってもARPANETは他のものよりも性能がすぐれている。

以上のような事情から我々が，ストア・アンド・フォアワード伝送とパケット・スイッチを使用してネットワークを構成することとした理由がわかると思う。

なお，このネットワークを設計する上での条件をChart 3に示す。



メッセージの伝送

このARPANET でメッセージがどのようにして伝送されるかを話してみたい。まず，発信側のＨＯＳＴと受信側のＨＯＳＴの間にそれぞれのＩＭＰが置かれる。発信側のＨＯＳＴにあるＩＭＰが伝送するメッセージを分割してそれぞれのＩＭＰに送り出し，受信側のＨＯＳＴにあるＩＭＰに伝送されるとそこで分割されたメッセージが一つにまとめられ，受信側のＨＯＳＴへ送られる。メッセージを受けたＨＯＳＴは受信承認のメッセージを送り返すわけであるが，この場合，受信ＨＯＳＴ側にあるＩＭＰから分割された承認がそれぞれのＩＭＰを通って発信側のＨＯＳＴにあるIMPに入り，一つのメッセージとしてＨＯＳＴに送られるようになっている。この場合，メッセージの許容範囲は６３語ぐらいで16bitから 1008bitまでである。最大のメッセージは 8025bitぐらいである。

例えば，２つのパケットを伝送するとする。まず最初のパケットが発信側のＩＭＰからＩＭＰに送られ，次のパケットが違ったＩＭＰに送られ，受信ＨＯＳＴ側のＩＭＰに両パケットが入り，最終のパケットが入れば，そこで一つのメッセージとして受信側ＨＯＳＴに送られる。この場合，各ＩＰＭでは受取ったそれぞれのパケットに対するＡＣＫを送り，目的地ＩＭＰではメッセージを組み立て，目的地ＨＯＳＴへ渡すと同時に，発信側ＨＯＳＴへＲＦＮＭ（Ready For Next Message）が送られる。

これが，メッセージを送る際の完全に機械化にしたコントロールである。これをChart４に示す。

そして，メッセージの形態をChart５に示し，各ビット数と，ネットワーク中の出発地と目的地をあげておく。

ARPANETの発展の経過

次に，このARPANETの拡張，すなわちその発展について述べてみることにする。1969年9月にまずＵＣＬＡ（University of California at Los Angeles）にＩＭＰを設置，10月にはＳＲＩ（Stanford Research Institute)に，１１月にＵＣＳＢ（Univ.of Calif.at Santa Barbara)，そして１２月にＵＴＡＨ（Univ.of UTAH）に設置し，ネットを結び，このネットワークの基本的な能力を試してみた。それから約半年後，これらの４つのＩＭＰノードに，６つ程加え，この１０のノードを50kb の回線で結んだ。このネットワークでは，何ドルで一秒何ビット得られるかと，そのネットワーク能力と信頼性を研究してみた。そして各地ともbranch exchange techniqueを使って大幅なプログラムを開発した。ここで各地の結果として出てきた最も興味深い点は，計算量が少なくてすみ，伝送が速いということであった。遅延も$\frac{2}{10}$秒以上にはならず，実用可能な結果が得られた。そこで，７１年には，さらに大きくし，７２年には２７のＩＭＰ採用場所を設けた。そして，今まで使用されていたネットワークに更に新しいノードを結ぶことにより一層すばらしいネットワークの効果を生み出すことが出来るようになった。さらに，ワシントンの海軍施設やＭＩＴＲＥ社，Illinois 大学，その他２～３の新規場所もネットワークに加わった。Chart6にその拡張のもようを示す。Chart7は1970年における接続コンピュータである。

このようにＡＲＰＡネットワークは非常に融通性をもっていることがわかると思う。融通性というのは，例えば非常に小さなプロセッサーを入れてこのネットワークの他の場所に接続したり，あるいは，全体的にネットワークから満足した結果が得られなければ自由にネットワークから離れることができるのである。Chart6によると７２年の２４のNode の場合，また２７のNode のネットワークの場合に気付いたと思うが，それらノードの接続がいろいろ違って来ている。

７２年のネットワークの能力は，１日に３千万パケットであったのが，今日では４千５百万から５千万パケットにもなっている。またネットワーク上のリソースも多くなっており，ＰＤＰ－10や360-91 も使っている。1973 年迄には，そのネットワークの地域は東西両岸にその中心をおき，東岸はボストンとワシントンＤＣ地域，西岸はロサンジェルスとサンフランシスコ地域である。（Chart8 ）その他フロリダのＲＭＬ（Range Measurements Laboratory)は，このネットワークを管理する役割を果たし，またアポロ宇宙計画に際しても重要な活動を行なった。

Chart9は，1973年9月当時のネットワークであるがNASA AMESではＰＤＰ－１０を使ってIlliac の大型コンピューターに接続した。ＰＤＰ－10 で受けたプログラムやデータはIlliac Ⅳに移して，実行される。非常に複雑なマトリックス インバージョンやその他の高速演算を必要とするデーターやプログラムをネットワークを通して伝送する。するとIlliac Ⅳがその結果を知らせてくる。ネットワークでIlliac Ⅳをこの様な形で結合してあるのは非常に便利である。

360-91も大型コンピューターの一例で，我々がよくバッチ処理に使用しているものである。Illiac Ⅳはバッチ処理に適していない。他にもハネウェル6180というのがある。これはＭＩＴ

のMULTICSで使われている。ＭＩＴには２つのＩＭＰがあり，一つは6180が接続され，新しい方にはＰＤＰ－10が結合されているが，この古い方も今でも時折使うことがある。新しい方には３台のPDP－１０を接続しており，多分４台目を連結すると思う。

さて現在のネットワークでは４０台以上のコンピューターを使い，このネットワークで使う全リソース額は毎年350万～400万ドルに達している。Illiac Ⅳはローンで恐らく2000万ドルの本体であり，このネットワークでの稼動は１日に２～３時間といったところである。現在ネットワークでは350万～400万ドルのリソースを扱っており，ネットワーク上の４０台以上のコンピューター間で相互に伝送しているが，今後は一層多くのコンピューターが使われると思う。Illiac Ⅳは１台で非常に多くのネットワークのアクセスをもっている。

Chart９の左側にＡＭＥＳからＨＡＷＡＩＩへ伝送しているのがおわかりと思うが，この伝送はハワイ大学と通信衛星を使って地震データ（Seismic Data)を50kbで 伝送している。また図の右側でもＳＤＡＣ（Seismic Data Analysis Center, Virginia）からノルウェーのＮＯＲＳＡＲ（Norwegian Seismic Array)に衛星伝送され，同じく地震データを送っている。ノルウェーには9.6 kb で衛星伝送している。更にノルウェーからロンドンへ伝送しており，この場合に二つの目的をもっている。一つは，アメリカ── ノルウェー── イギリスを結ぶ実験的なネットワーク上で地震データのアクセスをおこなうこと，二番目は衛星伝送するのに必要なプロトコル（protocal)を開発してゆくことである。

ＳＤＡＣもNORWAYの方もCODEX9600 MODEM を使っているが，Chart 10 に示すように 7.2 kb のみを使って残りの2.4kbは地震データーの方にまわし，衛星では統合され9.6kbになる。そして衛星からノルウェーに送られる場合も，ノルウェーのＴＩＰは7.2kb，地震データも2.4kbになる。

ノルウェーからロンドンの伝送の場合もＣＯＤＥＸ　ＭＯＤＥＭを使っている。

ネットワークの使用状況

ここで私が1973年3月からまとめていたデーターを基にしてネットワークの使用状況を述べてみたいと思う。７３年の３月当時一年の予算がネットワークに使われるリソースを含めても約200万ドルであったのが，それ以来，現在に至っては殆んど２倍近くになっており，うまく利用した場合でも350万から400万ドルになっている。このネットワーク内で使用率の高いコンピュータは，ＵＳＣ（Univ.of Southern Calif.）のＰＤＰ－10，ＮＡＳＡ　ＡＭＥＳのIlliacⅣ（含PDP－10），ＵＣＬＡの360／91，BBNのPDP－10，SRIのPDP－10，UCSDの360／75，MITのMULTICS の順である。一番使用率の高いのがＵＳＣ（Univ.of Southern Calif）のＰＤＰ－10システムであり，１日２４時間稼動している。ＮＡＳＡ　ＡＭＥＳのIlliac Ⅳの設置場所では，実際にはＰＤＰ－１０をフロント・エンド（Front－end）として作動させて，その他バロース6700のコンピューターでIlliac Ⅳの計算のコンパイルやアセンブルをやっている。３月にネットワークで使うリソースを見積った時，これらの全設備で４７万ドルであったのが，恐らく今ではその３倍の額になっていると思われる。

ＵＣＬＡ（University of Calif.at Los Angeles）の360－91 コンピューターは，３月の時点で３４万ドル程度の処理がネットワークを通して行なわれたと思われる。ＢＢＮはネットワーク内のＰＤＰ－１０ユーザーにオペレーティングシステムを提供しているが，ここでネットワークを通じて使われるのは全コンピュータ使用の30％~35％位であり，それでも年間17万9千ドルぐらいである。またStanford Research InstituteのPDP－10は15万1千ドルぐらい使われている。ＵＣＳＤ（サンディエゴ）は，36700ドル，ＭＵＬＴＩＣＳでは５万ドルぐらいである。次にこのネットワークを多く利用しているユーザをChart 11に示す。まずこのネットワークの主要ユーザーとしてイリノイ大学があげられる。このイリノイ大学では非常に高度な並列処理研究（Parallel Processing research）を行なっているが，面白いことに以前はイリノイ大学自身で多くのコンピューターを持っていたのだが，ネットワークが使用出来るようになってすぐに，コンピューターを取り除いてしまいネットワークを非常に有効な方法で使って経済性を見出している。したがって大学自体で一つもコンピューターをもっていないイリノイ大学が一番多くこのネットワークを使っていることになり，ネットワークに開口（Port）を設けることだけで良く，200万ドルの価値があるネットワークを３６万ドルぐらいで使っている。

第二番目に使用状況が多いのはNASA AMESでIlliacで別の処理をやっており，３２万８千ドルを費している。続いでRand Corp.が多く２１万ドルを使っている。次にApplied Data ResearchがIlliacのFortranコンパイラー開発の為に１５万１千ドルを費している。我々ＡＲＰＡは内部の目的に７万７千ドルを使っており，ＢＢＮは５万５千ドルを使いシステムサポートをネットワークに与えている。殆んどが，プロパゲートオペレーティングシステム（TENEX）のディバッギングなどに使用している。

フロリダのＲＭＬはARPANET の管理を行なっているわけであるが，非常に大きなプログラムを扱っている。Institute for the Future では１万３千ドルでTeleconference researchを行なっている。これからわかるように２～３の大きなユーザーおよび多くの小さなユーザーはこのネットワークがより信頼性が高くなり，更に使用しやすくなればなるほど，いろいろな活動を増やして使っている。

Chart 11 の右側の項は，ネットワークなしで左側の同じ計算を行なうのにかかる費用を示している。あるいは以前に様々な研究をネットワークなしで行なっていた場合にかかった費用である。イリノイ大学の場合，大学自体のコンピューターを持たないという理由がよくわかると思う。というのは，110万ドルも年間にかかり，ネットワークを使用した場合の３倍以上にも及んでいる。

ＡＲＰＡネットワークに連結されているコンピューターは，おそらく世界で最も優秀なものであろう。これはハードウェアだけでなく，コンパイラー，マトリックスの演算，あるいはエディティングなどを始め，ソフトウェアの点でも，最もすぐれている。したがってこのネットワークを使用する価値は益々上ってゆくわけである。そして，ローカルなコンピュータを使用するよりも安くなるのである。

ARPANET の全コミュニケーション・ラインに対する費用は年間８０万ドルである。

Chart 11 によれば全体のRemote Usageの200万ドルに対し，Local Replacementは 600万ドルであり，$\frac{1}{3}$に費用が節約されたことになる。

ネットワークにかかる費用は，約百万ドルが回線に，百万ドルがオペレーション，開発，デバッキングにかかり，したがってネットワークへの全費用は２百万ドルで，６百万ドル分のコンピューターパワーを得ることができたのである。全体的に言えば，コンピューターに２百万ドル，ネットワークに２百万ドル，合計４百万ドル費したわけで，結局２百万ドル残すことができたわけである。しかし我々の予算の中では，その２百万ドルを他の研究目的にまわすことができた。これは経済面での良い例である。我々のような公共的機関と民間の商業機関とでは，受ける制約というものが違ってくるであろうが商用のネットワークにおける経済面の検討も今後やっていくつもりである。まだ結論は出ていないが，商用の場合でも必要な費用は先程述べた以上にさらに大きな数字にはならないであろう。

Fig 5にネットワークのコスト効果について，Nodeを20，40，60 の場合に分けて，そのコンピュータリソース共有に年間費す費用（コンピューター費用）とコミュニケーションコストの関係を示している。コミュニケーションコストの方は百分率で表わしている。図からわかるように，コミュニケーションコストを１０％とみた場合，約２千万ドルのリソース共有費用で，２０のNode が１０％弱である。４０のNode の場合は 14～15%，60のNode で１７％ぐらいのコミュニケーションコストである。この場合 20のNode が一番安くてすみ効果的である。

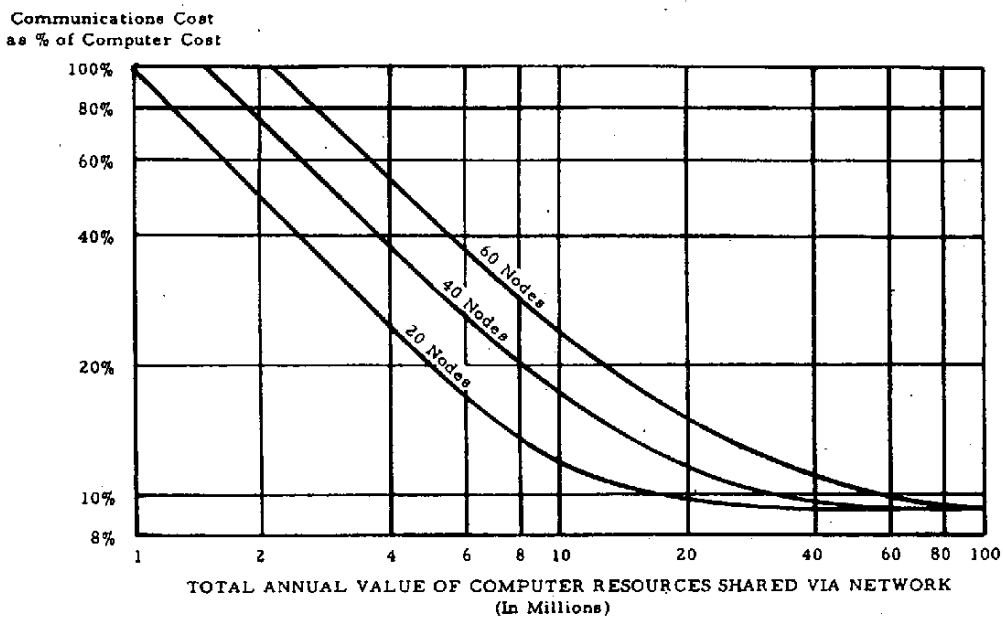


Fig 5

しかしながら，2百万ドルの場合をみてみると，60 Node は約 100%強を示している。これは非常に面白い図である，というのは，バランス的には，コミュニケーションコストをそう下げる必要がなければコンピュータリソースを共有する為に，さらに多くのコミュニケーション設備を使うことができるのである。

メッセージの伝送時間

前述したと思うが，我々が作成したネットワークは平均トランジットタイムが，当初我々が考えた時間より遅いものであった。平均50～100ミリ秒ぐらいである。Chart 12 に示すように初期に我々が幾つかのメッセージ用のトランジットタイム計算をおこなった。そのメッセージというのは，1つは１６ビットという非常に小さなメッセージで例えばACK用のメッセージで，もう１つは1000ビットのパケット，そしてもう１つは8パケットからなるメッセージの3種類である。我々が計算したトランジットタイムは1語のメッセージを送る為の５ミリ秒から，最大のメッセージを送る為の360ミリ秒迄の巾があった。仮定としてはネットワーク内で，何等停滞がなくキューイングのディレイもない。その場合，小さいメッセージの速度の最低は５ミリ秒であったが，最高は，どれぐらいの距離であり，幾つのＨｏｐがあるかということで決まってくるものである。恐らく50Hopsで8パケットのメッセージは，360ミリ秒から195ミリ秒ぐらいの間で得られるであろう。ラウンドトリップの時間は片道（１Way）走行の時間より早いという事実がある。ラウンドトリップの時間というのは送り先からACKが送られてきてそれを受けとる迄の時間で，ACKというのは全メッセージがHost へ入る前に送られてくるものである。ACKは最初のパケットが送られた後戻ってくるわけで，全メッセージが目的地に送られる前にソースに戻るのである。これがＡＲＰＡに対する考え方であり，変えることも可能である。１秒間のメッセージ数については，次のメッセージが前のラウンドトリップを待ってすぐ送り出される状態とすると最高で１秒間100のメッセージを，最低で３つのメッセージを送ることができる。Chart13は，ラウンドトリップの速度とメッセージの長さを対照した図であり，100マイルと1000 マイルの２つに分けてみた。横軸はパケットの数，即ちメッセージの長さである。そして一定の距離をもったネットワーク中のHop数によってどのような変化がみられるかを示している。６本の破線は１から６のHop数による違いを示す。パケットによりかなり急激にカーブが上昇し，それからややゆるやかになる理由は，Host に end-to-end の承認手続が使われているからである。それにかかる初期時間としては１０ミリ秒から最悪の場合は100ミリ秒ぐらいの間である。

ＡＲＰＡＮＥＴの利点

さて，今までネットワークについての多くの話し合いが経済的な面での利点に向けて行なわれてきたわけであるが，我々は従来そういう点に大きなメリットがあたえられるとは思っていなかった分野に非常に大きな利点がみられるということに気付いたわけである。ここに少々そのことについて述べてみることにする。重要なことは，初期の頃にくらべ十分にサービスが得られるようになってネットワークの価値が急激に増加し，そこから得られる作業の質が非常にすぐれているというメリットがこのネットワークを通して得られ始めたということである。例をあげれば次のようになる。Chart14 にこれを示す。但しこれらの利点は，必らずしも民間会社やその他のユーザーが考えたものではない。ネットワークの利点として，ＡＲＰＡで考えたものである。

まず最初に，我々はコンピュータをネットワークに連結して見て始めて，その利点を得るようになったのであるが，それはサーバーの良いサービスが与えられるようになったということである。サーバー（ネットワーク上でサービスを提供するコンピュータ）の全体的な質の向上としてはドキュメントが整備され，ユーザーに対するレスポンスがよくなった事があげられる。またコンピュータを手許に持たぬユーザは，他のどんなコンピュータをも選べるのである。元来は，民間会社や大学には夫々コンピュータセンターというのがあり，サービスを供給していたのであるが，そのセンターがサービスが悪かったり，あるいは大学などでは職員が学生であったりということで，良い設備を選ぶ能力が無かったのである。また，そのサービスをどのようにしたら良くなるのかを理解している者も殆んどいなかった。ネットワークの場合，ネットワークを使うユーザは，どのコンピュータセンターを使わなければならないということはないわけである。従ってコンピュータセンターの管理者は，ユーザーに対するサービスを望ましいものにするのにより努力せねばならなくなった。

第２番目の特徴はソフトウェア，記憶（Storage）などの互換性を増加させることができるという点である。以前は２台のＰＤＰ－１０があり，ソフトウェアシステムをその一台のＰＤＰ－１０からもう一つのＰＤＰ－１０に移すということはできなかった。というのは詳細が正確にとらえられていなかったからである。一台の記憶量は小さいものであり，もう一台はコントロールキャラクターがモディファイされている等ということがあったからだ。しかしネットワークの場合は，少くとも同種のマシンでは，より標準化されたインストレーションの傾向を示すようになり，結果として，一台のＰＤＰ－１０にかけたプログラムをもう一台のＰＤＰ－１０に移すことができるようになった。私がカリフォルニアのコンピューターを使って，プログラミングを行ない，それを私自身のマシンに何時でも移すことができるわけである。

第３点は，システム開発のためのファシリティを向上させることができるということである。どの装置にもそれぞれのプログラム・エディターやアセンブラ等をもっているが，ネットワークが使用できれば，例えばＳＲＩのすぐれたエディターシステムや，ＵＣＳＢのファイルシステム等が利用できる。コンピュテーションの機能だけでなく，このような副次的ファシリティーを独自のシス

テムに加えて，研究開発の機能向上のために使うことができるのである。

新しいシステムのリアルタイムの配布についても，テープを郵便で送るよりも，電子を利用して送った方が良い。早く且つ誤りが少ない。もし何かの変更がおこった場合でも，電子を使って２～３秒で他の誰とも連絡できるわけである。例えば共通なプログラムの変更は，ネットを通して２～３時間ですべてのノードに送ることができる。

私は他の人が作ったディバッグ中のプログラムを使用することさえあるが，研究段階では新しいコンセプトを出来るだけ早く取り入れる事が重要であるから，正式リリースの１～３年も前にそれが利用できる利点は大きい。

私の考えでは，ネットワークがあるということは，コンピュータセンターがより良いサービスを行なうのにも有益であったと思う。また，コンピュータセンターにとっても，このネットワークは，よいセンターをつくる実用的な道具であると考える。

最後の特徴として，あげておきたいのは，システム担当職員を省力できるということである。一般的に言って，ネットワークの回りにある各地点はほぼ同じようなシステムを運用していて，我々はそれらを１つづつサポートする必要はないのである。例えば１０ケ所の各システムプログラマーは，言わばシステムプログラマー１０人のチームを組んでいるようなものであり，お互いにデータやコメントの相互交換を行なうことができるのである。ある１人のプログラマーが有益な変更点をみつけ他の９人がソフトウェア開発チームとして協力する。これは非常に有効である。我々の予想では，ＴＥＮＥＸを運用するＰＤＰ－１０を所有している各地では少くとも１人のシステムプログラマーの省力化がはかれるであろう。さらに，システム自体に問題が生じた時は，１ケ所で解決するより，他の９ケ所と共に助けあってお互いに種々の問題解決にのぞむことができると共に将来の開発にものぞめるわけである。

ネットワーク統計

次にネットワークのサマリーをChart 15 に示す。この表は 1971 年 9 月から 1973 年 1 月迄のネットワーク概要を統計的に表わしたものである。最初の項は，ネットワーク内で回線が取得できなかった平均時間，すなわち回線の総事故率（ total outage ）を全時間で割ったものを表わしている。平均して１～1.５%位である。一般的には非常に良いが，最悪の場合８～１０%になったことがあるが，こんな状態が起こったのは，種々の問題を解決しようとして手間どった最初の１～２ケ月ぐらいであった。それ以後は良くなり１%以下ないし１～２%であった。勿論，この故障率といっているのは一時的なものではなく，１５分～３０分以上の故障を示しているのである。

次の項にはネットワーク中のNode の事故率を示してある。数字が大きいところはネットワークを修正していた場合であり，ネットワークの拡大時にはソフトウェアに於いてもまだまだ詳細なところまで修正する必要があったからである。しかし７２年の夏頃までには，事故率は１～２%にまで下がっている。ここで強調したいのは，ここにあげた数字は全ての故障に対する数字であるということであり，操作中のネットワークを考えた場合，ネットワークの操作時間は最初は殆んど１日８時間にしていた。あとの３分の２の時間はとても操作することができなかった。その頃は，ソフトウェアの障害が起こり修理するのに１～３日もかかるというようなこともあった。７２年６月から１日２４時間のフル・オペレーションに変えたわけである。当分はあまりかんばしくない状況が続いたが次第によくなってきた。

次はハードウェア，ソフトウェア両方の故障率を示したものである。ＭＴＢＦは３時間とか７時間，ＭＴＴＲは２分とか３分である。

我々は，２４時間のフルオペレーショングを行ない始めてからハードならびにソフトウェアの統計も出し始めた。

さて右側の項はHost traffic の平均を表わしたものである。1971 年の初期は，Node が１５のネットワークでプロトコルを設けてテストしたりシステム開発を行なっていた。そして同年８月か９月頃から１日10,000 パケットから51,000 パケットを使用し始め，１０月には95,000，11月には116,000のパケットに増えた。７３年１月には１日約150万パケットとなった。それ以来，９月には310万，そして今月（７４年１月）では350万～400万パケットになっている。Fig３にこの伸びを示す。

リソースが増えればこれらはもっと増加するであろう。

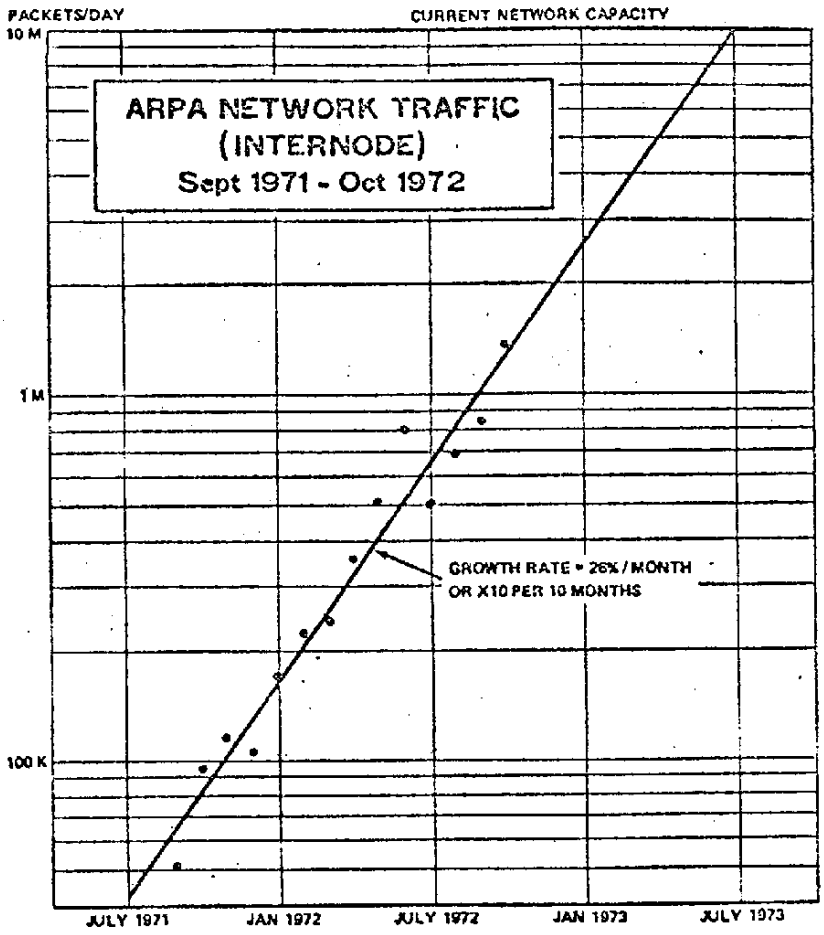

PACKETS/DAY
CURRENT NETWORK CAPACITY
10 M
ARPA NETWORK TRAFFIC
(INTERNODE)
Sept 1971 - Oct 1972
1 M
GROWTH RATE = 26% / MONTH
OR X10 PER 10 MONTHS
100 K
JULY 1971
JAN 1972
JULY 1972
JAN 1973
JULY 1973

Fig 3

ネットワークの信頼性

ネットワークの信頼性(Reliability)は元来2つのノード間の複数パスの数を基礎にして考えられたものである。我々にとって一番大切なものは，ネットワークによる処理の可能性である。

Chart16 はトポロジー分析によって計算したもので前述のChart6の(d)の23 Node，28 Link 回線の場合の計算結果を示す。28回線のうちどこか1つのLink がブレークしただけでは，ネットワークには結合できないNode はない。仮に28のうち2つのLink が故障したとしたら，378もの調べなければならない組み合わせがあり，そのうち30がCutset でネットワークに結合しない。28のうち3つが故障したら，3000 強の試みが必要であり，その$\frac{1}{4}$がCutset でネットワークにつながらない原因になる。4つの場合，20000 にもなり，約半分がつながらないし，5つの場合は殆んどつながらない。7つ以上になると，全くつながらないということが保証されていることになる。

ＴＩＰ

すでに述べたように，自分自身のコンピューターをもたなくともネットワークを使用することが出出来るが，我々がそのために開発した機器をＴＩＰと呼んでいる。Fig 4 にこれを示す。

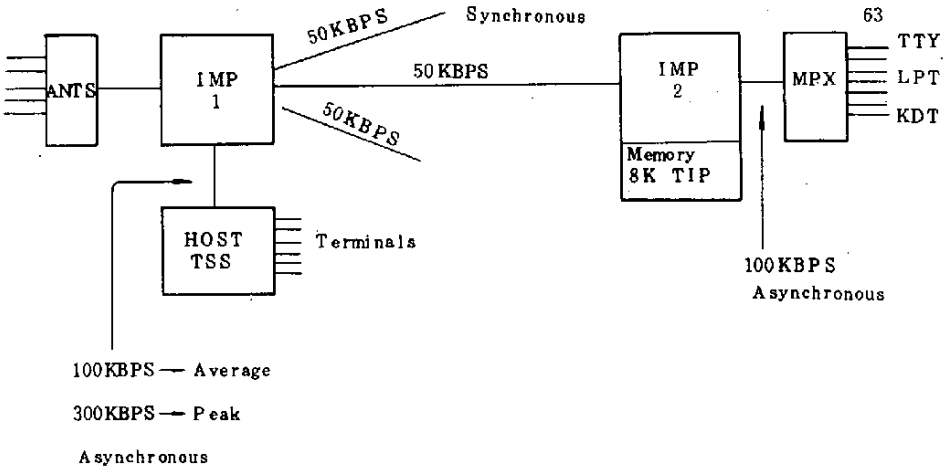


Fig 4

ＩＭＰ１とＩＭＰ２とは50kb／s でSynchronous につながっており，ＩＭＰ１とHOSTとは平均100kb／s で直接つながっている。しかし実際にはピーク時には300kb／s になる場合がある。これはAsynchronous である。

Host 側が大きなTime Sharing Systemであることもあり，これが一般的な場合である。そしてそのＴＳＳのターミナルがネットワークを使うことが可能であるが，HOSTがないところではネットワークを利用する際ＴＩＰを使うようにした。しかも直接ターミナルに結ぶことができるような複数ユニットを開発した。例えばFig 4 のＩＭＰ２からマルティプレクサーを経てキーボード，ディスプレイターミナル，テレタイプ，あるいはラインプリンター等６３ユニット迄つなげることが出来る。ＩＭＰ２はこれらのターミナルからはＨＯＳＴとみなされる。

マルチプレクサとＩＭＰ２の間はパケットで転送され100kb／sのAsynchronous なHOSTコネクションである。ネットワーク・アクセスのためにいくつかのシステムを各自が開発しており，例えばイリノイ大学では，ARPA NETWORK TERMINAL SYSTEM(ANTS)を開発し，これには３２のターミナルがつけられる。他のグループも小型コンピューターをあたかも大型ＴＳＳコンピューターのように使っている。しかも費用が安く，普通なら５０万～500万ドルかかるところを２万～５万ドルでやっている。

ＢＢＮの場合，出来るだけ簡単にして，記憶装置をＩＭＰと共有したわけである。そしてHOSTコネクションをとるよりＩＭＰの記憶容量を増やし，ハードウェアの代わりにソフトウェアでコネクトしたのである。記憶内容をＴＩＰ側のメモリーからＩＭＰ側のメモリーにムーブしてトランスミットする。ＢＢＮがとったような方法がＴＩＰと呼ばれるものである。扱うターミナル数は６３である。このＴＩＰ用に設けられたマルチプレクサーはプログラムが可能であり，従ってどんな種類のターミナルが連結されているかをしらせることができる。速度はインプット，アウトプットとも０～19.2kb／s　の間であればいかなるSynchronousスピードのターミナルでも接続可能である。Asynchronous の場合はインプットで０～2.4 kb／s，アウトプットで０～19.2kb／sである。

スピードとしては75，110，134.5，150，300，600，1200，1800，2400，4800，9600，19200 bps 等がある。

それではこのＴＩＰを使った際の会話を例にとってみよう。Chart 17 に例を示す。まずテレタイプでＥを打ったとする，するとＴＩＰプログラムが＂Ｅ＂を承認する。このＥは150c／s のＡＳＣＩＩターミナルの場合で，Ｅを入力されたことによりＴＩＰプログラムはその端末の種類がわかりそれに合うコードでHELLOと打ち返してくる。これが134.5 c／s のＩＢＭターミナルならばＥではなく＂Ｊ＂を入力するのだが，＂Ｊ＂を入力するとＩＢＭターミナルとわかりEBCDEC ＡＳＣＩＩでＨＥＬＬＯと打ち返してくる。

さてこのダイアローグの続きで，ユーザーは２３番のHOST えのコネクトをＴＩＰに依頼する。そこでユーザはＴＩＰにＬ 23↙とインプットする。（↙はＣＲＬＦ）ＨＯＳＴ23に対して結合がうまくゆくとTIPはTR OPENを打ち返し，これによって相手の タイムシェアリングシステムと交信できる状態となり，先方からネットワークを通してメッセージをうち返してくる。この例はARC TENEXの場合である。なおネットワーク上のHOSTはすべて番号がうってある。

また Close（閉鎖）の場合は Close の頭文字をとってＣ↙をインプットすれば，このHOST（Ｌ23）とのコネクトがクローズする。実際にはターミナルをOFF にすれば，自動的にＣの処理が行なわれる。

ＴＩＰユーザはひとたびHOST23 にコネクトされてしまうとあとは直接このHOST と会話をすればよく，もうネットワークにかかわる必要はないわけである。そこでユーザはＬＯＧ↙で log in を行ない，名前，パスワード，アカウント番号等の問いに答える。JONES，SECRET，1000 などである。ＨＯＳＴ２３を呼び出すまでは，このやりとりは必要ない。なお，使用料はこのアカウント番号にもとづいて請求される。そこでシステムは job 番号は６番，ＴＴＹ番号２番およびその時の日付と時間を出力してくる。これでシステムは準備を完了し，ユーザもシステムえの log in を終了し，システムはユーザが操作要求を行なうのを待っている状態となる。以上がネットワークが用いられる一般的な方法である。ユーザは何等かのターミナルの前にすわりプログラムないしコマンド言語でもってインタアクティブに操作を行なおうとする。この様なシステムでユー

ザが気にするのは記号Gを打ち込みそのエコー（ＥＴ）が直ちに帰ってくるか，または長時間かかるかどうかである。さてこの例ではARC TENEXに<WORK EASY>という名のファイルをGETしてBRANCH 3.（3は第3パラグラフという様な意味）をプリントせよと指令している。これが終了すると，次は log out を行なってジョブを終了する。その下はユーザが１０分３１秒に log in し0.０２秒使ったことを示す。経過時間は8時１２分から8時２２分の１０分間である。最後にシステムはクローズしている。

ＴＩＰやＡＮＴＳはネットワークの利用を著しく向上させた。これを我々が最初に採用したのは1971年の中頃であるが，それ以前は大型コンピュータのみが結合されていた。各地のシステムプログラマ達は他のコンピュータと相互コネクトをすることに非常に興味を持ったが，しかし，ユーザ達は他のマシンを使わねばならぬという理由はなかったし，自分自身の施設を使うということに慣れていた。またすぐれたリソースをもっているところは，どこか他のリソースを使おうとはしなかった。従って，ネットワークにＴＩＰやＡＮＴＳが現われるまでは信頼性のあるシステムは一つもなかったし，またドキュメントの整備やProtocolを作る意図も全くなかったわけである。

ところが，突然我々が数多くのユーザーに，どこかにあるコンピューターへの接点を設けるようにしたのである。しかしユーザー達はそれらを使うことが出来なかった。なぜなら充分なドキュメンテーションもなく log in を行なったが，間違いばかりで一つも正しい結果を得ることはできずじまいであった。そのため電話のトラフィックがその月は10,000～50,000回に，翌月は更に50,000～90,000回にも登ったのである。しかしながら，システムを使いたいというユーザのニーズは充分あり突如としてシステムもよくなり，急にtraffic も増え始めたのである。従って今後もさらにターミナルをHOSTやＩＭＰとはセパレートしたシステムとして使用していくユーザが多くなるであろう。そして，ターミナルを使うシステムでさらにネットワークを向上させることになる。現在でこのターミナルを扱うシステムを用いればネットワークが非常に便利なものであるということを殆んどのユーザが理解している。そしてこれによってＨＯＳＴ の利用率が３０%から８０%に向上している例もある。また大型コンピューター数台を自分自身で所有しているユーザー達はそれらコンピューターを一ケ所にプールしてしまうという方法をよく考えている。そしてそれらをこのネットワークに結びＡＮＴＳやＴＩＰによりすべてのコンピューターを利用することができる。効率も向上し，そしてオペレーションやメインテナンスも一箇所に集中できる。管理も便利になるし，建物もエアコンディション付きのものが一つでいいわけである。そしてプログラムは離れたところにあるそれぞれの端末から自由に操作することができる。この案については，現在，３～４ケ所の候補地を考えている。しかし，自分自身のコンピューターを動かしてもかまわないというところとそうでないところの意見が同数であるので目下のところ難しいように思える。

ＩＭＰとＨＯＳＴの台数

元来一つのＨＯＳＴはＩＭＰと一体になり，ＩＭＰはＨＯＳＴのすぐ傍に置かれるものである。しかし我々がその決定を行なった直後に，我々のプランをユーザの全員に伝えたところが，あるユーザは，既に５台のマシンをもっているので，とか，コンピュータを４台入れてしまって１台はあるビルに，２台目は隣のビル，３台目は市内に，４台目は３マイル離れたところにあってどうしようもないということであった。その後，我々は４台のＨＯＳＴまでＩＭＰに連結し作動する方法を最低の段階として考えたわけである。即ちその４台のＨＯＳＴのうち１台が離れていてもよいということを意味しているわけで，離れているHOSTとIMPとの距離が2000フィート以内であれば 連結することを可能にしたわけである。このためにＩＭＰへの新しいインターフェースを設けどんな距離に離れていようとエラーコントロールがついた同期回線のインターフェースを可能にしたのである。またそれにより1000 マイル離れているＨＯＳＴも直接連結できる結果となった。現在，可能なことは，単一ＩＭＰに４台のＨＯＳＴを連結させ，そのＩＭＰに７つのModemと７つのラインをつけることができるということである。しかしパッケージィングの制約から，HOSTとラインの合計が７以下でなければならない。例えばHOST4台に３つのライン，３台のＨＯＳＴと４つのライン，あるいは２台のＨＯＳＴと５つのラインといった如くである。またＨＯＳＴの距離は，第１台目は３０フィート以下，２台目は３０フィートから 2000 フィートまで，３台目は 2000 フィート以上であってもよい。

ＩＭＰは基本的に言ってミニコンピュータである。その役目はプログラムやメモリによって，ハードウェアのインターフェースのような働きをさせるわけである。

ネットワークの効率

まずネットワークを構成する主要な要素は何であるかを説明してみる。私がかってNetwork Analysis CorporationのH・FRANK氏や，ＵＣＬＡのＬ・Kleinrock 氏と共同して数ケ月にわたって一つの特別難しい論文を書いたわけであるが，実際，この論文[*1]の最初の部分で我々のフィロソフィ（philosophy）が非常に違っており，相互の意見調整のプロセスがきわめて難しかったと述べている。しかしその反面，ネットワークの振舞（behavior）に関してはよく似た結論が得られた。

フランク氏はNetwork Analysis Corpに籍をおいて，特にネットワークの信頼性や構成がネットワークのサービスにとってどのように必要であるか，またその場合，費用がどれ位かかるかといった問題を研究しているわけである。例えば，このネットワークではどのような信頼性が得られるか，その信頼性に対して経済的にどのように対処するか，回線をどこに配置するか，どのようなネットワークであれば費用的にも信頼性の面でも最良のものであるかなどを研究し提案している。基本的にはARPANETをさらに効率的にしようとすることに関する研究である。しかしながらNetwork Analysis Corpはこのネットワークがどのように動作するかとか，スループットはどの程度かというようなことには全く関与しなかった。一方ＵＣＬＡのKleinrock氏はネットワークのモデルの研究を行なっており，例えば２つのノード間の遅延時間の関係についてはどのようなことが言えるか，種々のネットワークに対しどのようにして遅延を計算するか，このネットワークは他のネットワークより遅延が多いか少ないか，その場合，ある回線を他の回線と代えるべきかどうかということなどについて研究している。論文264ページには同氏がまず最初にＡＲＰＡの標準計算理論を述べている。これはよく知られている単一サーバーのキューイングモデル（queueing model）である。サーバー１つとトラフィックが与えられるそのモデルではある分布に基いてメッセージが流され，メッセージがシステムに存在する時間を計算する。それから，また同氏は，ネットワークを形成するのに多数の単一サーバーキューをどのように合成するか，また同種のネットワークについて全遅延をどのように見積もるかについての理論的なアプローチを研究したのである。また彼の研究したもので「Independent Assumption」と呼ばれるものがあるが，この中では，仮にリンクの各々を別個の伝送系として扱ったとしても，全体のシステムの作動はシミュレーションで得られるものと非常によく一致しているということを述べている。265ページの(7)の式[*2]はかなり複雑であるが効果的である。

＊1 Computer Communication Network Design Experience with Theory and Practice 1972 SJCC

＊2 $T = K + \sum_i \frac{\lambda_i}{\gamma}\left(\frac{1}{\mu' C_i} + \frac{\lambda_i/\mu C_i}{\mu C_i - \lambda_i} + P_i + K\right)$

この式はNetwork Analysis Corpが遅延を計算するのに使用した。K termはＩＭＰでの処理時間であり，またｉは回線上のトラフィックを表わし又γはトータルフロを表わす。その他のtermとしてPacket 長を表わしたり，伝播遅れを表わしたりするものがある。これはかなり全体的なモデルであり簡単に理解できるものと思う。

さて，以上のような問題について共に研究した結果，殆んど同時に達した結論はＡＲＰＡのようなNetworkのモデルの遅延はFig 6に示すようなカーブとなるということであった。ρは負荷－－－はCapacity point である。

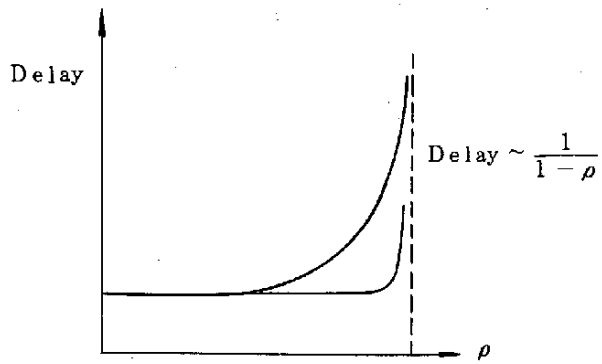


Fig 6

しかし実際には適当なフローコントロールを行うことでかなり正常な状態にすることが可能になった。即ち，ネットワークへ過度のトラフィックを送ることを止めれば，少ない能力でも短い遅延を保持することができたのである。

我々が成し遂げた方法はネットワーク内のバッファー記憶を減らすことであり，一方Network Analysis Corp やＵＣＬＡが得た結果はネットワークが無制限のバッファー記憶量をもっていることを仮定していたわけである。現実にはネットワークでのバッファーの数をかなり少なくしてしまった。ところが実際には，かなりの負荷に対してもネットワークの遅延はさして大きくならなかった。ネットワークに許容以上のトラフィックを入れて，ネットワーク内で大きなキューイング遅延を行なうよりもネットワーク外のバッファーに保持した方がよいのである。これによって多くの遅延構成の計算が可能になり，複雑な方程式を計算するより，その特徴づけも簡単になった。

我々は構成を特徴づける非常に効果的な方法を，いくつかの簡単なフローコントロール技術で得たのである。この件については後程説明することにする。論文Ｐ 266をみれば，ＡＲＰＡ機能としての遅延特性を４つの違ったケースについてそのカーブを示していることがわかる。ＣとＤのケースは実際的な場合であり，全ＡＣＫ信号とＲＦＮＭ等のオーバヘッド・トラフィックを考慮した場合である。Ｃのカーブは1000 ビットパケットの場合であり，Ｄのカーブは500ビットパケットの

場合である。これは１０のNode をもつ全ネットワークを通じての全パケットである。AとBのカーブはすべて右寄りになっているが，実際にシミュレーションを行なって再計算したものである。この場合，実際のデータと理論的に比較する目的からオーバーヘッドビットは考慮されていない。これら両方のシステム能力は約400kb／秒である。A，Bのカーブの近くに，２つの小さな⊗印があるが，これはかなり複雑なシミュレーションにより得られた結果に相当し，私がＢＢＮでやった実験の結果である。これによると，ネットワークに全く余裕がない時，フローコントロール処理でこれを救うことが出来る。即ちネットワークに余裕がない時，キューを短かく切ってしまうのである。私の考えではキューの長さは４～８ぐらいで，これはＩＭＰ内の内部再伝送に対してとる政策によって違ってくるが，しかしどんな場合でも８以上になることはない。従って，ネットワークを通しての全体の遅延は普通の場合１～２回，最悪の場合でも８回以上にはならない。これは今まで全体のテストを行なって出て来た面白い結果である。我々はネットワークがスタートした時にはこれを十分理解出来なかった。重要なことはネットワーク動作のモデルを複雑にする必要はなかったということであり，かつ内部でバッファーリングを行なったり，そのバッファを大きくすることで多くの利点が得られたのである。我々がつきとめた統計では，キューラインにおくバッファーの理想的な数はそのキューラインを100％生かせるような数であればよいということであった。その数は簡単に計算できると思う。非常に長いラインを50kb／sの通信チャンネルを使って1000 ビットのパケットを送るとすると，約１０ないし１２パケットに相当するバッファーを準備する必要があるわけである。勿論ラインを十分生かした上でのことである。ラウンドトリップで，１０マイルの距離を持つ回線に，パケットを送りだすとして，２番目のパケットを送り始める時には，すでに１番目のパケットに対するＡＣＫが戻ってきており，そして２番目のパケットを送るということになる。従ってキューラインで必要になるバッファー数は最低２個であり，非常に長い回線にはキューラインに１２，そして時には再伝送するために余分のバッファーが必要となる。恐らく短かい回線には４～５，長い回線には15~16 ぐらい必要であろう。現在のＡＲＰＡ　Network は非常に少ないバッファーで作動している。最高で８ぐらい使用している。今までこの数でネットワーク・コンジェスションを非常に効果的にコントロールすることができたのである。

ネットワーク・デザイン

ここで，我々が行なったネットワークのデザインの問題のいくつかをあげてみることにする。

① ＩＭＰのデザイン── これはスウィッチング・ノードの設計である。

② 伝送系のデザイン── ＡＲＰＡではＡＴ＆Ｔが使われたが，ＡＴ＆ＴはＡＲＰＡに対し特に高い信頼性を保証してくれた。

③ トポロジカルデザイン── 信頼性，スループットおよびコスト等に対してのトポロジカル・デザイン

④ 一般的なモデリング── これは時にはトポロジカル・デザインの範ちゅうに入る場合がある。

⑤ プロトコル

⑥ ユーザーアスペクト（user Aspect）

ヒューマンエンジニアリング

ソフトウェアシステム

ネットワークの分析と理論的なネットワークの振舞いの解析はこのトポロジーのデザインと共に行なわれるものであり，理論的なモデルについての知識がなくても，プラグマティックな面からもトポロジーのデザインを行なうことができる。モデルを使うよりも，そのデザインはより良く行なえるのである。そしてネットワークコミュニケーションの費用は３０％より５０％，７０％というように，コミュニケーションのラインのロケーションによって決まってくるわけである。この構成要素の機能を最大限に発揮させ，設計を最良にしようとする為にはリピーターや接続器の位置を，選ばなければならない。しかし必らずしもそのロケーションを選択しなければならないということではなく，どうしても費用を安くあげようとすれば，選ぶ必要があるということである。我々の場合もそうであったし，恐らく民間企業であれば選ぶ必要があるのではないかと思う。殆んどの場合は選択しているようである。

さて，その他の必要条件の種類を述べることにするが，一般にターミナルシステムすなわち，ＴＩＰやＡＮＴＳといったいくつかのシステムがあり，これによって自分自身のコンピューターを持つ必要はないということは以前にも述べた。このようなシステムをユーザが十分利用するにはネットワークの信頼性が非常に重要である。そして電話のネットワークやデータ伝送以上に，サービスが複雑である。即ち，ユーザからはインタラクティブに端末を操作したり，あるいは表示画面への高速なディスプレイをする等の要求があるからである。ターミナルシステムに対するプロトコルは人間の操作性を充分考慮に入れねばならない。そしてプロトコルとしては，ユーザレベルのものとシステムレベルのものが必要になり，ユーザレベルではどのようなプロトコルをどのようにして使うかという人間工学的（Human Engineering）な面にも関連する。次にソフトウェアについて言えば，これは人間の行なう技術であるが最も重要なものの一つであろう。ネットワーク上では，

ハードウェア的ファシリティとともにこのソフトウェアがシステムの決定要因となる。ソフトウェアの技術的な問題は他のものよりも難しいものであり，ハードウェア施設に対する考慮よりも複雑なものである。我々ARPA がこの問題の開発につき込む費用は恐らく全体の $\frac{2}{10}$ぐらいでありかなり多い。

ネットワークを利用すれば米国内のどの地域でも仕事ができる。例えば私の場合電話で普通1日30~50 のメッセージや質問があって，中には一つが約15分もかかる場合もあるが，これらの殆んどがメッセージによりネットワークのラインで返答できる種のものである。もう一つの効果としては同一の電話を受けることがあるが，これらはむしろ電話によるメッセージの伝達よりも，ファイルにして渡した方が良い。即ちそのファイルなりファイルのコピーをネットワークを通して送るだけで良い。そうすればユーザーも目で見られるわけである。あるいは提案を郵便で送る代りに私のDirectoryに入れてもらえれば，私が読んで何処にいようとおもむくことができるわけである。ハワイへでもカリフォルニアでもすぐ行けるのである。このような考えは過去の慣習を破るものであり，我々のビジネスのあり方という問題にも大きな影響をあたえる。私が思うにはこのような手段は非常に価値あるもので，時間的な意味においても意義深いものである。

ネットワークを経済的に使えるようにとの努力は我々の目的達成の重要な問題である。アメリカからハワイまでの回線は50kb の回線，単一チャンネル，Voice-rate装置であり，単一チャンネルは 64 kb で traffic を送る carrier 用であった。その 64 kb チャンネルを分割して 50 kb にしたもので，その費用は50kb 回線と同一の価格であった。現在，アメリカにおいて 50 kb の通信衛星サービスの費用は1年間15万ドルぐらいかかるわけである。日本円にして3～4百万円といったところであろう。そして海外へ流せば同じ装置で関税も含めるとその5倍ぐらい必要になってくる。従って我々がカリフォルニアからハワイまで流すのに15万ドル払っている同じ回線は，全く同じ装置をつけてヨーロッパへ送る場合はその4倍，日本へ送る場合5倍必要になってくる。従って我々は，少しちゅうちょしたのである。実際のところ trafficの伝送も思ったよりもかなり少ないものであった。しかし英国は，この回線に結びつけたのである。勿論，上記のレートで損はしないのみならず，英国内でデーターコミュニケーションを向上することができたし，またこの回線に対しての需要があったからである。事実，殆んどのケースをみてみると，現在，ワイド・バンドの国際的なトラフィックに対する要求は大いにある。したがって英国の場合のように我々ももっと多くの努力を払うのに良い機会であると思うし，ワイドバンドをやってみようという国は，実際にコミュニケーションのサービスの費用に対し，関税も下げるようにしなければならないと思う次第である。

TELNETプロトコル

プロトコルの開発というのは，最もわかりにくい，難解な問題である。プロトコルにはFig 7に示すいくつかの異ったレベルがあるが，それがＨＯＳＴシステムのどこにあるかという事を図で示す。

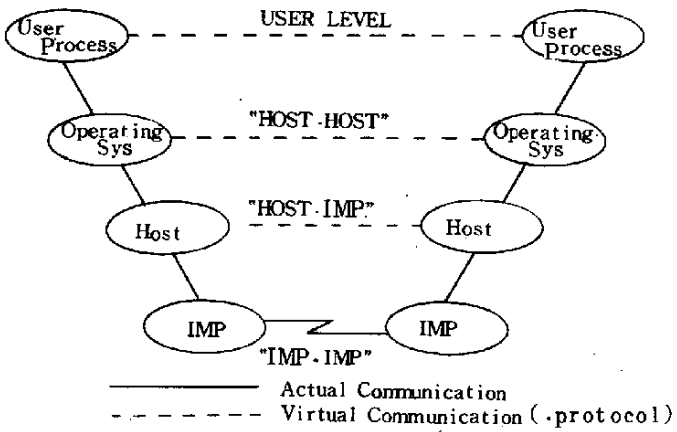


Fig 7

Chart 18 はＴＩＰ（Terminal Interface Processor）がどのようになっているかを示している。大変簡略化された図であるが，この端末がホストコンピュータと連結されるためには，プロトコルが必要になる。これは，プロトコルを通して他のシステムと交信するターミナルシステムで，Chart 19 に示すようにネットワーク・コントロール・プログラム（以下ＮＣＰと略す）が遠くのＨＯＳＴ・コンピュータの他のＮＣＰと交信している。ＮＣＰ同志で互いに通信し合うわけである。

プロトコルもＮＣＰの一部であると考えられるが，ＮＣＰには，端末と関連する働きは一切ない。ＮＣＰの働きはあるＨＯＳＴから他のＨＯＳＴへパケットの受け渡しを行う事であり，その位置はChart 19 に見られる通りであり，ＩＭＰを通してその両端のコントロールを行っている。

その連結は一続きの連続的なものではなく途中でスタートしたり，止めたりすることができるものである。

このプロトコルは，テレタイプ等で端末交信を行うので，TELNETプロトコルと呼ばれている。

このTELNETは端末にデータをバッファに入れさせたり，それを送り出させたりする。また，エコーの操作や，その他，端末が他の側と交信する時に必要となるすべて，つまりその端末が交信したがっているのは何処かといったことに関する記録や，その端末のための変換を行ったりする。

ＴＥＬＮＥＴ プロトコルの中には，ターミナル・コントロールとは区別されていて端末の使用をプロトコルに命じてやめさせる部分がある。これは，ユーザーのターミナルからＴＥＬＮＥＴへの指令のために使われる。この図で示すようにＵＳＩＮＧ ＨＯＳＴの側のＮＣＰがＳＥＲＶＩＮＧ ＨＯＳＴ

のＮＣＰと交信するわけであるが，これは端末がデータをそこへ向けて送信するユーザープロセスの一過程であり，このユーザープロセスは，その入力データに基づく処理結果をターミナル・コントロールプログラムと，ＮＣＰを通してそのユーザーの端末へ送り返す。このサーヴィングホストは，先方の端末とうまくコミュニケーションを行う為に，その端末の性格についての知識を持っていなければならない。例えば，このユーザプロセスが非常に長いファイルをもたらすものであったとする。そして，それはディスプレー・ターミナル，あるいはキーボードディスプレーの上に表示されるとする。ユーザプロセスは，十分なデータをスクリーンに送り返そうとし，ユーザは，更にキーをタイプして，もっとスクリーン上にデータがほしいという意志表示をすると，プロセスは更にデータを送って来る訳である。と云うのは，すべてを見るのに十分な記憶装置が，端末側にあるという風に考えるべきではないからである。非常に多くのテキストファイルが大容量記憶装置に入っており，それをレトリーブ（ retrieve）するかもしれない。ユーザは一時にはその一部だけを見るわけで，そういうプロセスのためのターミナル・コントロールも行なわれる。

ＨＯＳＴ　ＩＭＰインターフェース

ホスト・システムとＩＭＰのつながりは，二つの部分になっており，その一つはＩＭＰ側の標準的インターフェイスである。特殊なインターフェイス，即ちＨＯＳＴオリエントな部分はそれぞれのホストの側に組み入れられている。Chart 20 にこれを示す。

これは，我々の最初のデザインで，もしこの様な機能をホストの近くに組み入れたい場合には，これは現在も尚，望ましいものであると考えている。もし何らかの理由で，これ等の機能をホストから遠く離したい場合の望ましいインターフェイスは，パケットインターフェイス，つまりパケットにオリエンテドなインターフェイスであると思う。ＨＯＳＴは，ＩＭＰにメッセージを送り，ＩＭＰによってパケットが送出される。ＨＯＳＴとＩＭＰの間のインターフェイスは，非同期のハンドシェイキングオペレーション（Hand Shaking Operation)がなされるインターフェイスである。

Chart 20 の図はＨＯＳＴとＩＭＰの間の距離が３０フィートか，それ以下の場合である。我々がデザインした最初のものは，Fig 8 に示すように１２ラインであった。

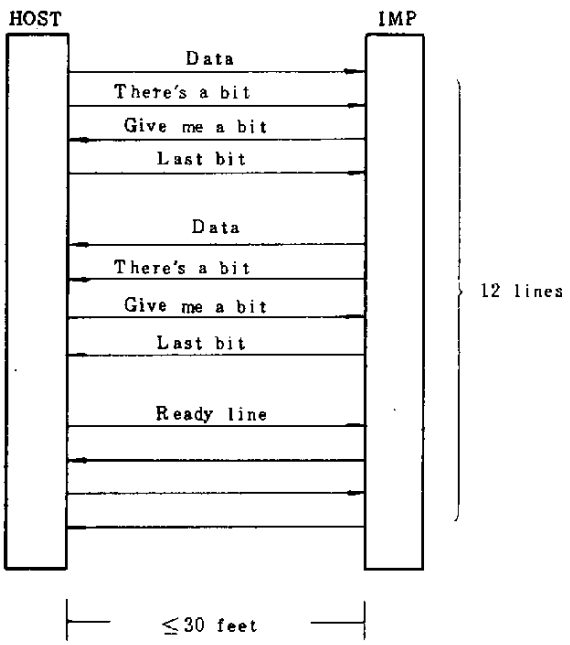


Fig 8

データのシグナルがホスト側からでる。そしてビットが届くと，「There is a bit」というシグナルが出る。今度は反対方向から「Give me a bit 」というシグナルが出て最後には「Last bit」というシグナルが来る。この様にしてデータの伝送が行なわれる。どちらか一方の側から，ビットの準備が出来たと云うシグナルが出される迄は，何も始まらない。ビットが流れる前に，一

方がビットを出し，他方がそれを受けるという同意が双方でなされなければならない。これは，全くの非同期のインターフェイスで，一方が「Give me a bit」と云い，このシグナルのレベルを高にしたまま待ち，他方が送るべきデータを手に入れると，「There is a bit」というラインを上げる。そしてデータが送られてくると，先程の「Give me a bit」というラインを下げて，それからもう一度それを上げて「Give me a bit」とする。片方は一度下げたラインを再び上げて「There is a bit」という様にくり返す。最後のビットが伝送される時，最後のビットを表わすシグナルが出され，それはメッセージが終った事を意味する。それをもっと詳しくしてみると次の様になる。

ビットがＨＯＳＴ側にあるとして，ＩＭＰ側が「Give me a bit」という信号を出す。すると「There is a bit」という信号がＨＯＳＴから出る訳であるが，時によっては，ＨＯＳＴに「Give me a bit」という信号を出した時，そのＨＯＳＴの所にビットが全然ないという事もある。しばらくして，ビットが手元に来ると，ＨＯＳＴはそのラインのレベルを上げる。そして，受け取る側即ちＩＭＰ側のハードウェアがビットが送信されたのを確認して，そのデータをレジスターにおさめ，「Give me a bit」のラインを下げる。そしてそのシグナルがＨＯＳＴにもどるとHOST側も「There is a bit」という信号を下げる。その信号が下がるのを見てＩＭＰ側は次のビットを受ける用意ができ次第再び「Give me a bit」というラインを上げる。「Give me a bit」というラインを上げるということは，いつビットが来てもいいという準備が出来た状態であるから，送る側のインターフェイスが，出すべき次のビットを手にするとすぐ，それを送る準備をし，もう一度「There is a bit」を上げる。そのうちに，「そちらが今受けとったのは，最後のビットです」即ち「Last bit」というシグナルが入るが，実際には，これは「There is a bit」のシグナルの直前に入る。この交信方法では，途中どの時点ででも止める事が出来る。その場合，両側が，交信再開迄，待つ様に準備されなければならない。遅延させる目的は，ＩＭＰが，記憶装置のバッファをもう一つ取って来るような場合である。しかしホストを５秒でも１０秒でも，何でも彼でも待たせるというのではない。例えば１０マイクロセコンド毎に出る事になっているビットをすべて100マイクロセコンド毎にするよう要求するというのではなく，時として，記憶装置或いは別の所からもう少しデータをもらわなければならないとか，それがどこにあるかを計算するとか…そのような，何かする事がある為に，その間５０マイクロセコンド位待ちたいということである。実際問題として，ネットワーク上のいくつかのオペレーティング・システムが非常にゆっくりで，ネットワークからデータを受け取るのに２０秒の遅延がしばしばあるということが起ってきた。しかし，我々は，オペレーティングシステムを早急に作り直す事は大変であると考え，まず待たせる事を可能にしたが，それは３０秒ということにした。もし，オペレーティングシステムが３０秒以内にネットワークからパケットを取り出したり，メッセージを受け取ったりしなければ，そのメッセージを捨てて，その旨シグナルを返す。しかし，それは本来の我々の意図とは違い，また，我々が提案している操作でも

ない。オペレーティングシステムというのはどんなものでも，十分な配慮さえなされていれば，一秒の何分の一かという短い間に，次に来るメッセージに答えることができる筈である。したがって現在不可能と思われているようなケースでもそれができないという理由は何もない訳である。こういった考え方を堅持し，それに基づき進めて行けば，時間の経過につれて，ネットワークに直ちに感応できるオペレーション・システムというものは次第に増加して来る筈である。

先程ＨＯＳＴとＩＭＰの間のラインが１２ラインという話をしたが，その中の４ラインがHOSTからＩＭＰに別の４ラインがその逆の伝送のためのものであった。残る４ラインは，インターフェイスの両面が正しく動作している事を確かめる役目を持っており，基本的には，一組のレディ・ライン(ready line)である。

そこでインターフェイスがいかに働くかという事であるが，まず，HOSTからネットワークへデータが出て行くケースを考えてみる。そこで開始の条件はまずHOSTのプロトコルが，データを１セット，サーキットポジションのメモリーにおき，それをインターフェイスを通して送るようにと指令する。そこでその分のメモリーを確保し，そこにデータをセットする。データとしては，３６ビットか１６ビットかいずれにしろ１語のデータであるとする。この語を，アウトプット・シフト・レジスタ（３６ビット）に入れる。すると，アウトプットコントロールロジックへシグナルが出され，「アウトプット・シフト・レジスターにデータを入れました，どうぞそれを伝送して下さい。」という。このハンドシェイキングプロシジュアを基礎にして，このコントロールロジックが，送信する用意ができているというラインをオンにするように指示し，それに対しＩＭＰ側のインプットコントロールロジックが，受信する用意ができている旨を伝える。伝送されたデータはＩＭＰ側のインプットシフトレジスターの中へ置かれる。このＩＭＰ側のシフトレジスターは，１６ビットであるが，これはメモリーの語のサイズによる。したがってデータは，インターフェイスを通って１ビットずつ流れ，１６ビット毎に，ＩＭＰ側のインプットコントロールロジックが，「語全体が受信されました。」といい，ＩＭＰのメモリーに移され記憶される。これらの経過はインターフェースの装置の種類によって若干違って来るが，インターフェイスを本当に便利なものにしようとすれば，どんな時点においてでも，その交信を，自由にスタートさせたり，止めたりできる能力が必要になってくる。それは，少くとも，我々が最初にそうありたいと望んだ考え方であった。そして我々がこのネットワークの為に作るようにと依頼したのは，かなり単純なインターフェイスであった。

Chart 20 で説明したラインは，本質的にはＤＣラインで，ここではシグナルが高電圧になり，ゼロがより低いレベルの電圧，ゼロマイナス５ボルトまで下がっていた。

その可動距離は，３０フィート迄であったが，これではすべての場合のインターフェイスとしては役割を果たせないことになるので，同じインターフェイスを2000 フィートのケースまで扱えるようにした。またそれだけの長距離を可能にする為にディファレンシャルドライバーを使った。こ

れがケースBのインターフェイスである。

ディファレンシャル・シグナル（Differential Signalling）はシグナルのレベルを上げたり，或いは下げたりするよりむしろ，一組のペアをとりそれを上げたり，又は下げたりという事で，たまたま一方が上っていて他方が下っている時，減法を行うとハイ・レベルが出る。そこで次に加法を行うと0になる。というわけで，ゼロでもハイレベルでもどちらでも出せる訳である。これは，もしノイズがあってもその影響が打ち消せるので大変便利である。そういったやり方によって，はるかに大きな距離を出せることになる。これはどんなコミュニケーション・エンジニアでも恐らく知っているであろうディファレンシャル・シグナルを行う為の標準的テクニックである。

次に我々が行ったのは，地下絶縁（Ground isolation）であったが，これは，二つの場所の間にしっかりとした地中連絡をとる良い方法であると思われた。二つの別々の建物などのかなり離れた距離にある二つの物を地下絶縁で他のパワーから隔離してつなぎ合わせることができた。そしてこれにはセパレートパワー（Seperate Power）の技術を使用し，両端から別々に電流を流し，実際問題として，長距離にわたって電力を流す事はなかった。

これらの方法が，かなりな距離をもつものにも大変効果的である事が分かった。しかし，これにも又問題があって2010 フィートの距離のホストの場合にはうまく行かなかったので，大体2000フィートというのが限度であると思われた。そこで我々は，コミュニケーションのテクニックが全く異っている超遠隔ホスト・インターフェイスと呼ばれるものをやってみた。その説明に入る前にまず一組のＩＭＰをどのようにつなぎ合わせかを話しておきたいと思う。

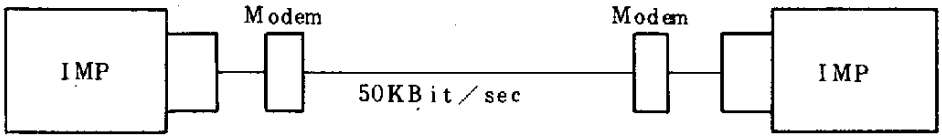


Fig 9

Fig9のように二つのＩＭＰが配置され，このＩＭＰは互いに，Modem インターフェイスを通して交信するが，このインターフェイスは，ＨＯＳＴインターフェイスとは異っている。各々のModem の間にあるのは50kbit／sec のラインである。何も伝送されていない時はただＳＹＮ character（ＡＳＣＩＩコード）がライン上を送られるだけである。他の手法としては何も起らない時には，１を送りつづけるとか，スタートを表わす為にそれをゼロまで下げるとかいったような，多くの異った対策がある。我々の場合は，ＳＹＮ，ＤＬＥ，ＳＴＸのコンビネーションを発見すれば，パケットがスタートし，Fig１０に示すようにHeaderおよびＴｅｘｔ（データ），そして

ＤＬＥ，ＥＴＸとつづくと仮定した。ＥＴＸ　character の後には３つの check character があり，これは，２４ビットのエラー・チェッキングで，その後にまた別のパケットが来て同様につづくわけである。

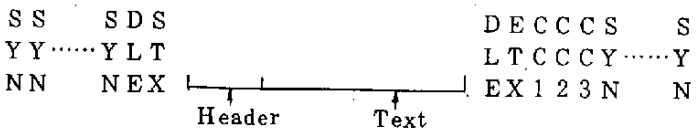


Fig 10

以上がネットを通じてライン上にパケットが現れる標準的行程である。つまり，パケットのスタートを知らせたり，その終りを知らせたり，強力なエラー・チェッキング（２４ビットまで）などがあり，この間にあるテキストは，他のすべてのものがハードウェアで翻訳されるのに反して，ソフトウェアで翻訳される。ＤＬＥ，ＳＴＸは，パケットが到着する前にハードウェアによって見分けられる。テキストは，ハードウェアで識別されて，ＩＭＰの中に送りこまれ，以後ソフトウェアで解読されるが３つのキャラクタのエラー・チェッキングはハードウェアで行なわれる。そして更に次のパケットが来るまで待つこととなる。ここで，もしテキストの途中で，別のＤＬＥ，ＳＴＸが来てしまったとしたらどういう事になるのかという質問が出るかもしれない。

この場合にはどうすれば良いかというと，我々が行うのは，途中でＤＬＥが起る度に，もう一つ入れるという事である。一つのＤＬＥ毎に，それを二つセットする。ＤＬＥが見つけ出される度にレシーバーは，次の character をしらべる。即ちＤＬＥが見つけ出されると，常に，最初のものは捨られることになっている。次に来る二番目のものが何であるかによって，もし，それが単にもう一つのＤＬＥであれば，それはそのまま通り，すべての事が普通に続けられる。もしそれがＥＴＸであれば，それがメッセージの最後であるという事が知れるわけである。というのは，もしそうでなかったら，次の characterは，ＤＬＥであった筈だからである。テキストの途中では，ＤＬＥ― ＤＬＥか或いはＤＬＥ－ＥＴＸのこの二つ以外のものはあり得ない。他のいかなるコンビネーションも起り得る論理的可能性は全然ないわけであるから，ＤＬＥが来れば，常にそれを捨てて，次に来る character に注意を集中すればよいがもし次に来たのが別のものであった場合，エラーを入れて，パケットの交信を避け，次のキャラクターを求めるわけである。

次にホストが遠くに離れている場合を説明したいと思うが，ＩＭＰとＨＯＳＴはある距離をおいて別個に設置されるので，ＨＯＳＴがデータを受けとったかどうかが，確実ではない。したがってＨＯＳＴとＩＭＰの間に受領承認を送り返す方法をとり入れる。これが＂Ｃ＂とよばれるインターフェイスでChart21 に示す。ＶＤＨ（Very Distant HOST）というものである。これは2000

フィート以上の距離の場合，非常な有益な方法であり，基本的に２つのチャンネルがあり，次のような機能で作用する。

チャンネルＡとＢがあり，パケットはまずチャンネルＡにプットされ，新旧いずれのパケットかを示すビットを０とする。次にチャンネルＡに送られたパケットに対してはそのビットは１となる。新らしいパケットを送る時はいつもそのビットを反転するわけである。受け手はまずＡをそしてＢまたＡをそしてＢをという順でパケットを受けることになっている。最初のパケットがＡに送られ受け手に渡る。それから次のパケットがＢに送られ受け手に行く。これもビット０かも知れない。チャンネルＡは正確に受領された旨のＡＣＫを受けとる。パケットＡがＡＣＫされたと分かれば，次のパケットは違うパケットであることを明示する（ビットを１とする）。同じようにＢはＢに送られてくるパケットのＡＣＫを受ける。従って，どのパケットが伝送されたかを確認することが出来る。

例えばＡを０で送り，続いてＢを同じく０で送る。しかしＡＣＫはこないとする。そこで再びＡを０で送る。しかしまたＡＣＫはこない。そして再びＢを０で送る。もしここまだＡＣＫがなければ再度Ａを０で送らなければならない。恐らくこれでＡＣＫがでる。そしてやっと次にＡを１で送ることになる。これは誠に悪い場合で普通は一度送れば良いであろう。そして受け手の方は０として３つのパケットが入って来たことを理解し，最初のパケットを正しいものとしてその他の２つのパケットを捨てて，次にくるパケットはＡの１であると思うのである。したがって，この場合ビット０というのは，同じパケットが再度送られたことを伝えるだけに用いられる。新しいパケットが１で入ってくるまでは古いものと思うのである。

以上がＶＤＨインターフェイスの作用のしくみである。このＶＤＨインターフェイス上での作業を簡単にする為にＤＬＥの２重挿入は止められており，その代わりカウントが入れられている。このインターフェイスは，ワードの単位で記憶するというより全パケットを単位として記憶することができる特殊のものである。普通のシフトレジスターであれば１語（３６ビット）ずつしか記憶出来ない。このインターフェイスの場合，ホスト側が全パケットを記憶してしまう。したがって一度パケットをライン上に送り出してしまえばあとはＨＯＳＴが，それをシンクロナスに操作し続ける。即ち一度に全パケットを送り出すことになる。勿論ＩＭＰの方は問題はない。何故ならＩＭＰが一つのパケットを投げだしても，事実シンクロナスにそのパケットを送り続けることが出来るからである。

ネットワーク内のバッファー

先程も述べたがネットワークにはあまり多くのバッファーを含まないようにすることが望ましい。Chart 22 に示すように例えば50kb／sの回線で約1000 マイル以下の距離であれば，このラインに必要なバッファーの数は100％ラインを使うものとして2個から１２個ぐらいの幅がある。一番多くのバッファーが必要となるところでは（図の一番上のカーブ）短かいパケットと長いパケットの割合が１：０，すなわち全て短かい小さなパケットの場合である。バッファーが少ないケースというのはその割合が０：１の場合で，即ち全て長いパケットばかりである。全てが1000bit の長さのパケットである。このような場合，実質的にバッファーの数は1回線につき2個だけが必要なのである。その他の場合は，短かいパケットと長いパケットが同数かあるいは2倍，また8倍の場合もある。そして9.6 kb／s の回線でもバッファーの数は大して変わらないし，通信衛星回線の長さである45,000 マイルまで可能である。しかし通信衛星レベルでの必要バッファー数は１０個以上多く必要になってくる。 230. 4kb／s の回線速度では， break point は，500 マイルぐらいである。通信衛星の場合は必要バッファー数がかなり増える。

次に1400kb／s（1.344 ～1.544 Mb／s）では break pointは約100マイルぐらいである。通信衛星の場合はこの図ではスケールをオーバーしてしまっている。

ＩＭＰのコアマップ

ＩＭＰの core storageは非常に貴重である。Chart 23 は516および316 ＩＭＰの core storage Map を示している。これは２４ページからなっているが，新しいバージョンでは更に４K，即ち３ページ位増えている。

例えば，データがＨＯＳＴからＩＭＰへ入るとする。これはこのMap中にあるＨＯＳＴ to ＩＭＰと呼ばれるプログラムで処理される。データはパケットに入り，キューに置かれる。HOST to ＩＭＰは各回線毎に２つのコピー（Multiple copy）が必要で２つのエントリがあり，合計約700語のプログラムである。このプログラムは，まずメッセージをうけ入れ，ヘッダーを理解し，そしてメッセージをパケットに分解し，さらにパケットのバッファースペースがあるかをチェックする。この場合，シーケンス番号やディレイなどを記した多くの表が内部に作成される。それと同じく，逆方向の場合，送られてきたパケットをプログラムが受け，それをメッセージにして，HOSTへ送るプログラムは IMP to HOST と呼ばれている。（ Fig 11）

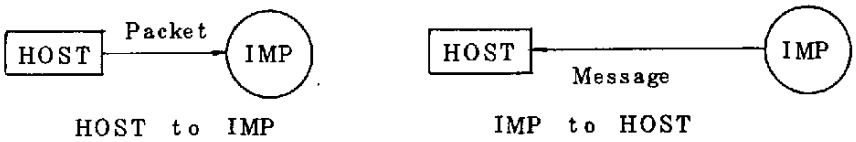


Fig 11

Modem to IMP（Fig 12）も複数コピー（multiple copy）が必要で，短かいプログラムである。我々のプログラムでは１つのコピーと３つのエントリがある。このバージョンではプログラムはまだリエントラントになっていない。この逆は同じように IMP to Modem でプログラムがパケットを受けとり，キューラインに置いて，返送されるACKを待ち，それによってまた再伝送するか放棄するかを判断する。

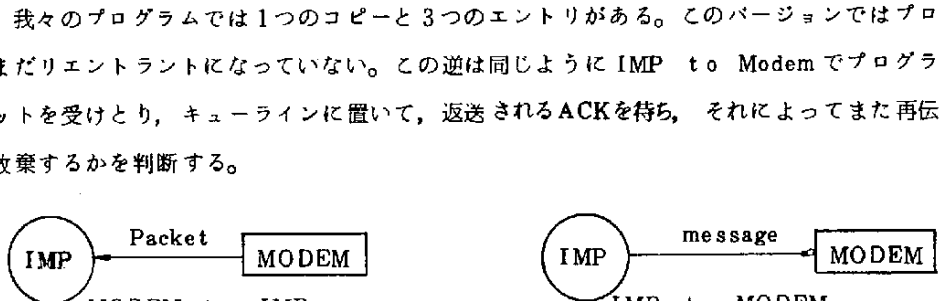


Fig 12

次にChart 23 の中央部にあるＴＡＳＫと呼ばれるプログラムを考えてみることにする。TASK は Fig 13 に示すようにＨＯＳＴ⇄ＩＭＰ，ＩＭＰ⇄MODEM等のプログラムの中央にあり，役割としてパケットを受けとり他のＩＭＰへ送り込むことや，パケットのリアセンブル（reassemble）や伝送の ACKを扱ったりしている。したがってHOSTから出されたパケットは一応ＴＡＳＫへ入り，そこでそのパケットに対する処理，即ち Fig 13 に示すように他のどのプログラムに処理をまかせ

るべきかを判断しパケットを送り出すわけである。ストア・アンド・フォワードの処理もここで行なわれる。

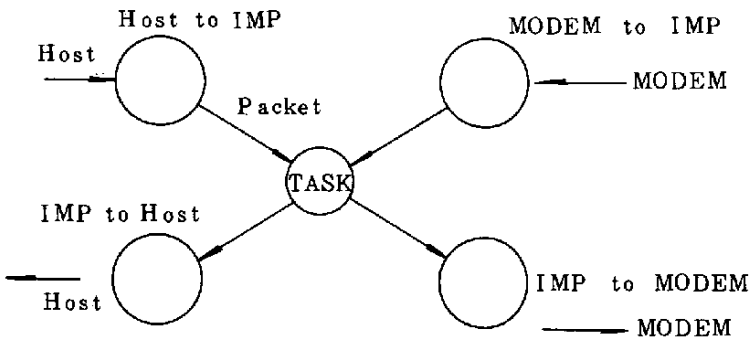


Fig 13

その他の機能としていくつかあるが特にプログラムの流れに関係がないので，詳細は省く。共通記憶（COMMON STORE）は，プログラムすべてに共通して使われる変数や連結されているIMPの番号などが入っている。

Core storageの2ページ目はプロテクトされており，これは作業中に，何かのせいでIMPのソフトウェアに間違いが見られた場合でもこのプログラムだけは生きており，ここにはreload routineが入っている。その場合このプログラムは，近くにあるIMPからコピーをとり，それをcore にストアして全プログラムを再生させるのである。このreload program を行なうのに50kb/s回線では5～6秒かかるのが普通である。

ダイアゴノスティック（DIAGONOSTICS）はキューの状態やその中のコード等をトレースする等のデバッギングのためのルーチンである。

テーブル（TABLE）はMODEM TO IMP のコネクションテーブルなどである。

イニシアリゼーション（INITIALIZATION）はたとえば，IMPプログラムが最初にロードされた時などに初めに実行されるプログラムである。何も別に起っていない時にはバックグラウンド・プログラムが普通は定常的に流れている。バックグラウンド・プログラム の仕事にはたとえばインプットやアウトプット・キューの状態やメモリー内容の変化などのトレース や特殊テレタイプのハンドリング等の多くの仕事がある。通常これらのプログラムを非常に速度 のはやいサイクルでループしており，何か特別な事が起らなければ，そのサイクルが続行している。もし他のプログラムの出動が必要になった場合にはそのプログラムにコントロールが移り，実行し，それから再びバックグラウンドへと戻る。

タイム・アウト（TIME OUT）は，クロックによるオペレーション終了を実行させるプログラムである。クロックはクロックパルス間隔が25.6ミリ秒であり，これはインターフェイスのハード

ウェアで構成しているものである。

クロックパルスの周期が来る毎に，クロックがプログラムに働らいてタイムアウトを起す。

タイムアウトには3種類あり，1つは25.6ミリ秒毎のもの，1つは5単位分即ち128ミリ秒もの，それにもう1つが25単位分のものである。

これらはそれぞれファースト・タイムアウト，ミディアム・タイムアウトおよびスロー・タイムアウトと呼ばれている。

これら種々異ったタイムアウトを使うのは扱うジョブの緊急度と関係している。

スロー・タイムアウトの使われるジョブにはリアルタイムが要求されないもの，たとえばルーティング・テーブルを更新するような場合などに当初使われていた。現在ではすべて変っているがこれについては後で述べる。オリジナルのシステムではそれは緊急に行う必要がなかったのでこうした更新の作業の際にはスロー・タイムアウトが使用されていたのである。このスロー・タイムアウトを使ったやり方でたくさんのホストや回線に対してその接続が切れていないかどうかを調べるためのテストが行われた。スロー・タイムアウトの周期は1秒の約10分の6以上になると思うが，殆んどの目的に対してこれはかなり適応したものであった。

ミディアム・タイムアウトを使ってチェックしたものは，再伝送を行なうような場合であった。キューにあるパケットが送られる時にACKが200ミリ秒以内に返ってこない場合，パケットの再伝送が行なわれるが，その時に使われるのがこの中間の128ミリ秒のタイムアウトである。

もう一つのファースト・タイムアウトは非常に使用度が高く，正確度が要求されるようなインターナルタイムの保持に使われている。ソフトウェアのタイミング用のクロックは50ミリ秒以内の精度を保っている。

以上がタイムアウトの機能についてである。

つぎに，システムの中には多くのアカウント機能が働いているが，これらはNCCリポートと呼ばれるプログラムに於いてつくりだされる。

またIMPに直接，接続されるテレタイプがあり，これはホストのものとは少々異っていて，システムがうまく働いていない時にそれをデバッグするために使われる特殊な目的に使われているものである。TTYプログラムはIMPに接続されているこのデバッグ用のテレタイプからディバッグのための交信をホストあるいは他のテレタイプとおこなうものである。

デバッグのルーティンはまた別のプログラムであって，これはこのデバッグ用テレタイプやネットワーク内の他のテレタイプに接続されて使うことのできるものであり，コアメモリやレジスタを変更することができる。このプログラムの使用には潜在的な危険性が存在していることから，このデバッグプログラムには多くの保護機構が備えられており，その意図に反したやり方で使用されないように保証されている。そのためのスィッチがコンソール上にあってコントロール・センターからプログラムを制御しており，二重のインタラクションではじめてそれが働くようになっている。

しかし一度動くようになると，記録されたメモリを読みとったり変更したりすることができる。

ルーティング・プログラム（ ROUTING） は非常に小さいもので遅延時間の計算はそれ程大きなプログラムにはならない。

統計プログラム（ STATISTICS ）はネットワーク及びIMPの性能を測定するためのもので，統計量が多く，かなり多くのテーブル・スペースを使うため相当な大きさになる傾向がある。

MESSAGE TABLE, ALLOCATE TABLE等はフロー・コントロールやシーケンシングに関係があるものだがこれについては後述する。

フローコントロール

前述のように，我々が現在ネットワークで使用しているオリジナルのフロー・コントロール技術には，次のような特徴がある。即ち，ネットワークに於いて，コンジェスション（Congestion）が起った場合，ネットワークが内部的にコンジェスションを起さないように，ネットワークからトラフィックを制止するわけである。このコントロールのアルゴリズムはいつも同じ状態であったわけではない。ここで最初に，従来のアルゴリズムはどうであったか。そしてそれがコンジェスションをなくすためにどのように変化していったかということについて述べてみたいと思う。

Fig 14 に示すように4箇所にＩＭＰがあるとする。トラフィックが左のＩＭＰから来て，ＩＭＰ1，2，3の径路を通ってゆくものとする。それぞれのＩＭＰの後にはＨＯＳＴコンピュータがあるものとする。1969～1972年の初期のフロー・コントロールのメカニズムは次のようである。

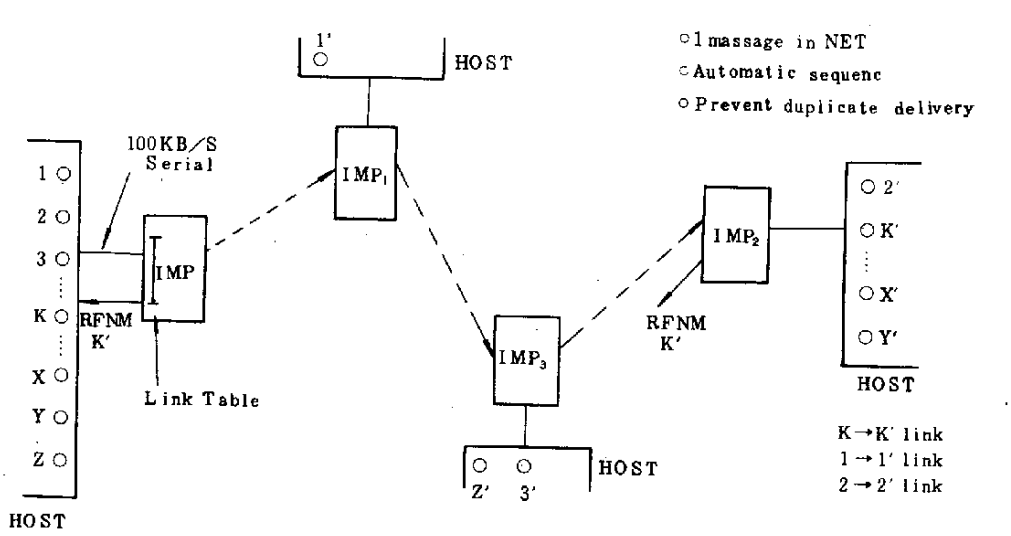


Fig 14

ここにプログラム1，2，3……K，……X，Y，Zがあり，プログラム1があるＨＯＳＴコンピュータに送られる場合，このコンピュータを1′とする。同様に，プログラム2に対応するものとしてコンピュータ2′，3→3′，K→K′，X→X′，Y→Y′，そしてZ→Z′と対応させることにする。

コントロールは，プログラム毎の単位をベースとして行なわれる。したがって，KとK′の間に一度に一つだけのメッセージが，ネットの中で許されているわけである。Kからメッセージを K′に送る場合，Kはこのメッセージが入り始めＲＦＮＭが返ってくる時まで待機しなければならない。この場合，K′のＲＦＮＭはＩＭＰ2を通して戻ってきてスタートＨＯＳＴに入る。このＲＦＮＭ

が返ってくると，K′ に対する次のメッセージを送ることができるわけである。

この伝送が行われた場合，ＩＭＰがK′ に対する新しいメッセージの伝送を，ＲＦＮＭが返ってくる迄ストップし，ＨＯＳＴを待たせておくという働きをしている。

一方，その間でもXはX′ にメッセージを送ることができる。プログラムKと同じホスト間ではあるがメッセージを送ることができる。勿論同時に1からのメッセージを1′ に送ることも許されている。このフローコントロールの機構では，送り先に現われるメッセージは一度に１メッセージと限られているので，コンジュスションがプログラムによってひき起されることがないという特徴がある。また一度メッセージが送り先に入り始めると，確実に連続して行われ，順序が狂ってくることがないところから，いわゆる順序付け（Sequencing）の問題はなかった。即ちこのシーケンシングは自動的に行なわれたわけである。

さて，一度に一つのメッセージが送られる場合のKとK′，あるいは1と1′ の間等に於ける各コミュニケーションは「リンク」と呼ばれる。K対K′ は１つのリンク，そして１対1′ は同様に第２の別のリンク，２対2′ は同じく第３の別のリンク等々である。もともと存在し得るリンクの数は０から６３である。

これに関するフォマットはFig15のようになる。

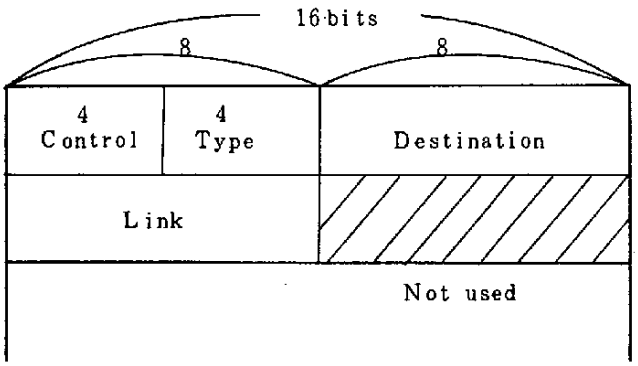


Fig 15

この１６ビットのメッセージが連続的に送られる場合を考えてみよう。ＩＭＰとＨＯＳＴの間でこのリンク情報が平均毎秒100キロビットでシリアル（serial)にハンドシェイキングの手順でおくられるが，この１６ビットのグループは８ビットずつ２つに分けられる。左の８ビットの内，右４ビットはどんな型のメッセージであるかを表わし，左の４ビットは制御ビットである。次の８ビットはリンクである。つまりホストに於いては $2^8 = 4^4 = 256$ 個の数のうちの１つをとりだすことができるわけである。

このシステムのＩＭＰでは現実には一時に６３だけがアクティブになる。各ＨＯＳＴでは256ま

でのリンクが存在できるわけであるが，そのうち４分の１だけが一度に使用できる。しかし，これらは循環的に使用することができる。

このＩＭＰプログラムの機構の中にはリンクテーブルがあるが，これは，各リンクのフローを調整するものである。このリンクテーブルは，全てのシーケンシングの為に使われているが，これはＲＦＮＭの返ってくる前にメッセージが送られないように確認を行うために必要であり，送信リンクテーブルと受信リンクテーブルとがある。ＲＦＮＭが返ってくるまではＩＭＰがメッセージのデュプリケート（複写）を持っており，ＲＦＮＭが返ってくれば，それを消す。このコントロールもリンクテーブルがおこないデュプリケートが再送されるのを防いでいる。

リンクテーブルに入るエントリーは，シーケンスナンバー，タイムなどいくつかのインフォメーションを持っている。タイムはタイムアッププロセデュアのためのものである。各々の送信リンクテーブルは，４８ビット幅×６４ビット長のものである。

あるＩＭＰにパケットが受け取られると，受信リンクテーブルにエントリーが入り，そのシーケンスナンバーとか，タイム等が記録される。リンクテーブルでこれらのインフォメーションを充分長い時間保持した場合はソースをタイムアウトにすることができる。仮定ではどんなメッセージも１つのシステム端から他端まで１５秒以内に到達することが可能であるはずである。このシステムは，遅延（ delay ）が0.2秒に設計されている。メッセージが１５秒以内に目的地に送られないということがあれば，それは何等かの故障でネットワークの多くの部分が障害を起した原因以外には考えられない。

したがってもしパケット（或いはメッセージ）が，１５秒以内に１つのシステム端から他端へと送られない時には間違いなくこれはタイムアウトである。こうしてテーブル上に長く保持されたパケットは不完全な伝送ということから再度，伝送のインプットとして送り返されることになる。

この種の機構は，もし第一にトラフィックが少ない場合，第二にもし個々のリンクの上にシーケンシングを持たせたい場合に，非常に有効なものといえる。

しかし次にあげるような問題が起ってくることも考えられる。

Fig 16 に示すようにいくつかのＨＯＳＴ（ H_1 H_2 ……H_N ）とそれに対応するＩＭＰ（ I_1 I_2 ……I_N ）とがあり，全てメッセージは一つの方向即ち H_X に流れているものとする。

ネットワーク内のＨＯＳＴは，パケットを受け取るのに充分な時間さえあればいつでもパケットを取り入れ得ることになっている。ＩＭＰがパケットを保持する最大時間は約３０秒となっており，ＩＭＰが３０秒間，パケットを保持し，パケットの到達速度が毎秒100キロビットであるとすれば，３０パケット（３０メッセージ）の到達時間は，もし２つの５０キロビットラインで伝送されると仮定すると，およそ１０（ミリ秒／パケット）×３０＝300ミリ秒，パケットがシングルラインで送られる場合は600ミリ秒となる。

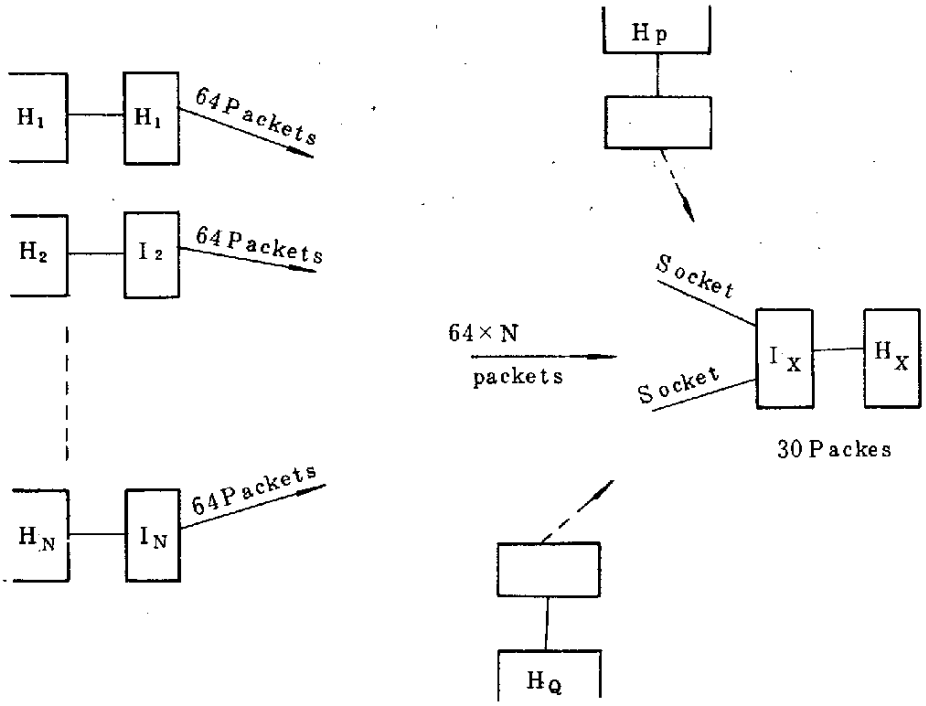


Fig 16

したがって３０パケットが300ミリ秒で到達することができ，一方，仮定としてＨＯＳＴは30秒の間，待機することができるという状態である。もし，３０パケットが到達するのに300ミリ秒かかるとして同じくＨＯＳＴが３０秒待機可能であるとすればこの場合にはパケットのフィルアップ状態になることが考えられるであろう。オリジナルのネットワークにおいては，ＨＯＳＴのバッファーは３０パケット分，あるいはその２倍のものもあるが，いづれにしろこうして全体をフィルアップしてしまうことが起り得るわけである。これら全てのＨＯＳＴがそれぞれ，１リンクにつき１つのメッセージ（１パケット）を送っており夫々６４パケットづつを各々違ったリンクで送るとする。到達するパケットの総合計が６４のN倍になり，ネットワーク全体は，非常に短かい間にパケットにフィルアップされてしまうということになる。例えばH_PやH_Qからの他のトラフィックは不可能となる。なぜならネットワークは完全なコンジェスション状態である。ここで私が話しているのは，ある径路を通るＨＯＳＴの遅延がこのようにネットワークを通して拡大してくるということである。

この問題が，オリジナルの機構に関する基本的に困難な問題の一つである。いいかえればトラフィックがかなり少ない場合には，この機構は最高に良く働く非常に有効な機構であるといえよう。実際にはこうした種類の問題はトラフィックが充分多くなかったために，一度も見られたこともなく，ネットワーク内でこういう形で起ったことはなかった。

しかしトラフィックレベルが充分大きくなけれ多箇所でのコンジェスションがネットワーク内に起ってきて，あるＨＯＳＴに送られたパケットがもはや素早くさばけなくなり，他のパケットを送るのに非常に大きい遅延を生じてしまうという結果を引き起すことが考えられる。事実，タイムアウトが３０秒以内に起れば，パケットは放棄されてしまう。

もう一つ，こうしたネットワーク内に起ってくる問題がある。つまり，これはもっと難かしい種類の問題で，リアセンブリ・ロックアップとして知られてきた。

リアセンブリ・ロックアップ

目的地ＩＭＰに於て，３つの異ったメッセージがリアセンブリされる場合を考えよう。Fig 17 に於てメッセージ１が送られてきて，これが３つの部分に分かれて１ａ，１ｂ，１ｃというパケットとして入って来る。またメッセージ２が３つのパケットたとえば２ａ，２ｂ，２ｃ 等と分れることにしよう。さらにたくさんのメッセージがあって，例えばメッセージＫがＫａ，Ｋｂ，Ｋｃ，Ｋｄ というように分けられるとしよう。

ここで目的地においてはメッセージが到着し，メッセージ１に対してリアセンブリをおこなうため各々１ａ，１ｂ，１ｃ に対してバッファーの保有が行なわれる。オリジナルの機構に於いては，これらのパケットのうちいづれか１つが到着した時，バッファーのリザーブが行なわれる。

１ａが到着した時，他の２つのパケットに備えて３つのバッファーがアロケートされる。図のように１ａはこの１つに入るが，あとの１ｂ，１ｃのメッセージは入ってこないという状態であるとする。実際に，オリジナルの機構に於ては１つのメッセージ用に８パケットを使ったわけで１ａの到着の際に８バッファーが用意されるが，１ｃが入ってきた時に残りの５バッファー分が返却されるわけである。オリジナルの機構では，１つのメッセージがどれだけのパケットを持っているかが知らされないので，パケットの数が１つ以上の場合，８であると仮定するのである。

さてメッセージ２の場合だと，２ｃと２ａが入ってきて２ｂのためにあと１つのバッファーが用意されることになる。同様に，メッセージ４の場合，４ａが入って来て４ｂ，４ｃのためにあと２つのバッファーが用意される。

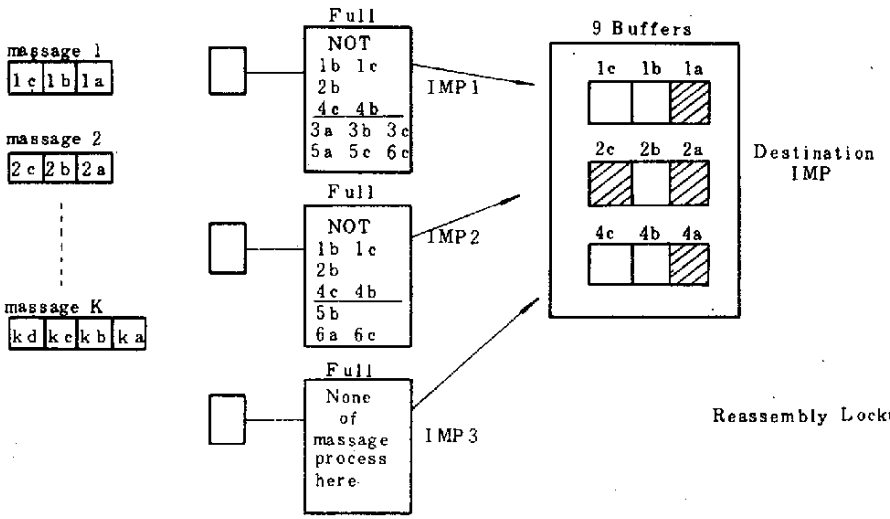


Fig 17

さて，ここで目的地ＩＭＰには全部で９バッファーしかないものと仮定して話をすすめていこう。そして，これら４つのパケットの到着後はメッセージ１，２，４の未到着パケット以外の他のパケットが着くことはない。

図のようにＩＭＰ１が１ｂ，１ｃ，２ｂ，４ｃおよび４ｂを持っておらず，ＩＭＰ２も同様に１ｂ，１ｃ，２ｂ，４ｃ，４ｂを持っていないという場合を考える。この時実際にこれらのＩＭＰが持っているパケットは各々，３ａ，３ｂ，３ｃ，５ａ，５ｃ，６ｂと，５ｂ，６ａ，６ｃ等々である。これらは各々フィルアップされた状態であり，しかもメッセージ１，２，４の未到着パケットはそこにない。ＩＭＰ３もフィルアップされた状態にあるがここにもメッセージ１，２，４の未到着パケットは入っていない。目的地に更に別のリアセンブリング・バッファーが存在しない場合には，このような状態に容易に陥ることになるだろう。そこでは隣接するＩＭＰが全て，フィルアップ状態にありながら，未到着パケットを１つも有していないという状態ができている。これらＩＭＰの２つ隣あるいはそれ以上離れたＩＭＰに未到着パケットがあるとしても隣接ＩＭＰがフィルアップ状態のため通りぬけができない場合どうすることもできないわけである。

こうした場合にはパケットを逆方向に返して他のパケットを入れるようにするしか方法がない。つまりパケット２ｂを持っていないこのＩＭＰはどこにその２ｂがあるかを知らないだろうからである。

パケット２ｂがずっと離れたところにあってそれがそのあるべき位置に到達するまでどの位の時間を要するかわからない場合，実際には，どのようにしてこのパケット交換をするか，その一般的方法はないであろう。実際，これは通信施設を使う上で非常に非効果的なものである。

Chart 24 も同様な例でＩＭＰ２が一ぱいのため，ＩＭＰ１にあるＡ１とＢ２がＩＭＰ３に運べず，お互いにロックアップしている。

このリアセンブリ・ロックアップがどの程度の割合で実際に起ってくるかという点について質問があるかと思う。さきほどコンジェスション状況のところで説明したことを思いだしてほしいが，コンジェスションは単一のあるＨＯＳＴが非常にゆっくりとしかネットワークからパケットを受け取れない場合に起される。また単一のHOSTに多数の他のHOSTから交信する場合にもコンジェションンが起される。また前の機構の中でネットワークをトラフィックであふれさせるとコンジェションンを起すことができる。

一方ＨＯＳＴにリアセンブリ・ロックアップを起させることが容易にできるかという質問だがその通りである。実際非常に容易にできる。コンジェスチョンを起させるよりももっと簡単にリアセンブリ・ロックアップを起すことができる。

フローコントロールのところでも述べたシミュレーションについて考えてみよう。ARPANETでは多くのシミュレーションが試みられ，その結果の正当性が良く評価されている。これもその例

の１つである。Chart 25 に示すように六角形のネットを考えてAからA′へ８パケットメッセージのトラフィックを送る。８パケットのうち４パケットがA→A′のルートに沿っていくと同時に残り４パケットはもう一つの長いルートの方をとって進む。これはルーティングが２つの経路を許すものだからである。

このようなネットでリアセンブリ・ロックアップは１１(10か11)以上のリンクをランしたときに起ることがわかった。Chart 25 の上方にスループットとリンク数の関係が示してある。単一リンクのランに伴うメッセージは，システム内に於いてある一時刻に１メッセージだけが存在し得るわけであるからロックアップの起る機会はない。

ここには３２のリアセンブリ・バッファーがあり，この場合８パケットのメッセージの内４パケットづつが１つのルートを通りいづれも一度にリアセンブルされることが可能である。

１つのリンクだけだと，スループットの割合は毎秒３０～３５キロビット程度で，これが１つのリンクの伝送に対する理論的限界である。特別の場合に２つのリンクを使用するとすれば，これは図に示す様にトラフィックの変化は前よりも大きく，秒あたり５０キロビットよりわずかに小さいものとなる。

３リンクの場合，基本的には同様の範囲内にあり，４リンクではトラフィック変化は毎秒４５キロビットあたりから６０キロビット以上になる。それから少しの間，リンクの数を増やしてもそのままの状態でとどまっている。

５～６リンク以上になってくるとスループットは減少しはじめるが，リンクが９の場合でもなお，常時良好な動作を示しており，ロックアップの問題はなかった。ところがリンク数が１０のところで最初に×印が軸上に交さしてくる。実際に１０のところでロックアップがおこったし，１１のところでもおこっている。上方に×がある時はロックアップは起らなかったケースである。１２では常時，ロックアップされた状態であり，ランのない状態である。

ロックアップの起る迄に１２パケットは送らなければならないが，大体20~50 パケットのかなり少数のパケットでただちにロックアップ状態になってしまう。それ以上の数だと勿論直ちにロックアップが起ってしまう。実際，最初にシステムをつくった時に，ネットワークに属する全部のコンピュータに対して，１つのプログラムが１リンク以上を使用するということのないようにしておいた。

したがって，つまり上記のような状態にするためには，コンピュータが単一の他のホストへできるかぎり迅速にデータを送るプログラムを１０或いは１２持っていなければならないということである。しかしネットワークが使われはじめてから何年もの間，この様なプログラムを作った人はいない。これはある特定の特殊なテストを行った場合にのみ起り得るのである。

ストア＆フォアワード・ロックアップ

システム上の困難な問題の1つとして，まずコンジェスションについて，そして2番目にロックアップの形について述べた。その1つとしてリアセンブリ・ロックアップについてのべたが，もう1つのロックアップの形として，ストアー＆フォアワード・ロックアップがある。

1. Congestion
2. Lockup
   a. Reassembly
   b. Store and Forward

このロックアップは一般的に次のような性質をもっている。

最初のシステムではＩＭＰの中で，全てのバッファーを1つのキュー（Queue）と考えた。即ち，バッファーは何の制約もなしにリアセンブリにも使われるし，またストアー＆フォワードにも使用されることにした。しかし我々は少なくとも，いくつかのバッファーはストアー＆フォワード用に，そしてあるバッファーはリアセンブリ用に分けてとっておくべきであるという結論に達した。

ストアー＆フォワード・ロックアップの状態を作ることは可能である。Chart 26 にストアー＆フォワード・ロックアップが起り得る場合の図解がある。相隣る2つのＩＭＰに於てどちらにもフリーのバッファーがないとすると1つのパケットも一方から他方のキューへと移動することができない。また同時に隣接のＩＭＰから，このいづれかのＩＭＰへのパケットの移動もない。従ってすべてのバッファーが同じキュー上に占有されつづけているとすれば，パケットの交換の可能性は全くない。

1つのバッファーをフリーにしさえすれば，これを簡単に避けることができるということがわかる。従ってこれを避けるために，或いはこの状態を瞬時に経るために多くのメカニズムを採用することができる。

Chart 27 はストアー＆フォワード・ロックアップによるコンジェスション状況をあらわしたものである。

こうした状況が起るのはネットワークが，たぶん回路の障害あるいは設計の欠点によるものと考えられるが，図に見られるような（  ）形のトポロジーの配置をとっている場合であり，ここでBとB′から互いにトラフィックを送り合っている。Bはデータを B′ に送り，B′ はBにデータを送り返す。同様にAとA′も互いにデータを送り，送り返している。トラフィックが多重となるノードCとDに於いて各々の回路が同じ速度をもっているので，５０キロビットの回路で各方向のトラフィックとして100キロビットの速度が必要となる。そこで待機している全てのバッファーはたちどころに一パイになる。

図の上の曲線は使われているリンクの数の函数として，システムに於けるスループットをプロットしたものであるが，ここで単一のリンクに対してのスループットはおよそ毎秒３０キロビットと

いう値が得られる。

ここでは1000 ビットのメッセージであるとし，前回の８パケットのメッセージではなく単一パケットであるとする。ここで仮定として無限大のリアセンブリ・バッファーがあり，リアセンブリロックアップの問題はないものとしている。

ストアー＆フォワード・バッファーの数をある有限の数，ここでは２１であるとする。

２リンクの場合には，トラフィックが毎秒５０キロビットをわずかに超えている。おそらくちょうど５０キロビットではないかと思う。

６リンク以後になるとわずかに落ちこんでそれから一種奇妙な状態でしばらくつづき，リンクが１１のところまでそういう状態であるが，11リンクを起点としてロックアップが始まりだんだん頻度が高くなる。このようにしてロックアップを起すことは可能であるが，ネット内部でのトポロジーが，この種のロックアップを引き起さない形をとることがまず必要である。しかしネットの一部の障害でこういう形になる可能性はある。

今ここで述べたのはダイレクト・ストアー＆フォワード・ロックアップについてであり，実際にはこの他に Co-indirect のストアー ＆フォワード・ロックアップと呼ばれるものがあるが我々はこれを起すことがかなり困難であった。

フローコントロールの改良

このように最初のメカニズムでは以上のようなさまざまな問題が起ってくることがはっきりしてきたため，我々はオリジナルのフロー・コントロールの機構を修正することを決めた。まず充分なソースからトラフィックが多量にネット内へ送られている場合についてであるが，ここで概念を広くして，あるリンクに対して一度にネットワーク内に存在するメッセージがただ1個であるとは考えないことにする。

実際に我々は完全にリンクの概念を取り除いてしまった。1972年 初めにフロー・コントロールの機構が変ったときにはサブネットワークに関するかぎりリンクというものをもはや考えなかった。しかしＨＯＳＴは全てリンクを組み込んだプログラムを使っていたため，新しい機構に於いても，従来のものとコンパラブルでなくてはならず，そのためＨＯＳＴ内ではそれが丁度リンクであるかのように働き，全て以前の働きと同じ効果をもたらすものであれば問題はないわけである。

ＨＯＳＴのプログラムはなおリンクのような働きを示しているが，ネットワーク自体はリンクに少しも関知しないというものである。

実際に使用している手段に一種のスペース・リザーベーションのテクニックがある。Chart 28にこの手順を示す。

現在のバージョンにおける基本的な動作は次のようなものである。

ここではしばらくただ1パケットのメッセージの送られる場合を考えて，そのあとで複数個のパケットのメッセージについて考えることにしよう。

パケットのコピーがソースＩＭＰに保持され，その間にオリジナルのパケットはネットワーク内を伝送される。それが受信側に到達すると，もし受信側にそのパケットを保持するバッファースペースがあれば，オリジナルが受信され，ＲＦＮＭがネット内部に戻されHOSTへと返される。

ＲＦＮＭがソースのＩＭＰに返ると，ＲＦＮＭはソースのＩＭＰに働いて，コピーを捨てる。これが通常の状態でのプロセスである。

さて受信側に於けるオリジナルのスペースについて考えられることであるが，オリジナルが受信側に到達したときに，コンジェスションが起る場合がある。すなわち，受信側に過剰のトラフィックがやってきて，しかもそのパケットを運ぶ速度がかなり速い場合，或いは，受信側でパケットを取り込む速度が非常に遅い場合，そのいずれの場合でも，バッファースペースはなく，ネットワークの他の場所でコンジェスションを引き起す可能性があり，それを防ぐため，受信側はそのオリジナルを放棄（discard）するということを行う。それは，ソースにコピーが保持されているからである。そして受信側でスペースが使えるようになる迄待って，使用可能となれば即座にリザーブする。それから再びソースに対して，放棄されたパケットのコピーを送るように指示を出す。

実際には次のように行われる。

受信側にはリクエストに対するテーブルがあり，パケットがやって来るとそのテーブルに「ここ

に１つのＩＭＰ或いは多数のＩＭＰからのリクエストが待機しています」という内容のエントリが入る。そしてＩＭＰが５つのパケットを送りたいというと，５つのバッファーが空き次第，それはリザーブされて，ＩＭＰには「５バッファーを貴方のためにとりました」というメッセージが返り，ＩＭＰはパケットを送り出す。この場合受信側に用意されている５つのバッファーの全てを使うのだが，実は５以下なら数はいくらでもよいのであって，たとえば１つでもいい。この状態では，スペースが確実に先方に確保されていることがわかっているので，パケットを送った時にＩＭＰではコピーを消してしまう。そして受信側ではバッファーにパケットが入ってくると集められ，HOSTに送られる。それからＲＦＮＭが返ってゆく。

ＩＭＰには送られるのを待っているパケットのコピーがキューに保持されている。ＲＦＮＭが返ってくる時はパケットのヘッダーだけを一しょに持ってくる。したがってＩＭＰにＲＦＮＭが返って来た時そこにある各パケットのヘッダーと比較され，正しいものが見つけられる。また各々のHOSTに入ってくるパケットに連続番号が付けられ，その連続番号もまた送り返される。したがって，リンクナンバーを見なくてもどのメッセージであるかを見分ける方法があるわけである。

このネットワークのいくつかの非常にすぐれた性質の１つとしてネットワーク内での遅延時間が可能な限りに短縮されているという点があげられる。それは，なにも設定（ setup ）時間というものを要しないからである。パケットはただ一貫して流れており，もしコンジェスションが起きそうになってもその時点でコンジェスションを防ぐために流れが遮断される。すなわちある部分で全てのバッファーがフィル・アップ状態になることがあっても，ネットワーク内の他のパケットが更に蓄積されることはないわけである。というのは，全てのバッファーが満員状態のときにその箇所へ入って来たパケットがあってもそれは放棄されてしまうからである。したがってこれで全てのコンジェスチョンが非常にうまくHOSTでペンディングされる。これはリンクがある場合もない場合も同じことである。

またあるＩＭＰのバッファースペースが１つのＨＯＳＴに対するものでフィル・アップされる場合もこれによって他のＩＭＰのストアー＆フォワード・スペースに対する影響は無い。即ち，このＩＭＰには一時的には他のパケットがやってくるが，これはすぐ放棄されるのでネットワークのトラフィックには何ら荷重がかからず，他のＩＭＰには結果としてコンジェスションは全然見られないということになる。

以上はシングルパケットメッセージの話であった。シングルパケットに対するネットワークのディレイは10msから15ms位で大変早い。また一般にシングルパケットの場合はパケットがＩＭＰにやってきてリザーブをして返っていくというような手続きは不用である。こうした手順の必要な場合というのは，全てのバッファーがフィルアップ状態になり，新しいパケットをネットワークが受け入れない限りネットワーク内に入ってこられないことを確認したい場合だけである。

複数個のパケットメッセージの場合，望ましいのは広域バンド巾通信ができることである。即ち

可能な限り多くのデータを流し得ることである。複数個のパケットの場合でも，もしバッファースペースに関係しないとするときほど問題はない。ここで問題となることは，例えば８パケットのメッセージを送った場合にソースに８バッファー分のコピーを残しておかねばならないということである。

各々のメッセージが長いとバッファースペースが非常に急速に増大する。最初のインプリメンテーション時に各ＨＯＳＴのオリジナルなシステム内では特別にメモリーをつけ加えなければ受信，送信のメッセージのための充分なバッファーをそなえうるものはなかった。

そこで我々が最初に考えた方法は，これはやがて変更されるものと思うが，パケット№１，即ち複数個のパケットメッセージの最初のパケットがソースＨＯＳＴからソースＩＭＰに到着した時にこのソースＩＭＰではこのパケットが単一のパケットメッセージであるか，複数個のパケットメッセージであるかを見分ける。これはそのパケットの終端でHOSTのインタフェースから最後である旨のビットインディケータが得られることによってわかる。そして，もし最初のパケットの終端部でそれが最終のビットである場合，即ち単一パケットである場合にはコピーをＩＭＰ内に留めておいてネットワークを通して送り出す。そしてＲＦＮＭが返ってくるか，或いは，スペース・リザーベーションがおこなわれたと伝えられる迄，待つ。

メッセージが１パケットより大きいものであるとわかるとインタフェースが停止して，（つまりパケット１個だけを取る）パケット№１がネットワークに送られている間，HOSTからそれ以上にメッセージがとりだされることはない。

パケット№１が受信側に到達すると，受信側ではこのメッセージに対して８個のバッファーをリザーブする。そしてこの８バッファーがリザーブされると，先方から「８バッファーがリザーブされました」というメッセージが送られて来て残りの７個のパケットがネットワーク内に入ることができる。そこで最初の１個のパケットがネットワークを通過してその返信が返ってくる間，ホストラインは停止状態におかれている。このような典型的な伝送の仕方でHOSTの一時停止が行われるわけである。

通常，８パケットがネット内に入るには，およそ８０ミリ秒かかる。しかし私が今述べたような状況を考えると，８０ミリ秒ではなくなるわけである。１つのパケットが入って来るのに１０ミリ秒かかり，そこで５０ミリ秒の遅延が生じたとすると他の７パケットが７０ミリ秒でネット内に入ってくるとしても伝送には８０ミリ秒かかるわけで，全体ではたぶん130ミリ秒かかるということになるであろう。残りのパケットが伝送可能かという確認を得るのに少々時間をとるわけであるが，これによって内部的にコンジェスチョンが起るのを防ぎ，またリアセンブリ・ロックアップを防ぐことが保証される。つまりネットワーク内に入ることのできるメッセージに対するスペースがあることが常に確認されてから入ってくるからである。

ここで私が述べたのは一応の結論としてであって，５０ミリ秒もの間HOSTを待機状態にしてお

くことは理想的ではない。つまりあとの残ったパケットはそのままHOSTに待機させられるわけであるから。しかし実際はこのことに関してはそんなに困難な問題はなかった。むしろ我々の直面した最大の問題は我々が非常に僅少数のパケットに対するコピーしか保持し得ないということであった。これはメモリー不足のためだったのにすぎないが，最初はコピーを保持し得たのは4個分だけで，これを8個にすると少しは助かるが，それでも充分であるということまではいかない。これをうまくやるには多量のバッファーを持つか，HOSTに返すかである。

そこで我々は，新しいプロトコルとしてHOSTに再伝送の機能を持たせることを考えている。サブネット内に於て，メッセージが再伝送される必要がある場合にはサブネットからHOSTにこのメッセージを送り返すことができるという性質をもっている。

以上述べたようなリアセンブリ・ロックアップやコンジェスションの両方に対処して使用されているフローコントロールの形式を私はEnd to Endフロー・コントロールと呼んでいる。これは送信側及び受信側に於いて起ることに対してフローのコントロールをする方法である。

内部の問題としてストアー&フォワードのロックアップへの可能性について考えるといったような問題になるわけである。

前述のように単一のバッファーをインプット・キュー，或いはアウトプット・キュー，或いはそのいくつかの組みあわせたものに対してうまく割り当てることによって，ストアー&フォワード・ロックアップを起きないようにすることができる。実際問題としてはネットワーク内に非常に大きいキューを蓄積したくはないわけであるから，最低必要量より多く，たぶん1.5個乃至2個のバッファー分だけ多く得るためにモデム・チャンネルにキューをためておく理由はなにもない。もし4個のバッファー分だけで完全にラインが占有される場合を想定すると，このＩＭＰ内部に4個以上のバッファーの存在し得る理由は全くないわけである。

もし必要があって4個の代りに１６個であったとすれば，１６個を送り出すことはできないから伝送されるのを待たなければならないわけだが。その場合１つか２つ手前のＩＭＰで待たしておいた方がより早く目的地に到達するので良いと思われる。さらにこの議論を進めて，バッファーを送信側ホスト自身に待機させたままにしておいても結局同じ程度の速さで到達できるのではないかといわれるかもしれない。それが実は我々がとった方法である。

バッファー数への考察

あるＩＭＰから隣のＩＭＰにパケットが送られるとその到着に対する肯定応答ＡＣＫが返される。ＡＣＫが返ってくると，送られたいくつかのパケットについて調べて比較，確認し，その送られたパケットをキューから削除して再び送らないようにする。もしこのＡＣＫが待っても返って来ない場合にはリストを注意深く調べて受け取られなかったパケットを全部再送する。

いくつかの困難な問題があった。その一つは，これらの送受信用の１つのチャンネルに存在するバッファーを何個にするかといった制限をしなかったことである。

そこで次のようなある機構を考えた。それはキューに８パケットがあるのではなく，キューが８チャンネルを有していると見なすことである。勿論これは論理的に考えると異なる。ここでこのチャンネル上にパケットを順序付けしてゆく。50キロビットの回路の１回線を考えてＡ～Ｈのチャンネルにそれぞれ第１番目のパケット，２番目，３番目のパケット等を入れるが，これで，全体の回路を効果的に使うことができるわけである。プログラムとしては，割り込み無しでスキャンと継続的な再伝送を行うだけのものである。これは我々がＶＤＨに対してとった方法である。

ＡＣＫを受け取るとすぐに例えばチャンネル３のパケットが放棄され，また新しいパケットがこのチャンネルにきて伝送される状態となる。

そのチャンネルに新しいパケットが入るとフェイス・ビットが０から１へと変えられる。各々のチャンネルに０と１のフェイス・ビットがあって新しいパケットが入る毎に旧→新→旧→新とそのビットの反転が見られるわけである。そして配分されたチャンネル数と同数分のパケットだけはここで待機状態にしておくことができる。こうしてＡＣＫをシステムの伝送部から独立させることが可能となり，これは非常にうまく分離されることになる。

これはプログラミングの点から見ても非常に効果的であると云える。ＡＣＫはチャンネルのフェイス・ビットをもっているのでＡＣＫが返ってきた場合にいつでもすぐそこにあるものと比較すればよい。またＡＣＫはそのパケットが必要であるか否かを判別するのにも使われる。多くの複写もこのチャンネルを通して送られるわけであるから，インプットのプログラムではオリジナルのパケットが入ってきたか，複写が入ってきたかという点を見定めるために各チャンネルのフェイス・ビットをじっと見つづけておく必要がある。

伝送の極端なバースト状態に於いて多くのパケットが失われるような場合，次のパケットが来る迄に要する時間は極めて短かいものである。というのは，前の場合のように200ミリ秒待機するのではないわけである。たとえば１つのチャンネル上に１個だけのパケットがあった場合を考えてみると，そのパケットをいつまでも待機状態においておき，回線がクリアされるとすぐにパケットが入ってくるようにし，このチャンネルをフルスピードで運転させることができる。したがってこの

ようなバーストの場合にはパケットを得るのに要する時間は200ミリ秒から40ミリ秒へと減少することになる。遅延特性が向上し，インプリメンテーションが簡単であり，さらにバッファーの必要数を最小にすることができ，回路利用率をコントロールするのは非常に有効な方法であることがわかる。

また融通性があるということが云える。つまり，回線が短かい場合には数を2とか3に減らせば良いし，衛星用に使う場合には10，20或いは30に増やすことができるわけで，完全に汎用性がある。そしてこの数は，回線数の倍数以上にする必要はない。また各回線に対してバッファーが常に確実に用意されており，今まで一度もストア&フォワード・ロックアップを経験したことはなかった。

ルーティング

最初にルーティングのアルゴリズムの本来の目的は何であったか，また，アルゴリズムが僅かではあったが変更されたその修正部分はどこであったかということから始めよう。

アルゴリズムのもともとの目的としてはメッセージを伝送する場合，そのルーティング・プロセスに発信地と目的地の情報のみを与えて，送り側から受け側へとパケットを送ることであった。そして内部のルーティングに関して，送信側は何も知らなくてよいようにするべきである。

最初に決定していたメカニズムは最小の時間遅延を可能とするルーティングのアルゴリズムである。パケットがある発信地からある目的地へと運ばれる場合，その最小の時間遅延が推定され，その遅延が最小になるような経路上の各ノードを経てパケットが送られる。

このアルゴリズムはコントロールの中心をもっていない。即ち，どのノードもルーティングに関しては全て同じ機能しか持っておらず，インフォメーションが1つのＩＭＰからその隣接するIMPへと渡っていき，それから先のルートまでコントロールする機能はない。ルーティングのパケットは，ただ隣接ＩＭＰへと送られるだけであるからアドレシングは何等行われなかった。

ルーティングのパケットは過剰のトラフィックがネット内で起らないようにコントロールされており，ルーティングは内部測定と隣接ＩＭＰが送るインフォメーションとを基礎としたものである。ルーティング・インフォメーションは互に隣接するＩＭＰの間でのみ送受が行われる。

オリジナルのインプリメンテーションにおいては，ルーティング情報は2分の1秒毎に送られ，これはスロータイマーによって制御されていた。

つまり，625ミリ秒毎に新しいルーティング情報が送られていた。このため，我々はルーティングに関してかなり注意深くコントロールすることができた。

またルーティングのインフォメーションがネットワーク内を伝わってゆくのにどれ程の時間を要するのかということを正確に決定することができた。さらに，ルーティングが毎秒約2回の割合で起るが，ネットワーク内の回線或いはＩＭＰの動作開始あるいはダウンのときにもこのルーティングを適応させることができる。もし，ある回線が故障した場合，ルーティングメッセージの1つがその分のインフォメーションを含んでいて，それが次のＩＭＰへと送られることになる。こうしてトラフィックが，故障したＩＭＰを通るような経路を取らないで，このダイナミックなメカニズムを利用して，さらに効果的に，他の経路を通るような再ルートの手順を得ることができるわけである。

送られるルーティング・メッセージはFig 18のような形のものである。トップの1ワードは，ネットワークで統計が取られる場合に，同期を行うために使われる。これを用いて全てのＩＭＰについておよそ20～25ミリ秒以内のクロックの同期をとることができる。

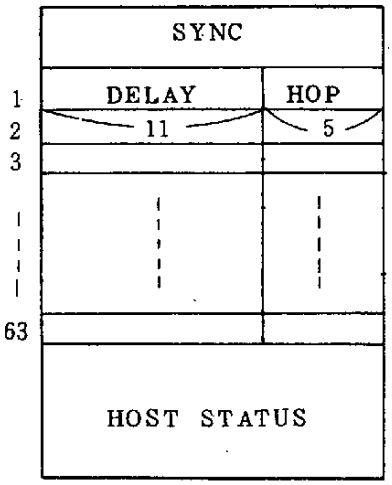


Fig 18

ルーティング・メッセージは2個のデータから構成されておりその1つは5ビットのホップ（HOP）数でもう1つが11ビットのディレイ（DELAY）数である。

ホップ数とはそのノード（IMP）から他のすべてのノード（IMP）に最少幾つのノードを経てゆけるかという数である。例えばFig 19に於て各ノードを1，2，3，4，5，6，7，8，9すると，＊からノード7，9，6えのホップ数は1，8，1，5へのホップ数は2，2，4，3えは3となる。

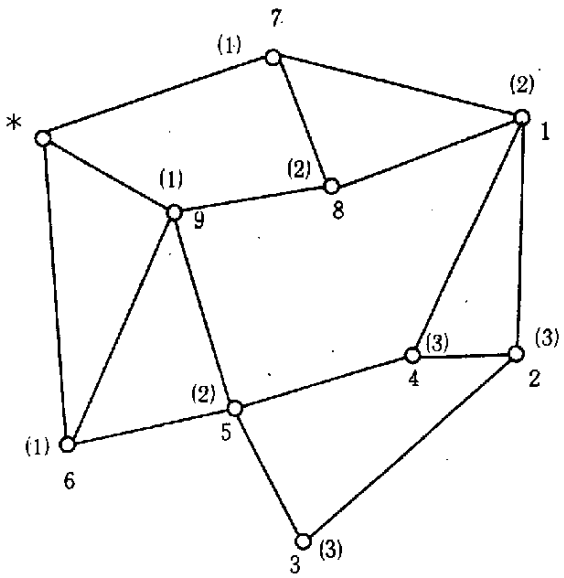


Fig 19

ホップ数は回線網の故障が起ったときに用いられるものである。ホップ数は，隣接ＩＭＰが目的地まで何ホップでいけるかを教えてきたとき，目的地が自分自身以外のときには，それに１加えることにより求めることができる。

あるノードへのこのホップ数が大きくなりすぎると，実際には全ノード数プラス１になるとそのノードへのルートが遮断されたことになる。それ以下の場合にはどんな最悪の場合でもネットワークの働きは維持されているということになる。このホップ数は現在インフォメーションの経路があることを示しているのであり，これは必ずしもその経路が最適のものだというわけではなくただ経路の存在を示すものにすぎないわけである。

ディレイ数はそのノードから他のノードに到達するのにどれ程の時間がかかるかという見当をつけるためのものである。各ノードに於いて，あらゆる瞬間のディレイがすべてのノードに対して測られているが，これは各ノードへのホップ数とその時点でのキューの長さの関数として計算される。そして各々の隣接したＩＭＰから伝送されたディレイ情報に更に隣との間のディレイと自己ノードにおけるキューにもとづくディレイをつけ加えて更新される。そして自己ＩＭＰに複数の隣接ノードがある場合は，その中で行先に到達するのに最小のディレイである方向に出力ラインを選択する。

オリジナルなルーティングの機構においていくつかの点を指摘することができる。

第一に，このノードの動作は他の部分で起っている事象とは独立したものであり，他のノードがどのようにデータと関係しているか等には全然関知しない。データが送られている伝送の途中で故障が回線に起った場合は，ノードはこのデータをその経路では送ってもらえなくなり，その時，ディレイは無限大であるということが告げられ，データは方向転換させられる。そして他の経路を見つけだす。

さてここでいつの時点をもってパケットが行先に到達したかどうかを決めることができるのかという疑問があるが，基本的にはタイムアウトを用いて状況を把握する方法がいくつかあるので問題はない。つまり，パケットが送られて最終のHOSTに入るとRFNM が送り返され，パケットの入ったときからタイマーが働いてRFNMが30秒以内に返って来るのを計測するがもしRFNMが返って来ない場合にHOSTに対して，"伝送が完結していない。何か異常が起った"ということを伝える。これは，パケットがループ内をさまよっている状態を仮定するわけで，実際にはヘビー・トラフィックの場合以外では考えられないことである。即ちこのことからやはり，この機構は，トラフィックレベルが非常に大量にならない限り，すぐれた機構であるということがわかったのである。そしてこの機構でトラフィックレベルが大きくなった時に伴う困難なこととというのは，600ミリ秒毎にデータを送るだけではネットワークの他の部分に関して十分に良好なインフォメーションを保持することはできないということである。云いかえれば，600ミリ秒の間に全ラインに送ることのできるのは30パケット分のインフォメーションであり，２番目のルーティングのトランスミッションで60パケット分が送られるのに1.2秒を要するわけである。通常，トラフィックの流れはル

ーティングから知ることのできる変化よりもさらに大きいものであった。そしてこのディレイの考えはローカル・ルーティングを行うためには非常に有用だったが，遠くにあるトラフィック構成の最適化を要するデータを扱うには非常に都合の悪いものであった。結局我々は，重要な事は，さらに合理的な方法でヘビートラフィックを取り扱うことができるようなネットワークをつくることであり，むしろ経路の取り方の是非を論ずるような特殊な場合を数多く設定することではないという考えに至った。

最も重要だと思われることは，ネットワークをラインの故障に対してさらに適応性の高いものにするということであろう。

ルーティング機構に於いて回線の故障が起った場合を考えてみよう。

回線が全部つながっていてうまく働いているときには各々のノードはルーティング・メッセージを625ミリ秒毎に交換しておりそのルーティング・メッセージの先頭の1つのビットが"Hello"といい一方で"I heard you"と答えている。

毎回ルーティング・メッセージ交換をするたびに一方から隣接IMPのノードに"Hello"というビットを送り，逆に"I heard you"というビットが送り返されている。ここでラインに故障が起きたとすると，このうち1つのIMPは"I heard you"という返答ができない状態になりその相手のIMPは"Hello"を云いつづけるが何の返答もない。こうして，ラインのスピードに依っていろいろだが，ある時間（5～10秒）が経過した後に，このラインのダウンがつげられ，復旧する迄はディレイが無限大であるということになり，別のルートに対してディレイが測定されるようになっていく。またホップについても他のルートに対して計算され，この故障したラインは再び復旧したという知らせがない限り使用されなくなる。トラフィックが再び通るようになっても，しばらくの間はそれが確かであるかどうかが確認されてはじめてその復旧が正式に行われる。

さてこのような回線故障があった場合にはそれが確認されるのに1/2秒，1秒或いは$1\frac{1}{2}$秒という時間がかかる。またダウンから復旧する際にもそれを完全に確認するために余分な時間がかかるので回線はかなり長い時間ダウンの状態におかれることになる。

問題は回線が故障だということを見つけて決定するまでどれ程の時間がかかるかということである。もしバッファー数が非常に少なければ，データの流れは非常に速い。

もし10か20或いは50，100のバッファーがあるとして回線が故障していると決定するまでどれ程待つかということである。たぶん全てのバッファーがフィルアップしてしまうであろうが，そう長い時間フィルアップさせておきたくない。というのはそのままにしておくと他のトラフィックの流れを妨害することになるからである。我々は回線の故障は，5或いは10秒程度で，故障の宣言をしているのである。つまり，米国では種々のキロビット回線のバーストの割合は一般にミリ秒のオーダーだからである。

30ミリ秒或いは50ミリ秒，100ミリ秒のバーストがあっても1秒或いは5秒もかかるバーストは殆んどない。

5秒のバーストの場合，実際には完全な故障と云ってよい。何時間もその状態が続くであろう。

ホップがラインの故障決定に使われたことを思いだして欲しい。ネットワークがその決定をするのにかかる時間は15秒以下である。実際にはその間に，正常なメカニズムが働きだしタイムアウトとなる。設定はノーマルなタイムアウトよりわずか長く取られる。

これまでにルーティング・ストラテジーに対する修正案を色々考えてきたが，そのうちのいくつかは既にネットワークに適用されている。これらの修正案は次の様な要因が基となって考え出されたものである。

第1に，フロー・コントロールは余り多量のバッファーを1つの回線キューに費すことが不可能であるため，もしいくつかの回線（経路）を有効に利用しようとするならば，各回線キューが小さくてもこれを実現できるような機構を考えなくてはいけない。

第2に，これは短期的観点からみてさらに大事な点であると思うが，ネットワークの状況表示は数秒以内に得たいものである。即ち，もし回線が故障で遮断されたとき，これを基本的にはミリ秒のオーダーで知りたい。或いはできればミリ秒の1／10のオーダーにしたいと思う。そうすれば1，2秒という中断された状態を現出させずにトラフィックにどこか他の経路を速かに適てることもできるであろう。

第3はテーブルストレージに関連したことである。現在，行われているルーティング・アルゴリズムではFig 18に示すようにメッセージの最後部にHOSTの状態を示すインフォメーションが送られてくるが，ネットワークが十分大きくなってくると，その情報自体が膨張しこれを送ることが不適当であるといえる。従って，その後のルーティングではこれは送られていない。その理由は，ネットワークが非常に大きくなった場合，HOSTの数は非常に多くなるから，HOSTの状態をネットワークが常に保っているよりもむしろ，そのHOST自体を使いたい時に，そのHOST の状態を決定するためにその都度テストをしたり，そのための機構を用意したりするということでよいからである。最初のバージョンでは完全にHOSTの状態はフォローされていた。即ちルーティング情報に含まれたHOST情報はネットワーク全体に伝えられ，そしてHOSTが故障で活動停止したかどうかの判定は，ルーティング情報の中で1ビットの0か1の表示をテーブルで確認することにより発信地でたゞちに知ることができるようになっていたわけである。

オリジナルのアルゴリズムに於て，これらのルーティング情報は隣接ＩＭＰから発送された時には必ず受けとられる。即ち，隣接した複数のＩＭＰから夫々625ミリ秒毎に情報が送られるが，それらは各自，非同期に送られ，そのＩＭＰは送られて来た情報を全て，その都度スキャンする。（1時的にまとめておいて，全部揃ってから総合的に調べるわけではない）そして，最少のホップ数およびディレイの経路が選ばれるが，その過程では2種類のテーブルが作成される。1つはル

ーティング・テーブルで，すべての他のノードへのその時点での最短経路を示し，もう1つは隣接ＩＭＰに送るためのルーティング情報である。

これらのルーティング機能は，全く信頼度よく動作する。つまり，ある場所でデータ伝送が止まることがあっても，そのために待たされることもないし，非常に緻密な同期が要求されるところでおこってくるような通常の問題にもわずらわされるようなことがない。

ルーティング・インフォメーションは非常に重要なものである。たとえばもしなにかの間違いでルーティング情報の中の数ビットを失いディレイが0或いは非常に低いルートが指示されたような場合これはずっと後まで影響することになる。そこでソフトウェアでルーティング・パケットの最後部に於いて，チェック・サムが行われる。

これはルーティング・メッセージをチェックするだけの特別チェック・サムというものである。パケット及びソフトウェアへの内部チェック・サムの導入についてお話するのはこれが最初であるが，実際これは非常に重要なものである。これは shift & rotate type のチェック・サムである。

これと別にまた非常に古いインフォメーションの部分が使われないようにすることも行われる。実際，タイムアウトはデッドラインのプロセデュアを経て起ってくる。即ちもし，これらのエントリーのうちの1つがルーティング・インフォメーションに対して " I heard you " という応答をしないようなラインに対応している場合，このエントリーはルーティングの計算を行う際には使用されない。即ち，この計算を行う際に使うデータは一応，新しいものでなければならない。

ノード数が非常に大きくなると，テーブルストレージの量は無制限に増え始める。そして我々がすでにルーティングに対して行ってきた変更の一つはこれらすべてのテーブルをネットワーク内に放置しておかないということであった。コア・ストレージを過剰に占有してしまうからであるがそれよりも，これらのテーブルがやってくるといつでもこれを別のキューに蓄えておいて，そのテーブルから必要なデータを計算し，実際のテーブルに入れる。即ち，ルーティング・インフォメーションを隣接ＩＭＰから得ると，その値を使って計算しテーブルの中に入れる。こうしてこのテーブルを放置しておかなくてもすむようになる。というのは一般的にこれらのテーブルは，到着するパケットに備えて使われるバッファーに於いて存在するもので，機械の中のテーブルストレージに存在するのではないということである。

旧来のアルゴリズムにおけるテーブルの大きさは縮小される予定であったがネットワークの拡大とともにいくらか大きくなった。しかしそんなに過剰に大きくなったわけではない。ルーティングのインフォメーションが入ってきてそのステータス（状態）の変化が検出されると，ルーティングを再度計算し次に送るが，このステータスのインフォメーションはメッセージがネットワーク内を送られるのと同じ程度の速度で送られ処理される。

ルーティングに関するもう一つの我々の試みは，ラインの速度の関数として各回線からのルーティングの時間を計測することである。ここでどんな問題があるかを考えてみよう。

ここに50キロビットの回線があり，ルーティング・メッセージが64 ワードの長さであると仮定しよう。64ワードの長さで全て1ワードが16 ビットであるとすると全部で1024ビットの長さになるわけである。したがって50キロビットの回線だと1ビットにつき20 マイクロ秒の割合でルーティング・メッセージが運ばれる。

新しい機構においても従来の場合と同様に625ミリ秒毎にルーティング・インフォメーションが送り出されるが重要な状態の変化，たとえば回線の故障とかIMPのダウンとかが丁度起ったときに，いつでも，このルーティング・インフォメーションが送り出される。そして内部状態の変化が計算されつつ，それが伝播されるが何も変ったことが起らない場合にはそのままの周期的繰返しをつづける。インフォメーションの利用の割合は$\frac{20}{625}$即ちおよそ3％である。即ち回線の帯域の3％がルーティング・インフォメーションを送るのに使われており，CPUの僅かな部分，およそ同程度のパーセンテージの部分がこのインフォメーションを計算するのに使用されている。

ここで50キロビットの回線の代りに9.6キロビットの回線を考えてルーティングを送ってみると，全体のルーティング・メッセージの長さは1024ビットで前と同じだが回線速度が1／5 になるのでメッセージの運ばれる速度は1ビットにつき20マイクロ秒の代りに約100マイクロ秒となる。そして50キロビットのラインでは3％だったが，9.6キロビットの回線ではインフォメーションの使用度は増えて$\frac{100}{625}$即ち全体のキャパシティの15％以上になる。そしてさらにいくらかのディレイがでてくることになる。つまり，ある時間をおいてパケットが到着するが通常は20ミリ秒のキューイング・ディレイだったのがこの場合にはそれが100ミリ秒のディレイとなるわけである。そこで回線速度を落す場合は，ルーティング・メッセージの大きさを小さくするのが最良であるが実際には完全なインフォメーションが欲しい場合にこれを行うことはできない。あるいはもう一つの方法としてルーティング・メッセージを送る回数を減らすということが考えられる。つまり完全なルーティング・インフォメーションを得るために，基本的にはルーティング・メッセージの伝送と伝送の間隔を調整することであり，50キロビット回線の時に比べて，5倍の間隔にするわけである。

同様にして，もしさらに回線速度を落して，たとえば，この速度では遅すぎると思われるのであまり使いたくないのであるが，4.8キロビットの回線を使用した場合には，これを10倍にするのである。現在（50キロビット回線で）行われているようにルーティングを1／2秒或いは6／10秒毎に1回送る代りにその10倍の6秒毎に1回という割合で送ることになる。実際にはトラフィックがもっと多いのでルーティング・インフォメーションをもっと数多く送っている。

ではどのようにして回線速度を決定することができるかというと，IMPについてはこれを知るための条件は何もつけられていない。そこでパケットの長さはわかっているので，これを送って送り始めた時を記録しておき，伝送の終了時も記録しておくと，テーブルにエントリーが読みとられてそのインタバルが測定されるというわけである。

異なるネットワークの結合とプロトコル

プロトコルの構造をFig 22に示す。すでに述べたようはHOSTとIMPの間にIMP-HOSTプロトコルがあり，HOSTとIMPのコミュニケーションをおこなっている。HOST-HOSTプロトコルはHOST同志がコミュニケーションするものである。

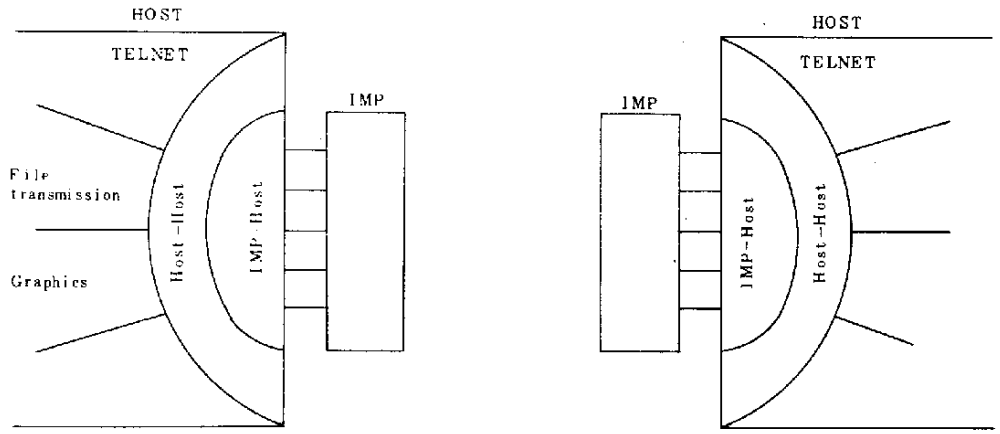


Fig 22

更にハイレベルなものとしてターミナル用のTELNETプロトコル，それに一般に多量のデータを非常に速く伝送する，スループットの高いファイル・トランスファー・プロトコル，またグラフィック・プロトコルといって映像用或いはファクシミリ用などに便利な線や図を送るのに使われるものがある。

システム設計に関するレベルに於ては，HOST-HOSTプロトコルをそのシステム設計と切り離すことは絶対にできないものである。それは，性能，効率に大きく関わっているものである。1つのパラメータ，或いは時間定数，変数の大きさ等をほんの僅かでも変化させると，急にシステムとして成り立たなくなるような場合があることは多く経験されてきたところである。そしてもう一度最初に戻って全体のプログラムを再設計しなおすことになるのである。従って我々はこれに対して非常に細心の注意を払って取りくんでいるわけだがこの理由の1つは，まだ完全にシステムとの分離ができるような仕様を作りだすことが可能になっていないということである。この完成に向って前進することは可能だと思うが，少くともこのプロトコルが完全な仕様のできる形に標準化されていないという現在の状態からは前進できるだろう。既に98%迄の仕様ができ上っておりあと残りの2%が未決定の部分である。たとえばHOSTはパケットを得る迄にどの程度の待ち時間

を要するか等が未決定のために，このプロトコルをシステムから完全に分離することができないのである。

異機種ネットワーク間で相互接続したときに起ってくるいろいろな問題を調べていくと，ここで起る問題が，HOSTの動作にそのまま影響を及ぼしているといえるが，特にパケットは小さなパケットに分割されなければならない。ここに直面する典型的な問題がある。

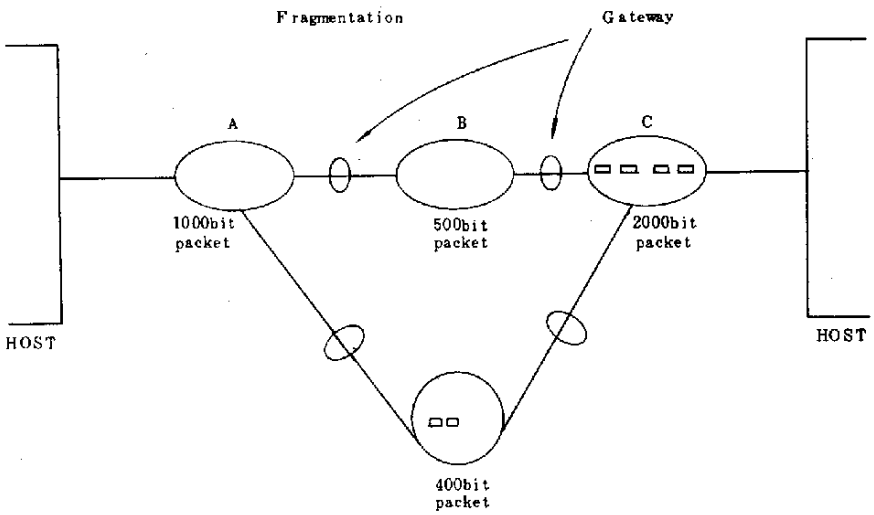


Fig 23

いまFig 23に示すような500ビット／パケット，1000ビット／パケット，それに2000ビット／パケットのネットワークA，B，Cがあるとしよう。このネットワークが接続されている場合，たとえばプロトコルがどちらも全く同じコンピュータ同士が丁度電線で直接結ばれているようになっていても，1000ビットのパケットがこの500ビット／パケットのネットワークに送られてくると500ビットのパケットに分割され，再び送り返す時には結合されて1000ビットにしなければならないということがあり，これはすなわち，リアセンブリであって多くの点で良くないのである。つまりこのリアセンブリに際しては再びパケットを集めて1つにするための時間がディレイとなってくる。多くのバッファリングを必要とし，うまくやらないとロックアップ状態になる場合がある。また，再伝送しなければならないときは，HOSTは，ともかく何らかのソーテング（分類）をすることができなければならないであろう。このように，ネットワークにリアセンブリを起させたくない理由がおおいにあるのである。

初期の頃は，ネットワークでリアセンブリを全部引き受けて，HOSTはただパケットを送ったり返したりするだけの機能で良いとしていたが，最近ではHOSTに リアセンブリの機能もいくらかもたせることができるところまできている。こうして500 ビットに分割された2つのパケット

のインフォメーションオーダーを乱すことなくこのHOSTで1つに結合する方法ができなければならない。こうして1つが米国のそしてもう1つが日本のネットワークであったとしても，各々のネットワークが互いに相手のネットワークの特有性を知る必要がないということになるが，これは，さらに複雑なものであろう。というのは，たとえば400ビットのネットワークに，あるパケットが入って来て3つの断片に分かれて，このうちの2つが取り入れられ，あとの1つが経路を違えて来た場合も同じようにリアセンブリが行われて送られるか，或いは個々の断片が別々に送られてHOSTにこれを判別するように命ずることもできるのである。

私が現在進めようと考えていることは，実際には，これら全部のパケットの断片をそのまま行先に流し送って，それらを1つに結合するかどうかは行先で個々に理解できる方法をとるという自由な選択に任せることができるようにしようということである。これはつまり，それぞれの中間段階での協力が必要となってくるわけであり，中間点のゲートウェイ（Gateway）に，パケットの断片化（フラグメンテーション）を許容する性質を持たせることが必要である。そして受信端のHOSTまで送られたパケットの断片が個々の意味をもって判読できるようでなければならない。

ARPAネットで現在使われているHOST−HOSTプロトコルの機能をいくつか述べてみたいと思う。HOST−HOSTプロトコルのオリジナルの設計についての参考文献には，1970年のSJCCの論文と1972年の同論文がある。

このプロトコルの設計は1967，68，69年頃のオペレーティングシステムにもとづいて為されたものであり，ネットワークで使われるパケット・スイッチングのやり方と少々ちがったアプローチを使っている。

そこでは，メッセージが1つの機械から他へと流れるようにするのではなく，むしろ，ただメッセージの前にアドレスを置くという仕方であった。

開発されたプロトコルはデータが流れる前に1つの機械中のプログラムともう1つの機械のプログラムとの接続が完了していることが必要とされた。今振り返って考えてみると，HOST・プロトコルに対し，さらに良い方法を採用することができたのであろうが，このプロトコル開発当時のオペレーティング・システム担当の人々が考え得る方法としては，それが唯一の方法だったのである。従って，開発当時の最初のプロトコルは一対のコンピュータ間の内部の点と点でコネクションを確立して対話を行うというやり方の一種の接続指向型プロトコルが使われた。

プロトコルは次のように働く，これをFig 24で示す。

まず左側のプロセスがNCPにあるメッセージ，たとえばサイト名を指定してCONNECT TO MIT と指定する。これは実際には一種の内部的なシステムコールであり，このような一般的なメッセージの形式をしている訳ではない。

ユーザは命令コード列のどこかである特別なインストラクションを実行してたとえば典型的な例としては"BRS42"というコードとあるデータでもってこのインストラクションが実行され，あ

るいは別の機械の場合は " JSYS " というような形で実行される。

プログラムがこのようなインストラクションを実行すると，EXECUTIVE（またはMONITOR，これはタイムシェアリングシステムであるのが多い）がこれを捕えて，NCPにCONNECT というコマンドを伝える。NCPがコマンドを受け取るとNCPはネットワークを通してこれを送り出してやる。これはパケットとして送られる。そして，右側のNCPは，左のHOSTに対するコネクションはオープンしている旨を伝える。これには識別番号がつけられており，これはソケットと呼ばれている。そしてNCPはどのソケットに接続すべきなのかを識別する処理などを行なう。 NCPはあるテーブルを持っており，それでソケットとプロセスとのマッピングをおこなう。プロセス 1，2，3，……，そしてソケット 1，2，3……というような対応関係である。

そして，パケットは左のNCPのセンダー・ソケットから送られ，右のレシーバ・ソケットで受取られることになる。

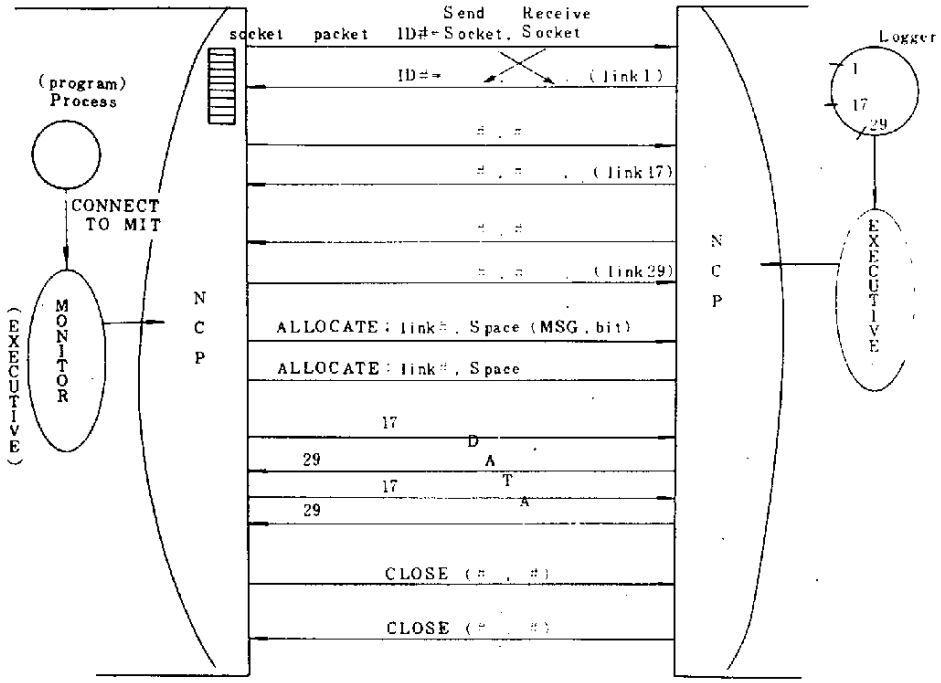


Fig 24

レシーバ・ソケットは常にロガー（ Logger ）プロセスにつながっている。

ソケットは両方向に対して存在し，すべて異なった番号がついており，センダー・ソケットには偶数の番号がつけられ，レシーバ・ソケットには奇数番号がつけられている。

これらについての詳細は Function - oriented protocols for the ARPA Computer Network 1972 SJCC の論文に述べられている。

NCPはソケットとソケットのペアをリンク番号によって示す事ができるようになっている。

右のＮＣＰは受，送の２つのソケット番号とリンク番号を送り返してくる。

これらのコマンドはすべて非常に特殊なリンクを使って行き来し，このリンクはコントロール・リンクと呼ばれ，特別の番号がつけられている。上記のリンクをリンク１とすると，その他行き来するメッセージはリンク１と異なった番号，たとえば17とか29とか他の番号のリンクが用いられる。

オリジナルのHOST・プロトコルにおいては，個々のリンクは一方通行のものを別々にはっていた。したがってフルデュプレックスの（ Full Duplex ）連結を得るには，行きのリンクともう１つの返ってくるリンクの両方をセットアップしなければならない。加えて，バッファースペースの割当てが行われる。メッセージに対してどれだけのバッファースペースが要るかということである。スペースはメッセージ数とビット数とで表わされている。

これらの情報は，すべてコントロール情報で，データは含まない。そしてリンク１を通して送られる。たとえば，このコントロール情報の内容は，リンク29は16000ビットで，４メッセージ分として割り当てられたというようなことである。これはビット数だけでもメッセージ数だけでも不十分で両方共必要である。というのはメッセージ数だけだと，メッセージの大きさがわからないし，16000ビットとだけ云えば,この16000ビットのメッセージに対して１つのバッファーを割りあてて，100ビットを送ったら全部のバッファーがこれに使われてしまうということになるかもしれず，そうなればそれ以上のメッセージに対してバッファーが割り当てられなくなるであろう。従っていくつのバッファーが供給されたか，このバッファーは何ビットを持つことができるのかということが必要である。

実際，オリジナルのプロトコルのこれらの用意されたリンクに対する働きといえばただ，一時的にこの割り当てられたバッファーに関する情報を送ったり，送り返したりすることだけである。

バッファーは指定されたリンクに対して使われるわけだが，そこでの目的は，これらが使われる予定の最終リンクに対してであり，使われる予定のスペース割り当てを知ることにある。実際の詳細については，前述の論文に書かれている。

HOST-HOSTプロトコルに於いてインフォメーションを表わすメッセージはFig 25に示すように１つのリンクにつき72ビット中56ビットが使われている。すべての処理が終了すると両方向にCLOSEのメッセージが送られ両プロセス間のコネクションは切断される。

このような管理方式で困った問題は，非常に大きなテーブルが通信の両端で要るということである。それはたとえ通信経路があまり使用されていない場合でも必要である。初期に我々の発見したことは，MCROSSのシステムの使用に際して起ったもので，その時，最初は多くの種々のメッセージがいろいろな場所へと送り出され，１つのプログラムによって全体のソケットテーブルがフィルアップされていた。従ってこのプログラムは一度に一つ以上のメッセージを送ることはできない状態であった。そこでそのテーブルに対して同時多重送信のできることが期待されていた。という

のは，たとえプログラムが1つでも，それはソケットを割り当てるのと同じだけのトラフィックをつくりだすことができるのである。

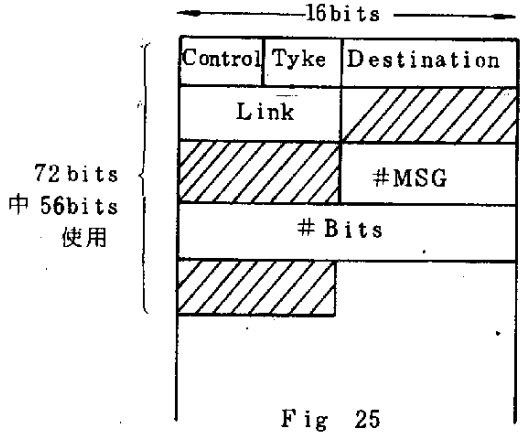


Fig 25

従ってWaldenがその論文でM－Sメッセージ交換プロトコルについて書いている理由も，こうしたやり方を特にやってみることにあった。その方法というのはHOSTがパケット交換モードにさらに関与することができるようにするというよりも，むしろ，テーブルの場所の固定というやり方であった。

Waldenのプロトコルに関しては少々困った問題があった。それは基本的には一度に1メッセージのプロトコルであった。送信側はメッセージを送りたい旨を告げ，受信側もメッセージの受信希望を告げるが，これら2つのコントロール・メッセージは，メッセージが実際に送られる前にどこか中間の場所で1つになる必要がある。一度に1メッセージしか送ることができず，第三の実際のメッセージを送る前にもう2つのコントロール・メッセージを送らなければならないのである。

この種のプロトコルを取り扱おうという全体の意図的なものは，このプロトコルを研究し，ネットワークを相互接続するといった問題 ── 今ではこの種のサーキットスイッチプロトコルのある限界点は乗り越えられたようである ── にするどく焦点をあてることにある。

実際，コントロール用に別の分離したリンクを使うという考えになると，完全にリンクを用いるという概念から離れることができる。即ち，各ＨＯＳＴが他のＨＯＳＴにメッセージを送るだけで，システムは丁度ＨＯＳＴの機械内部に，あるＩＭＰの機能を持ったかのような状態になるのである。

このプロトコルは基本的には次のようなものである。

パケットあるいは，メッセージが入ってくると，IMP−HOST プロトコルは，それをキューに登録する。

一方HOST−HOST プロトコルは，メッセージ用バッファイのアロケートなどの役割を果たす。

そして両HOST内のプログラム群は丁度チャネルで結ばれていると同じで，プロトコルはその間には存在しないと言える。各プログラム即ちプロセスは自分でバッファを用意してデータを要求すると，HOST-HOSTプロトコルは該当するパケットをさがして，そのプロセスからの要求に対応するデータを，そのプロセスのバッファに直接書き移してやることになる。

したがってフロー・コントロールはHOST間ではなくプロセス間のレベルで行われることになる。

各HOSTにおいては少量のバッファしかないため，HOSTはプロセスが取っていく前に少量のパケットのみを蓄積することができる。（2~3個位）そして丁度相当するものが入ってきたときにそのバッファーの相当位置を示すことができる。

ここで話しているプロトコルは中間（Inter - media）ネットワークの全体を通じての機能をもつもので，Fig 26の形式を持つ。

Inter Net Header

logical leader	S／D	Byte Count	SEQ		Text	check sum

Fig 26

先頭にインターネットワーク・ヘッダーというものがあり，これによって，送信側及び行先に対するインターネットのアドレッシング機構が識別される。これらの情報は両プロセス間で使われる。ネットワークを経てやって来るプロセスには全てシーケンシャル アイデンティファイア（S／D）がつけられる。そしてメッセージのバイト数とその開始位置，およびシーケンシアル番号がつけられる。ヘッダーは何ビットかのフラッグを持っている。またメッセージの最後には，ソフトウェアのチェックサムがある。

メッセージはこのネットワーク内を無事通過した後，他のネットワークとの接点のゲートウェイに入る。ここで2つのネットワークを各々500ビットパケット及び1000ビットパケットのネットワークとする。500ビットパケットのネットに1000ビットパケットメッセージが入って来た時これは2つのパケットに分割される。これはゲートウェイに於いて行われるわけだが，この場合，このゲートウェイによりネットワークは任意にパケットの大きさを変えることができ，少くともヘッダーの大きさ迄任意に小さいサイズのパケットにすることができる。これは，違った種類のネットワークを使う場合に融通性という点で極めて基本的なものである。またこれはネットワークに関して，その技術的要請がなされるのに応じかつ，続いて技術が発展していくことができるためにも必要なものである。

現在我々は500とか2500あるいは3000ビット／パケットのネットワークについて語ってい

るが，今から25年～50年もたてば100万ビットとか200万ビットのパケットについて平気で語っていることだろう。そしてその時のデータの伝送コストはほんとうに安いもので，非常に大量の伝送が行われるためパケットのサイズは巨大なものになっているだろう。今日のマス・メモリー，さらに最新のマス・メモリーについて見ればわかることであろう。

あるメモリーの1ユニットが小さく分割される場合を考えてみよう。今述べようとしている事はオペレーティング・システムの構造の中でやられている方法について述べている訳ではなく現在作製されている物理的なメモリ機器についてである。小さく分割されたものは"チャンク"とか"ページ"と呼ばれるが，ここで"10^7ビット"を持っている1ユニットを取って考えると，Fig 27に示すようにこれらのページの1つはこの小ユニットの各々に対して10^4，10^5，10^6ビット等を容易に持つことができる。従ってたとえば10^4ビットのものなら1000個の"ページ"で10^7ビットとなる。これがこれらのユニットの1つになるわけだが，このユニットが小さいか大きいかはどんな技術が使われるかに依っている。そしてこのようなメモリーからこの大きさのユニット即ち10^6ユニットのチャンクのデータを回収することになる。これができれば伝送にとって非常に自然な大きさになる。100万ビットというのは非常に良いのだが，これは今日では高価につきすぎて実際に使用することは考えられないものだが50～60年もすれば事情は変ってくるであろう。

衛星通信ではとくに，このような10^6ビット／パケット或いは非常に大きなサイズのパケットを送ることのできるネットワークが欲しいと思うことだろう。おそらくこれには多くの融通性が期待されていることであろう。プロトコルは，この種の融通性が入ってくるのを妨げてはならない。たしかに，あなた方がパケットサイズを変えることができないうちに，50ケ国間で国際協定を結ぶということはしたくないことと思われる。

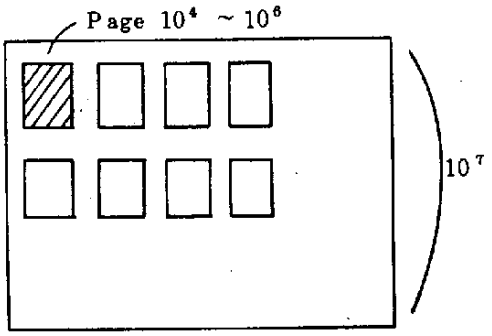


Fig 27

ネットワーク間のインターフェイスはできる限り簡単なものにしたいし，それがたとえばレベル変換機と同程度に簡単であれば特にのぞましいのであるが，同程度というわけにはいかない場合がある。2つの異ったネットワークに異った大きさをもたせておいて，その境界でのリアセンブリを

ネットワークに強制的に行わせることをなくすだけの複雑さは少くとも持っていなければならない。

リアセンブリは複雑な時間のかかる作業で，あらゆる種類の問題に関与してくるものである。

もし1つのネットワークで流れているパケットのサイズが小さすぎるという場合，入ってくるデータは多くの小さな断片に分割される。このネットワークに対する実質的な対策はネットワークに於けるパケットサイズを増大させるか或いは，他のネットワークから大きなパケットを分割して入れるか或いは又，このネットワークに衛星ネットを併行して使うかといったようなことが必要となるのである。

ある1つのプロトコルを持っているコンピュータが他のコンピュータと対話する場合には同じプロトコルを有すべきであるが，もし各々ちがったプロトコルP－1とP－2を持っている場合には第3のシステムとしてこのP－1，P－2の両方をもつものをインタプリタ（翻訳機）として使うことになる。

現在，異ったコンピュータ間でのデータ交換の標準方式に関してある種の合意に達することが困難な課題となっているが，もしこの国際規準を明確に定めることが不可能ならば，こうした方法を使わなければならなくなるだろう。しかし特にネットワーク間のゲートウェイについてはこれを適用するべきではない。

つまりゲートウェイ及び全ゆるPTTや各アドミニストレーション等に対して，ゲートウェイでの同じ標準方式のために1つのタイムシェアリングコンピュータの言語からもう1つ別のタイムシェアリングコンピュータの言語へと変換するためのオペレーティングシステムとかデザインについての知識を持つことを強いることはよくないと思う。

非常に大事な事だが，これらのネットワークはネット内の伝送において，どんな不適合なパケットも処理できなければならないし，特に潜在的な不適合としてパケットの大きさの不適合の処理が重要である。

また，ネットワークにはいくつかタイミング上の制限があるが，2つの異ったネットワークでそれぞれの待ち時間（通過時間）が，30秒と1／2秒という具合に違っている場合，なかなかうまく働かない。ネットワークから非常に特殊なメッセージが送られたり返されたりするとゲートウェイが全てを処理しなければならずこのときゲートウェイは実際には2つのネットワークに分割された状態になり，おそらく両者の間に回線制御方法とかエラーのコントロール法とかの標準化が必要になる。

しかしリアセンブリがゲートウェイの機能の1つであるということは実際には考えられない。さらに，ネットワークの相互接続には他の方法も考えられるであろう。

プロトコルについては一応この辺で終わり，Check sumについて述べてみる。私が言うCheck sumは，パケットに関する計算が行なわれ，ゲートウェイを通過した時にパケットが一応ばらばらにされ，それを目的HOSTに送るために，またすべて一諸にされるが，その時にCheck

sumが行なわれる。ゲートウェイでパケットをばらす時にはCheck sumはおこなわれず，ゲートウェイはCheck sumを確認しようとはしない。従ってパケットがゲートウェイを通過する時はいつもCheck sum情報をもって通過するのである。ゲートウェイを通過するパケットの順序に制限はなく，事実，reassemblyを必要とするなら基本的にはネットワークを通じて同じ通路をすべてのパケットが通るようにすればよい。再伝送に柔軟性を持たせる必要はないのである。しかし，各パケットが違った通路を通ることを許せば，到着の順序は不同になってしまう。一般にネットワーク内の通路はローカルなデータによりきめられる。

TELNETプロトコル2

次にTELNETのプロトコルのことについて簡単に述べてみよう。TELNETプロトコルはFig 28に示す機能をもちネットワークでターミナル機器のインターフエイスに用いられるプロトコルであり，エコーイング（echoing）のようなターミナルに関する機能を果たすわけである。エコーイングはTELNETプロトコルが演じる役割の中でも代表的なものである。

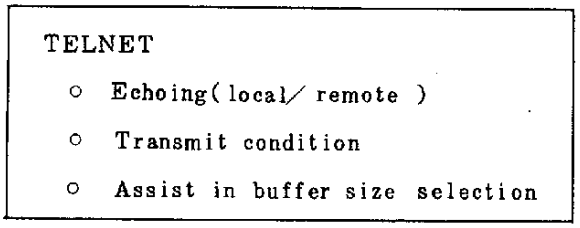


Fig 28

HOST 1に接続された端末からHOST 2を使用する場合，HOST 2のTELNETがHOST 1のTELNETにエコーを送っているわけであるが，ターミナルユーザのプログラムはすべてTELNETを通して交信される。直接，HOST−HOSTプロトコルを通じて交信されるのではない。TELNETが扱う処理の1つは"You echo""I echo"の問題である。ターミナルでのユーザーが「ローカルなエコーが欲しい」という場合，最初に協議をしておかなければならない。例えば，私が非常に遠く離れたターミナルにいて，目的HOSTのTELNETからのエコーを要求する場合，TELNETプログラムに，エコーをリモートにして欲しいと言うことができる。この場合TELNETはHOST-HOSTプロトコルを通じて私のTELNETに「エコーをしないように」というメッセージを送ってくるであろう。そしてそれに対し恐らく「私はエコーしません」というメッセージを戻すであろう。このようにエコーが行なえるのは，ローカルかリモートのどちらか一方でしかだめである。違ったシステムは違った方法で反応してくる。普通，ユーザーは好きなようにTELNETをセットできるわけである。例えばTENEXを使おうとした場合，TENEXが行なうということをTELNETに伝え，エコーがTENEXから来るようにセットされる。そしてユーザーがそのエコーは気にくわないので，ローカルのエコーにして欲しいと言える。この場合TELNETは，TENEXに変えるように伝えるのである。二番目に行なえることは，伝送状態を明確にすることである。例えばTELNETに対し，「パケットで各キャラクターを送って下さい」とか「伝送前にcarriage return keyが得られるまで待ってほしい」とか，あるいは，「16キャラター毎に，伝送して下さい」という具合に。TIPでは，バッファースペースの量は1ターミナルにつきほぐ16キャラクターのオーダである。例えばそれ以上のキャラクターを送るようにとは言えるが，あまりバッファのスペースがないので恐らく15〜16キャラクター単位で送ってくるのが普通であ

る。しかし，高速装置の場合はもっとバッファースペースが多く例えば100キャラクターぐらいは必要であるし，正常な流れを維持する為にも，より多くのバッファーが必要である。

第3番目にターミナルで行なえることはTELNETのプロトコルを使って，バッファーサイズの選択が出来ることである。TELNETで使用できるターミナルの種類は多いが，例えばスクリーンのあるデスプレイターミナルがあって，disc上にあるfileをとりだし，それをターミナル上に表示するとする。そのfileが100ページもの長さのものであっても良い。そしてどのTELNETであろうとお互いにそのターミナルの種類をまず確認する。そして一方のTELNETは必らず，スクリーンに合うユニットを送り返してくれる。これに対しHOST−HOST プロトコルはそれに応じたデーター量の伝送をおこない，スクリーンを満たす程のデーターを得ることができるのである。そして例えばcarriage returnのキーを打てば，次の一画面分のデーターが入ってきて，スクリーンを一杯に満たす。

以上がTELNETの役割であり，いかなるターミナルであろうと，それを扱う必要条件を満たすものである。

ここでパケットで起こる面白い問題をあげてみよう。例えば，あるプログラムの実行を開始し最後までランするようにとの指令を与えたが，何かの原因で動きがにぶくなりループをおこしてしまった。あるいはプログラムに多量のデーターを送ったところバッファーにたまった状態になって途中で止まってしまったとする。さて，これは間違ったものを送り込んだのではないかと思われるが，実はそうではない。というのは，このケースでは実はデーターが全部プログラムに入ってしまう迄には約30分という長い時間がかかるのであり，データーの伝送そのものはやめるわけにはいかない。こういう場合，これを防ぐ一つの方法がある。すなわち，TELNETを通じて，breakしろというHOST−HOSTプロトコルのコマンドを発することができる。HOST−HOSTプロトコルはデーターの流れに，中止サイン（interrupt signal）のようなものを送るのである。すなわちパイプラインには非常に多くのデータビットがたまっているが，TELNETはデータ ストリームの中の適当な場所にマーカー（Marker）をおき，先方のプログラムにデータをフィード（feed）しないようにと伝えるのである。受信側TELNETでは入ってくるデータ全てを捨ててしまう一般的な方法の代わりに上記のマーカーのくるまで，出来るだけ早く捨て去ってしまうのである。そしてパイプラインはフリーになり元の状態にもどる。これは，ターミナルコントロールのもとで多量のデーターをセンターからセンターへ伝送する際に非常に大切なオペレーティングシステムの設計上の問題である。プログラムのプロセスが大量データの伝送が終了するまで待たなければならないような場合にそれを1時的に中断するようにとの指令をだせる特徴は他のシステムにはないであろう。殆んどのコンピューターシステムではブレイクのシグナルは送れるが，これは単にチャンネルを通じてインタラプトの合図を得るためだけのものである。実際，Hi−bandシグナリズムには上記の機能が必要でありHOSTプロトコルもこれに相当するものを必要としている。しかし，実際にはTELNETまたはその種のプログラムによってinitiate（起動）されるものである。

さて，プロトコルの説明を終える前に，強調しておきたいことがある。すなわち，ネットワークシステムの総課題の中で一番重要なのがこのプロトコルであるということである。良いシステムを開発する為には，すぐれたプロトコルを得る程，重要な課題はないのである。コミュニケーションキャリアが長い間にわたって採用してきたデーターコミュニケーションサービスを供与する標準的な方法は今まで常に非常に明確なインターフエイスをハードウェアレベルで表わしてきた。そしてユーザープログラムに対する処理はすべてユーザー自身に任せたのである。データーネットワークにおいてネットワークへのインターフエイスはメッセージのお互いの交信がうまく出来るようなハードウェアとソフトウェアのインターフエイスが得られればよいと考えていた。

しかし，今では互いによくマッチしたすぐれたプロトコルが存在することが非常に大切なことであることがわかった。ネットワークを建設するどんなグループであろうとネットワーク上では協議を行ないあるいは忠告を行なったりすることが非常に重要であり，かつプロトコルの問題に親しまなければならない。しかし実際のところ，どんな問題が残されているか，どのようにして共同作業するかの定義はないのである。

新しいＩＭＰハードウェア

次にネットワークのハードウェアの面についてまとめてみよう。

我々が最初に設けたＩＭＰは516のミニ・コンピューターであった。ARPA ネットワークでの最初の15～16のNodeは，全部ハネウエル516のNodeであった。これらは12Kから16K まで上げることのできるマイクロ秒マシンである。そしてその後同じハネウエルマシンの316型にとってかえられた。2～3の例外はあったがプログラムはほゞ100％互換性があった。例外というのは，Watch dog timerがあるかないかということや，DMACチャンネルと DMCチャンネルを使う場合，少しではあるが変化がみられる。しかし，516型と316型の両方とも現在では，同一のプログラムで操作されている。チャンネルに関係する部分では，516型か316型かの区別をテストしそれに応じてあるinstruction 群をとばしたり実行したりという操作は若干ある。

さて，様々な理由で，我々はＩＭＰの新らしいシリーズの開発を始めてきた。1969年以来，ＩＭＰのソフトウェアの開発を行なってきたが現在のところかなり安定しており毎週火曜の朝に少しづつ新しい改良がおこなわれる。ルーティングの大きな変更でもない限り現在のＩＭＰではさほど大きな発展はもう望めない。我々は次のようなパラメーターを規定している。回線速度は250 kb／sが最大限であり，全throughputは恐らく1Mb／sそして98％の信頼性を確保するということである。ネットワークにおける本質的な要求である高い信頼性と，より高速の回線を扱うことができるように，我々は，Lockheed Sueをベースとしたの新シリーズを打ち出したのである。このマシンは現在開発中であり，1975年には導入し運転できるであろう。従って，今，詳細にわたって技術面を述べられないし，また殆んどの部分がまだ決められていないし，プロセッサのデザインも多分かなり変化するであろう。しかし第1にLockheed Sue は小型のプロセッサであり，Fig 29に示すようにカード状をしている。価格は一つ1,000ドル弱ぐらいで全くのマイクロプロセサーではないが小さいことにはかわりはない。これはUNIBUSとも呼ばれる。

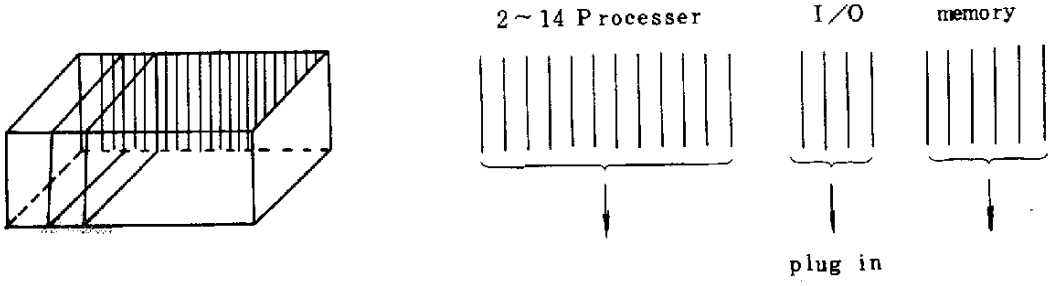


Fig 29　Lockheed Sue

UNIBUSのbussing構成により，多数のプロセッサーを小さなインターフェイスで，単一ラックに格納することができ，かつ，すべてのポイントは裏側で連結することができる。その場合カード状のプロセッサーを使用するのであるが最低2つ，最高14のプロセッサーを持つことができるようになっている。その数は，コンフィギュレーションによって違う。

これらの効果は次のように言うことができる。IMPのプログラムは数多くの小さなタスクに分けられ，各タスクは恐らく300マイクロ秒あるいはそれ以下の長さのものである。例えばMODEM TO IMPルーチンもタスクの一つになるものである。そしてキューに登録するのも別のタスクであり，とり出すのも別のタスクである。ルーティングの計算もまた別のタスクである。これらのプロセッサーはすべて個別に作用するものであり，プロセッサーには中央コントロールもなければ，エグゼクティブもなく，またスーパーバイザーもない。しかし，ハードウェアは，次のタスクが何であるかというトラックを保持し，Lockheed Sueの命令によりそれを読みとる。それによって次のタスクがロートされる。1マシンサイクルで，次のタスクが何であるかを知ることが出来る。

遂行すべきタスクのリストがシステムの中にプライオリティ順につくられており，新しく登録するタスクが最初に実行すべき時はそのタスクに高い優先権をもたせる。そうすれば新しいタスクが必要になっても，キューラインの最初に置かれることになる。そして，ハードウェアで再び読み込むだけで，プロセッサーが自由になった時，次に実行すべきタスクが告げられる。ハードウェアの中では，他のプロセッサーがタスクを処理しているのと平行して次のタスクを得るのに1サイクルかかるであろう。そして一つのタスクの処理を終えると，次のタスクを行なうためのコードをどこへ持ってゆくかを指適するだけでよい。仮に待機しているタスクがない場合，プロセッサーは，次のタスクが何であるかを求めつづける。そして一つのタスクが選ばれると，そのタスクはロードされ実行される。

すぐれた特徴としては，あるプロセッサーの状態が悪く，作動が効果的でなければ，そのプロセッサーを取り除くことができるということである。例えば，プロセッサーを14から13にしてもマシンの能力はいくらか低くなるであろうが，作動することはできる。各プロセッサーには他のプロセッサーがうまく作用していない場合，その状況を告げる手段がある。そして悪いプロセッサーをシステムから閉め出してしまい，連結されているテレタイプが例えばプロセッサー13及び6の状態が悪いということを告げるように，合図を送るのである。そうすると，その13と6のプロセッサーを使わず，残った他の12のプロセッサーだけで作用し始めるのである。作動状態が悪くなったプロセッサー13と6を修理の為にシステムから取り除し，修理を行ない，システムに再び戻し，プロセッサー13及び6を修理したということを告げる。ところが，恐らくまだ正常な作動が行なえるかわからないので，その状態をしばらくチェックしなければならない。チェックを続けて，良くなったとなると元に戻すわけである。この種の機械は非常に有益であるということがわかっていただけるであろう。我々の考えでは，このマシンの信頼性は単一マシンの信頼性よりかなり高いということが言える。

すべてduplicationであるし,かつ，システム内で故障を防ぐことに注意を払うことができかつ与えられたタスクを処理するのにプロセッサーの各々を個別に操作できるからである。ということは，チャンネル上のインプットを扱う時に複数のプロセッサーにより広帯域の処理が得られるということである。これによれば1.5Mbのフルタイムジョブが可能である。この方法であれば，ハネウエル516や316シリーズでできないような回線へ出入りするトラフィックの処理が可能になるという保証がある。第一のバージョンでFig 30に示すような2ないし3つの1.544Mbの回線あるいはHOSTへの1.544Mb付きの1.544Mbの回線を我々は考えた。また少々機能は低下するが4つの1.544Mbの回線も考えることができる。しかし，全バンド幅が設計値の5Mb以上になってしまうので，その可能性は確実ではない。恐らく最初は3つぐらいが適当であり，それに対して初期の応用を考えれば，同一のページングの環境でマシン全てを操作することができるようになるであろう。

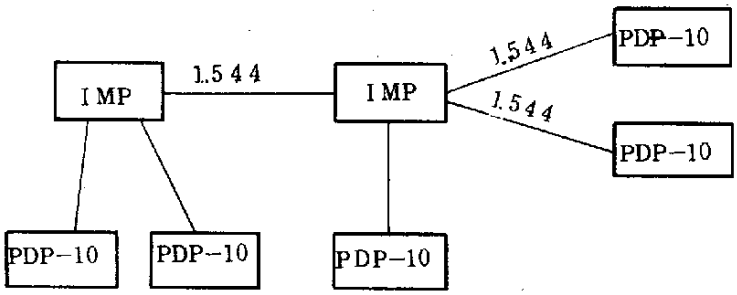


Fig 30

そしてバーチャルメモリーマシンをとり入れることができ，またFig 30の様にPDP－10を3台入れ，3台目のPDP－10を他のPDP－10，2台のスイッチとして使うわけである。そうすれば，マシン間でページングを行なうことができ，かつ，マルチタスクの実験を行なうことが可能である。これは我々が行なう，初期の実験の型を示しており，この場合ノードのスイッチポイントとして高速IMPを使うことになっている。現在のところ，まだ十分なトラフィックがないので，グラウンドラインに連結して使うことはできない。実際のところグラウンドラインは非常に高い費用が必要であるので，我々としては通信衛星チャンネルが安くなった時に，それに連結して使おうとしているのである。

ＮＩＣとＮＣＣ

ネットワーク・インフォメーション・センター（ＮＩＣ）はスタンフォード研究所（ＳＲＩ）にあるロジカルな名称であるが，そこはネットワークに関するドキュメントの博物館のような役割りを果しているところである。そこではインフォメーションが集められ，そこに送られた情報は蓄積され，分類され，多くはハードコピーに，一部はオンラインの状態にされる。ハードコピーの書類は保存されるが，それは２つの形式，即ち，プライベート用とパブリック用に分けられる。オンライン用のインフォメーションは，ＮＬＳと呼ばれるＳＲＩ開発のシステムに保存される。ＮＬＳはいろいろな編集とデバッキングのできるシステムで，このシステムはＳＲＩに代って TYMSHARE が運転しているものである。このシステムはTYMSHAREに収容されているが TYMSHARE のネットワークの一部分となっているのではなく，これはＡＲＰＡネットワークの一部である。TYMSHARE ネットとは接続されていない。ＮＬＳシステムはＮＩＣにあるいくつかのオンラインのドキュメントを持っている。それは，インフォメーションを作成せる責任を持っているわけではなく，ただ送られてくるインフォメーションを受け取り，もし外へ送り出してほしいと云えばそれを送り出すというようにちょうどクリアリングハウスのような働きをしている。

ネットワーク・コントロール・センター（ＮＣＣ）は実際にはコントロールそのものは何もやっておらず，そこではただネットワークに関するデータを集めているだけである。実際のコントロールは先に述べたアルゴリズムでおこなっていると言える。ネットワーク・コントロール・センターは実際にはネットワークの１つのHOSTであり，またネットワークを通してそこから種々のデータが出力される。

ＢＢＮのノードがネットワークの他のノードと接続されているとしよう。ＢＢＮのHOSTの１つはミニ・コンピュータ Honeywell 316である。１分間に１度，ネットワークの各ＩＭＰからこのHOSTにむけて１メッセージずつ送られてくる。各メッセージは各々のＩＭＰでその１分間に起ったデータを収集したものである。即ち，１分間の全トラフィック量，そのラインステータス等を算出してこれらのインフォメーションをパケットの中に組み込んで，毎分１回そのパケットをＢＢＮのHOSTへと送ってくるのである。ＢＢＮのHOSTは送られたパケットを全て，収集し接続されているいくつかのテレタイプで夫々のサマリーデータのプリントを行う。あるテレタイプではトラフィックのサマリーを，他のテレタイプではステータス情報をというように，プリントされる。サマリーデータは１時間毎にプリントされる。ステータス・インフォメーションについては，回線がダウンになったりアップになったりするとその都度，プリントが行われる。たとえば，ＩＭＰがダウンであることを知る唯一の方法は，ＩＭＰからのこのようなレポートを全然受けとらないことによってそのダウンを知るということである。従ってもし３分間のダウンがあってこの小さなメッ

セージが1つもBBNの316に到着しない場合は，そのIMPは何か故障があってうまく機能していないということを知ることになる。それで，これをテレタイプにつなぐとテレタイプからたとえばある時刻8時42分に回線7がダウン，8：47，回線7アップ，8：53 HOST 4ダウン，8：57，HOST 4アップ等の情報がでてくる。このように回線のステータス情報が与えられる。一方，サマリデータは，先刻の時間帯に於ける全HOSTからの全トラフィック或いは回線上のスループットをプリントするものである。

さらに我々は実際のコントロール手順がどんなものか，或いはネットワークのディストリビュート・マネージメントの問題等について考えるために，多重NCCに関する実験を開始しようかという段階に来ている。

NCCに集ってくるデータは一方通行であり，逆にコントロールセンターからネットワークへ出てゆくという情報は何もない。

BBNの316とネットワーク間のプラッグを引離した場合でもネットワークはなお完全にその機能をよく果し続けるだろう。しかしその時，ネットワークで起った事について記録が全然とられていないということであって，この316はちょうど会計機あるいは事務機のようなものである。

BBNには，このサマリについて研究をしている人々がいる。またBBNには別の機械PDP－10がある。316は，メモリーをたくさん持っていないので，トラフィックのサマリ或いはステータス・インフォメーションを計算する毎に，これらのインフォメーションはPDP－10に送り込まれ，PDP－10 オペレーティング・システムの大きなファイルに書き込まれる。このようにPDP－10システムを使って処理する方が都合が良いのである。

NCCでは前述のものの他にもっと色々な情報をとらえている。よりローカルな問題，たとえばユーザープログラムに関し誰が何の目的でネットワークを使っているかとか，あるHOSTがonであるとかoffであるとかどこの回線が故障であるかというようなものである。例えば1つの回線が故障であるということが知らされると，そのインフォメーションはテレタイプで印刷される。そしてNCCは電話会社を呼び出しその修理を依頼して電話会社は多分その回線を修理するだろう。しかしその修理の必要を起させる故障の原因となったネットワーク内部の問題の監視の責任は彼ら（NCC）が取るということであろう。

初期の頃は電話会社の彼ら自身の回線に対するテスト能力よりも，もっと多くのテスト能力を我々が持っていたことは明らかであった。そして我々は，回線について非常に多くのテストを行ってみたのだが，実際に，たとえば1つの回線につき859／100,000の誤り率が見られたこともある。これは，その回線の性能を示すものである。これがあまりにも悪すぎる場合は電話会社を呼び出してそのことを知らせるのである。ところが知っての通り，彼らは非常に疑い深い。回線が正常か異常かをどのように彼らに知らせることができるか，非常に難しいことであった。実際，彼らは我々の要請に対する処理の手順を持っていない。この回線の敷設されているのはユタ州とカリフォルニ

ア州であり，この回線を使用する権利を有する者はユタ州或いはカリフォルニア州の人間に限られていた。そしてこちら側の東海岸の者（我々）が西海岸の回線について呼び出していたわけであるから，彼らは困惑してしまい，我々の言うことがいつでも正しかったということを理解するまで大変だった。我々はこの回線には問題があると云っても，彼らは問題というが何が問題なのかと云う。しかし，これはすべて非常に早急な解決が要求されるものである。

現在まで我々の扱ってきた問題には，ネットワークの信頼性を非常に高く持続していくという問題がある。つまり，回線故障がしばしば起っても，我々はその両端部迄も調べるわけにはいかない。こんなときに週末で休業ということがある。最近我々は，サービス内容に依って，昼か夜か休日かといったことによらずいつでも調べることができるように，ネットワークの全部の箇所で1日24時間のアクセスが我々に提供されるよう依頼したのである。そして少くとも昼間のあいだは，すぐコンタクトできるようにだれか責任者を置いてもらうように依頼したが，あまり頻繁に使うことは期待していない。

RSEXEC

ネットワークに関する成果のなかで最近のものについて簡単にふれてみたい。

その1つとして，ＢＢＮが行ってきたリソースシェアリング エギュゼクティブがあげられる。これは，ネットワークに使われている大量のコンピュータが，ちょうど全国に分散した単一のコンピュータシステムであるかのようにする試みである。

御覧になった映画では非常に簡単にしか説明されていなかったが，これには長い過程があったのである。

ユーザーがＴＩＰのターミナルからLoginし，このネットを使用し，ファイルにデータを書き込んだりプログラムを書いたりランすることができるが，その際，どのコンピュータを使うかを特に指定する必要はない。我々が初期に使ったのは全てＰＤＰ－10であったが，FORTRANのプログラムなどであればＩＢＭやILLIAC Ⅳ あるいは他のマシンにも拡張できるであろう。

ＴＩＰはユーザーがスタートするのに適当な機械を選ぶ。各々の機械からユーザーに対して"幾らのコストでサービスをする"旨の申し出があり，ＴＩＰはそれらの答が返ってくる迄少々の間待機して，それから最も安い機械を選定する。

ユーザーはそれがどの機械であるかを知る必要はない。それからユーザーは続けてファイルに書き込む作業を指令するのであるが，このときはただファイル名を指名するだけでシステムはそれに対応するコンピュータがどれであるかを見つけ出してくれる。ユーザーがジョブを他の機械にスイッチする為の特別の対策をたてなくても，あらゆる機械に対して完全なアクセスが実現される。

これが大部分の機能であるが，とくにRSEXECに関するものをもう少し述べる。

まず，コマンド言語がどんなものかについて話してみよう。Fig 31に示すようにまず@はTENEXのHelloである。

```
@ RSEXEC. SAV ; 16          RSEXEC.MAC ; 3
- WHO MIT (CR)
- TYPE〔BBN〕NAME
- WHERE JONES
    TYY7, JOB14, STANFORD
- ADVISE JONES
-- BIND LPT USC-TIP PORT1
- LIST FILE (CR)
```

Fig 31

次に＂ＲＳ＇をインプットするとTENEXはエスケープメカニズム（ escape mechanism ）を持っていてコマンドを解読する。即ち，ＲＳとタイプしＥＳＣ（ Escape ）キーを打つと，RSEXECというようにタイプアウトされる。これはこのプログラムの名前である。ＳＡＶ：16は実行可能（ Executable ）のセーブ・バージョンを示し，それはたぶんバージョン16といったような番号だったと思う。こうしてＲＳをタイプし，ＥＳＣキーを打つと，ファイルネームの残りを完全にフィルアップしてしまう。

ＳＡＶの意味はエクセキュータブルな形になったバージョンということで，別にRSEXEC.MAC;3という形もあるが，これはマクロ・コンパイラーのバージョン３であり記号形式のソースプログラムのバージョンである。

いづれにせよRSEXECの実行に入ると，日付とかある応答が為される。そしてバー（ー）が出力されてプログラムはユーザーからのコマンドが与えられるのを待っていることを表示する。これにたとえば＂ＭＩＴには誰がいるか＂即ち＂WHO MIT (CR)＂といった指令を出すとＭＩＴではMULTICSやMATHLABなどのが使われているが，そのＭＩＴのマシンを使っている人の名前のリストがプリントされる。

あるいはファイルの出力を例えばTYPE[BBN]NAMEと指令すると，これはこのNAMEのファイルが例えば今，南カリフォルニア大学のマシンに入っているとしてもそのファイルが全部ＢＢＮのマシンにタイプアウトされてくる。

或いは＂Jonesはどこにいるか＂というような問いを入れると，ネットワークの全ての機械を探し尽くしてどこかのシステムに記録されたJonesという名はないかどうかが調べられる。そしてもし名前が見つかれば，例えばテレタイプ7，ジョブ14,スタンフォード……等という具合に出てくる。

さらに＂advise Jones＂というのはあなた自身のプログラムと彼のプログラムとを連結して１つのものとし，両者が夫々プログラムを書いたりディバッグしたり，同一プログラムを両方でコントロールできるようにする。

また＂BIND LPT USC-TIP Port 1＂はラインプリンタ（ＬＰＴ）をＵＳＣ　ＴＩＰに結合せよ＂という指令である。Port 1は別の名称である。これによりファイルのリストをＴＩＰのターミナルからローカル・プリンタに出力させることができる。

装置の構成は非常にフレキシブルに組むことができる。そのためのコマンドさえ与えればその指令に合ったコンピュータシステムの再構成が行われるであろう。

ARPAで我々が使っている装置はＴＩＰ，マルチターミナル，それにＩＭＰに接続されたＰＤＰ－10と，それにラインプリンタがある。これらは階はちがうが同じ建物内に備えつけられている。ラインプリンタにリストを行う時には階上へ上がれば全部のアウトプットが得られるのである。来

月にはゼロックス・グラフィックス・プリンター"ＸＧＴ"が設置できる手筈になっているが，我々の同じ階で接続されるからもっと速くデータを入手することができるようになるだろう。これは実際にはHOSTとして使用されるものである。

RSEXECのインプリメンテーションは，各HOSTに小さなプログラムを持つことによって行われる。これはＲＳサーバー（ Server ）とよばれる。各HOSTは勿論，自分自身のＮＣＰも持っている。各HOSTのＲＳサーバーはすべて同一プログラムで全て同じプロトコルに従うものである。

これらのＲＳサーバーは互いに通信し合っており，ネットワークでの各システムの関連するロードに対する対策をたてることができる。

例えば，ＩＳＩは 14.6 Jobs，ＢＢＮは 17.2 Jobs，ＣＣＡは 6.1，ＳＲＩは 8.3……等のロードを夫々持つという具合である。

ＲＳサーバーは他のＲＳサーバーからのメッセージやデータを，ローカル・オペレーティング・システムが使用するフォーマットへと変換する。

我々は，さらに高速のリンクを得ればすぐにページングに関する実験も開始することにしている。

このようなシステムの目的は，種々の機械を使う際に，各々の機械についての知識がなくても，それを使うことができるように全部の機械を１つに結合させるという技術を開発することにある。これはおそらくそんなに早急にはできないだろうが，たとえばユーザーは，"↑Ａ"が" delete character "ということを表わし，"↑Ｗ"が" delete a word "を表わし，"↑Ｑ"が" delete a line "，"↑Ｘ"が" restart "を表わすという具合にしてこれらを用いることができる。

システムによっては抹消がバックスラッシュであったり，また別の記号であったりするが↑の形で入力されたものをそれぞれのシステムが，自己の内部記号に変換する。

RSEXECにおけるパスワードは次のようにして符号化される。パスワードを送り込むと，あるアルゴリズムにより変換されもとのパスワードよりも長さのはるかに長いビットのストリングがつくられ，これがストアされる。これは以後，妥当なユーザーであることを確認するために毎度チェックされる。

他人は誰もそれを解読できない。たとえ，システムプログラマーであっても，そのアルゴリズムがわからないためにそれを使うことはできない。

そのストアされるものをＹ，パスワードをＸとするとＡ・Ｘ＝Ｙの関係がありＸの値を得るには $X = A^{-} \cdot Y$ を計算して求めなければならない。システムの開始にはＸの値の提供が必要である。ＸからＹへのインプリメンテーションは非常に簡単で選択可能な多数のアルゴリズムが存在する。他の方法でのインプリメンテーションは事実上不可能である。この方法によって多くのプロテクションが行われている。

RSEXECは現在では6～7台程度の機械の間で使われているだけであるが，依頼に応じて各場所のユーザーのディレクトリを調べてくれる。ユーザーはそのプロフィル，即ち名前あるいは，ディレクトリー等を指定するが，たとえばＳＲＩでKahnという名を指定し,またＢＢＮでもKahnを指定する。そしてこの両方の場所にファイルを保存する。後にそのファイルが依頼されると，これらの場所が全て調べられ，その名前のついたファイルを見つけだす。

これが非常に大きくなると，もう少し経済的な別のシステムがたぶん欲しくなると思うが現在のところ簡便さからいってこうした方法で処理されているのである。

衛星通信

これから述べることは，ネットワーク技術の衛星通信への拡張に関する事柄についてである。これはARPAネットワークの部分として述べるのではなくARPAにやがて関係してでてくる問題として述べる。既に周知の如く，我々が今までネットワークに関して述べてきた受信（インタラクション）というのは，一つのポイント・ツウー・ポイントへの回線を利用したものであった。そして，カリフォルニアとハワイの回線と米国東岸とノルウェーを結ぶ回線を除いては，他の回線はすべてグラウンドラインで結ばれているものである。従って現在我々が考えていることは，通信衛星チャンネルを使用して，ポイントからポイントへの回線を使って得られる効果より，一層高い効果をもつ方法を生みだすことである。この技術は現在我々は通信衛星を持っているので特に関心がある。又，国内の通信衛星にかかる費用も非常に安くすむことになりそうなのである。我々の見積ったところによると1.5Mbの回線を米国国内で通信衛星にした場合，必要となる費用は，voice grade回線の現状価格以下ですむのである。仮にvoice grade回線を50 kbとし，それを30回線使用するより費用面で安くなるのである。従って高バンド幅の回線に変えるのに充分理由がある。その正確な費用は回線上で実際にやってみないとわからないが，計画としては，単一広域バンドの回線を使い，米国全土で交信できるように考えている。すなわち，多角アンテナを通じてダイナミックな方法で単一チャンネルを共有することであり，本質的には，ポイント回線を多く得るようにすることである。どのようなものであるかFig 32を見ていただきたい。

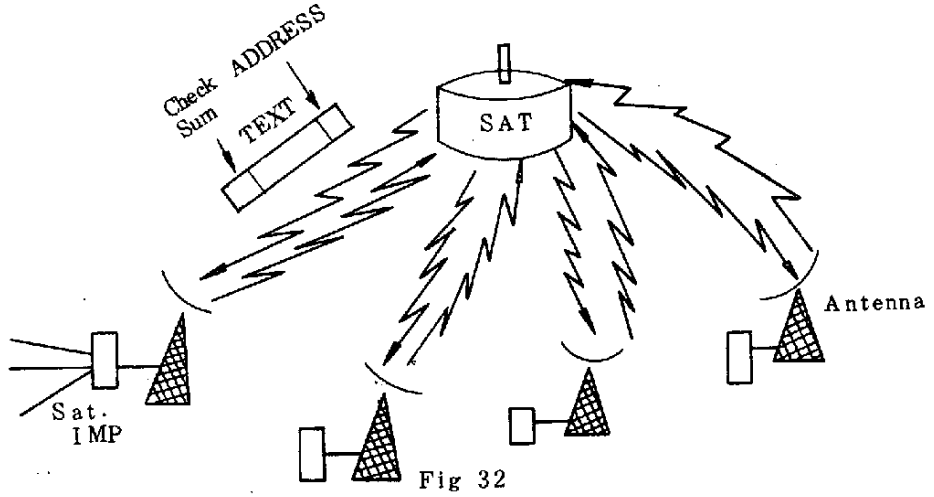


Fig 32

Fig 32は，一つの通信衛星を中心とし，4つのアンテナがそれぞれ発信・受信を行なっている。各アンテナにはサテライトIMP（Satellite IMP）が付けてあり，このサテライトIMPというのは，ネットワーク上の他のどのIMPとも連絡することができ，かつ，地上ステーションを制

御できるＩＭＰである。従ってパケットを通信衛星に送り，また送り返すこともできる。別に普通の場合と変わりはないが，ただ，我々としては，すべてのアンテナが同じ通信衛星チャンネルに連結できるようにしたいということである。従って，各アンテナはサテライトＩＭＰを備えつけていて，そのすべてが１つのチャンネルを使うことになる。すなわち，アンテナのどれもが同じチャンネル上で発信及び受信の両方を行なうことができる。それは単一チャンネルであるが，全二重スイッチサービスを行なっているかのように見える。その方法は，地上ステーションでエレクトロニクスがキャリアーを操作させたり止めたりすることである。各地上ステーションは両方向に５つのワイアーインターフェイスを有しデータ，クロック，グランド等の信号をワイヤに加えるようにするのである。一方，他のワイヤーはキャリアーにタイミングを続けさせたりするわけで，このタイミングはサテライトＩＭＰにより注意深くコントロールされる必要がある。技術的には誠に簡単であることがわかるであろう。問題になることは，効果的に利用するにはサテライトＩＭＰにどのようにしてチャンネルをコントロールさせれば良いか又送る各パケットがうまく行かなければどのような事が起こってくるかということである。

ＡＲＰＡネットにあったようなケースを１つあげてみよう。送り先のアドレスと何かテキストがあるとすれば，この送り先のアドレスは実際は一つではなくいくつかのアドレスが含まれた合成のものである。どのサテライトＩＭＰがこの特別な発信を拾うべきであるか，あるいは，以前示したと思うが，小さなローカルヘッダーがあるＩＭＰを示しており，これは，どのサテライトＩＭＰが発信を拾うかを告げる。つまり，ここに示されたアドレスはどのサテライトＩＭＰが受信すべきであるかを告げる。そしてこのサテライトＩＭＰのアンテナが発信すると，通信衛星が受け，それをすべての場所へ送るわけである。例えば，Ａに対して宛てられたとすると，ＡにあるサテライトＩＭＰが回集し，他のＩＭＰはそれを捨てさるわけである。一度回集したならば，チェック・サムによりエラーチェックを行なって，返答を宛てられた送り手に送り返すのである。実際にアドレスに述べてあることは誰が送ったのか，いつ正しく受信されたのかということであり，ＡＣＫ（承認）が返送され，そのＡＣＫによって発信地のコピーが捨てられることになる。仮に（ＡＣＫ）がうまく戻らなければ，データがさらにもう一度発信されることになり，処理もくり返される。これはシステムデザインが許す限り，できるだけ何度も行なわれるわけである。５回なり10回，あるいはそれ以上必要となっても良いわけである。これをもっと便利に修正する方法がある。２つあるいはそれ以上のアンテナに同時にキャリアーを乗せると，パケットが送られる時２つのパケットがぶつかり合い，両方ともある時間通信衛星上で，オーバーラップしてしまい，また両方とも違ったビットをもつことになる。そして，チェックに引っかかり両方とも捨てられてしまうことになってしまう。仮に２つのパケットが同時に送られたら両方ともなくなってしまい，次に２度目に送られた時は，１つだけが正確な目的地に行くようになっているのである。

この操作方法は"ALOHA"と呼ばれ，ハワイ大学で実現されたシステムで使用したものである。しかし，ここでは通信衛星を通して行なっていなかったのであるが，大学構内でターミナルから中央点へという方法で作業を行なっていた。この方法がどういうものであるかというと，パケットがあり，アンテナのうち1つでもこのパケットを送りたいということであれば送ることができ，通信衛星まで送られ，ある時間内に受信される。また適当な受信者が見つけられると，同じくパケットがある時間内に通信衛星に送られ受信される。トラフィックの流れがあまり大きなものでない限り，2つ又はそれ以上が同じ時間フレームから発生する確率は小である。また仮に，トラフィックがポアソン到着に従って発生するものと考えると，アンテナの2本あるいはそれ以上がオーバーラップされる確率を考えることができる。そして，この回線のthrough putも計算でき，その値は1／2ないし約18％である。従って，この技術を使って各アンテナがトラフィックを発生させ，また同一のパケットの長さで，コントロールせずにトラフィックを発生させようとすれば，システムは18％のdutyサイクルで飽和状態になる。言い換えるなら，再発信が起こった場合，後になって，ばらばらの時間に再び再発信が起こり，システムが完全に一杯になった状態になってしまう。あるいは，50 kbの回線上でthroughputが18％，約9 kb／sであるが，この場合も一杯になってしまう。従がってこの方法はあまり良くない使い方である。

2番目のケースを考えてみよう。これは，Slotted Alohaと言うもので，Slotted Alohaに於ては，固定された転送の時間サイクルにだけパケットを伝送することが許されている。そして1000 bitのパケットを50 kb回線で送る場合は20 msとなっている。パケットがある地点から送り出されるのは時間サイクルのまん中からではなく両端からである。従がって2つのパケットがぶつかり合う唯一の可能性は2つのアンテナが，インターバルの中心から発信する場合だけであるから，1つのパケットが一方のインターバルから，もう1つのパケットは他方のインターバルから発信されるような場合にはぶつかり合うことは決してない。各地上ステーションのハードウェアはパケットを正確な時間に送り出すことができ，同じことが，サテライトIMPも行なえるように考慮されている。そしてその中では正確な時間をきざむことができる時計（counter）が備えられている。それで，キャリアーが正確に動作するわけである。この方法によれば，チャンネルに関して，時間はディスクリートなslotに分割されるわけである。そして，これらslotの始点からのみパケットを発信することにより，ぶつかり合いが避けられる。これによって実際，その能力をAlohaの2倍にすることができる。$\frac{1}{E}$すなわち，36％，37％あるいは36.8％ぐらいの能力にできるのである。従がってslotを行なうことでチャンネルのthroughputを2倍にできるわけである。その場合，伝送するパケットはフルサイズである。これは固定長パケットを使ってsoltなしのAlohaシステムとslotされたAlohaシステムとを直接比較したものである。能力は，各々18％と36％でスロットされた方が2倍良くなっている。

今後，我々が行なおうとしているのは，このslotted Alohaのシリーズの実験成果を研究しそのパフォーマンスを測定することである。特に通信衛星におけるmodulation effectを計っていくつもりである。我々が理論的に研究したところによると，そのほとんどは重要な問題である。1つのメジャーとして，通信衛星上で大きなユーザーに与えるインパクトと小さなユーザーに与えるインパクトを考えると，大きなユーザーの場合，すばらしい特徴をもっており，それを干渉されたくはないということなのである。しかも大きなユーザーの場合，36％以上を期待しており，実際のところは50％以上である。

次にReservationの技術を考えてみることにする。この方法で我々は通信衛星チャンネルの能力に関して100％に到達することが可能であるかどうかを実験しようと思っている。Reservation技術というのは，Slotted AlohaあるいはUnslotted Alohaのどちらよりも遅延が長いものであり，slotの考え方で動作するものである。即ちサテライト・チャネルはslotに分けられ各スロットには番号が付けられている。ユーザーが大量のデーターなり，あるいは少量のデーターを送る際に，始めにReservationを送る。そしてReservationに対してスロットが割り当てられる。例えばパケットが3つあるとして，あるユーザーが，2つのslotを使用するとする。そして他のユーザーがそれ以上のslotを使用したい場合，サテライトＩＭＰの各々が，そのデーターにはどのslotを使えば良いかを記憶しslot時間が来るまで待機するわけである。それから，送り出しにかかるのである。従がって，最初に使用可能なslotがやって来た時に始めてデーターを送ることができるようになっている。Reservationを扱う一つの技術として，Alohaを使う方法がある。この場合Slotted Alohaを使ってもUnslotted Alohaを使っても良いが，普通はReservation用にはslotted Alohaを使う場合が多い。slotを行ない，1つのslotを非常に多くの小さなslotに周期的に分けてゆくことである。slot間の周期は，どれだけのトラフィックがあるかと云うことで決まる。仮に，トラフィックが多くあれば，全部分離することができるであろうし，全くトラフィックがなければ，すべてslottedにはならないでReservation slotになってしまうのである。Reservationは小さなものであるので，全slotをとり上げることはできない。Reservationを送ればReservation slotがもどってくる。そのスケールは50 kbぐらいで，12のスロットに相当する。Reservationが競合してやって来ると，それらのslotの1つを得ることができるが，得られなければ，再度，送り込まなければならない。あるいは，マルチプルコピーを送って，Reservationが得られるようにするコーディング技術を利用する方法がある。2重，3重，あるいは，さらに多くすることが可能である。しかし一般的には，Reservationを送ることによって行なうわけである。例えば7つのslot分が欲しければ，次にくる7つのslotを告げてくれる。各サテライトＩＭＰが，何がリザーブされているかのトラックをもっているのである。サテライトＩＭＰが7つのパケットを得たい場合，7つのパケット分をリザーブしていることを告げるメッセージを送るだけでよく，これらのパケットはすべてのユー

ザーに受け取られる。そしてあるユーザーが自分えの伝送を受信した時，７つのパケットが得られたことがわかる。ユーザーがすることと言えば，自分達のケーブルを用意しさえすればよいのである。そしてユーザーは，送られてくる時まで待機し，次に自分のパケットを送るわけである。この後は，Reservationはなく，後は小さなslot付きのslotted Alohaシステムに戻ってしまうわけである。

もう一つの技術として，TDMAシステムというのがある。TDMAは今まであげた他の三つの技術よりも簡単である。但し，あまり良いパフォーマンスは得られないのがいささか難しいところである。費用の面では同じ遅延時間で，他の技術と比較した時，高くはないが，先程のReserva-tionシステムの方が優れている。コストとディレイの関係を表わした曲線をFig 33に示しておこう。

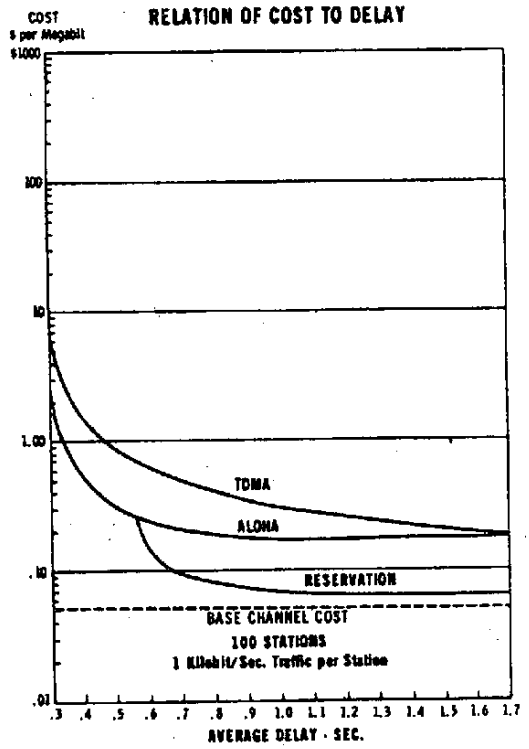


Fig 33

これは米国においては50 kbチャンネルの場合，１Mb につき約５セントであるというコストに基づいたものである。実際，高バンド幅の回線を使えば，もう少し安くなる。slotted Alohaの場合は約15セントであり,約３倍高い。

TDMAについて 同じコスト／ビット（コスト／メガビット）を得るためには，Reservationシステムに対しておよそ２倍以上のディレイがでてくる。

他のシステムとしては全てのポイントを他のポイントに結びつけることで，基本的に完全に結合されたネットワークを構成することである。又別のアプローチとしては，中間に一種のステーション・ポイントとして1点を指定し，全ての他の接続ポイントから1つの回線をとってくるというのがある。一種の衛星による星形構成であるが，これと完全に結合されたものは，みな，さらに高価なものである。

ここでコストをステーション数に対してプロットされた各ポイント間をつないでみるとFig 34のようになるが，リザベーションシステムの場合だとこれが最小のところに下がって来て，他の場合もみなTDMAの曲線より下に来る。

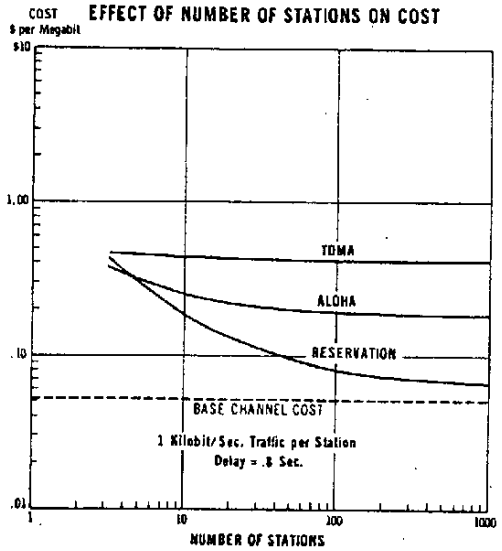


Fig 34

もし，いつも全回路をフルに利用できるように，システム内に十分なトラフィック量を得れば，実際 " point-to-point circuit " を用いるのと同程度の効果をあげることができるが，もし，バースト状態のトラフィック状況に近い場合はそのバンド幅の共有ができるということが基本的に必要となる。どんな場合にも，4種類の機構があって，我々はこれらを調査する計画をたてている。サテライトＩＭＰは既に開発されており，我々は国内のキャリアー達と話を進めているが，装置の組み立てに要する時間の関係から実際のテストに入る迄にあと6ケ月乃至1年間ぐらいかかるだろうとみている。

バースト・モードのＩＭＰを衛星システムに組み入れておかねばならないが，これにはある程度の時間がかかる。しかし少くとも近いうちに，これらの技術をパケット交換技術に織りこんでみたいと思っている。

たくさんの地上回線を敷くよりは長距離データ伝送を行う方がはるかに効果的方法だと思われる。

ARPAの援助 のもとに行われてきたこの種の応用の，かなり広範な理論分析があるが，ここに1973年の（National Computer Comference ）で発表された3つの論文がある。そこには，ALOHAに関するスループットの評価及び，ディレイ・スループット評価，それにリザベーション技術について書かれてある。この技術については近いうちに実験する予定であるが実験の総合結果はすべてキャリアー達にとってきわめて興味あるものとなろう。というのは通信にサテライトのバンド幅を利用する方がよりよいということになるからである。

経済の面で左右するものは実際にはターミネーションの部分であることがわかっている。そういうわけで，全体の経済分析では経済的ではないにもかかわらずTDMAに我々は注目しているのである。費用に関しては，大勢を占めているのは圧倒的にターミネーション，アンテナとか地上フィーダー回線等のコストであって，他の技術のどれを使ってみても結局大量の節約ということはできない。このことはスロット付きのALOHAシステムを使った場合，経済性の点に於いて全体に非常に良好であるということであろう。しかしながらもし，ディレィの方にもっと関心が置かれる場合には，TDMAの方が主たる利点を有しているので我々としては4種全部を研究してどれがネットワーク構造として最も経済的であるかを決めようと考えている。

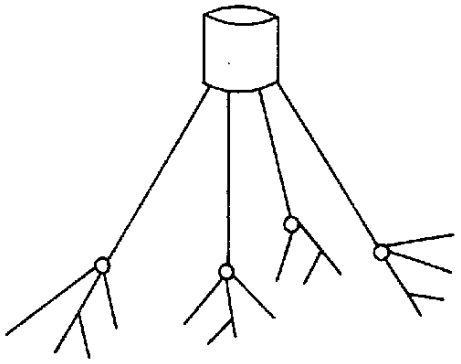


Fig 35

ここでARPAネットでの衛星及びターミネーションポイント等を考えるとその形はFig 35のようである。

いくつかのターミネーションポイントから更に分れる形になるであろう。

これら全体的なローカル回線は多くのブランチをもったサテライトﾁｬﾈﾙのようなものであり，それらの間に一時的なリンクを設けるのである。速度は1～10MB 位で地上のリンクはもう少し遅くなるであろう。そしてサテライトにより非常に大量の伝送を扱うことができる。

この方法によりファイル・トランスファの実験や，ネットワークの信頼性の実験にかなりの柔軟性がもてたのである。さほど長い距離でなければより多くの低速回線を増やすことが出来，また非常に高いバンド幅の伝送も扱うことができる。そして非常に長いファイルも安い費用で送ることができるようになるのである。

質問応答

質問 1

大きなファイルをどう扱ったのか。つまり，ネットワーク内でどのようにして大きなファイルを運ぶのか。

答

基本的には任意の長さのシーケンス・メッセージとして送られる。たとえば何百万ビットもの長さのものもあるだろう。例えば8000ビットのメッセージを送り次にまた8000ビットをひきつぐき送るという具合である。メッセージを何ビットにするかは，むしろオペレーティングシステムの性質によるもので，大きさは，8000ビットとは限らずそのオペレーティングシステムによって磁気テープから記録が取り出せ，また磁気テープに記録される大きさになるだろう。伝送は次から次へとシーケンス番号のついたメッセージを連続的なセットで送るという具合であり，メッセージは受信側に於いて一つに集められ，送られた時と同じ順序で，記録の順序付けされたものが磁気テープ上に書き出される。これを操作するのに使われるのが"ファイル・トランスファ・プロトコル"とよばれるものでプロトコルの一つである。

質問 2

もしあるジョブがどのHOSTでも処理することが可能であれば，リソースを適当なHOSTに割り当てるか。

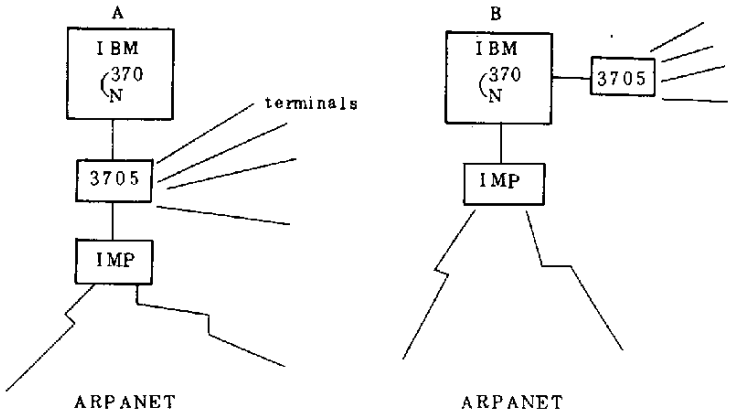
答

現在のところ，リソースをHOSTユーザ或いはHOSTプログラム等に自動的に割り当てることに関していくらか実験を行ってきているが，実際に行われている仕事の殆んど全部が自動的に行われてはいない。

ユーザが特にHOSTを指定してジョブを行わせ，ある場所からファイルを取り出す。全てのプロトコルとプロセデュアについて自ら知っているのである。少量の仕事でこのプロセデュアを自動的に行っているものもある。

質問3

ＩＢＭのシステムをこのＡＲＰＡネットワークに接続する場合は下図のA，Bのどちらをとるか。



答

どちらの場合にせよ，ＩＢＭの機械への接続は我々の実験で行われたものである。ＵＣＬＡの360−91は2703を経由して接続され，別のＩＢＭマシンにはセレクタチャネルに特別のＨＯＳＴインターフェイスが装備され，直接ＩＭＰと交信するようになっている。

質問4

ネットワークの負荷の大部分はどこにかかっているのか。またネットワークによるＨＯＳＴコンピュータの負荷は何％位か。

答

さき程ネットワークで非常に大量に使われている６つのコンピュータ（USCのPDP−10，UCLAの360／91，ILLIAC Ⅳなど）を紹介したが負荷はこれらの上に主に分布されるであろう。またHOSTの負荷を推定するのは難しいが，USCのISI（ISI：Information Sciences Institute，Calif），の場合だと機械の殆んど100パーセントにあたっている。MULTICSの場合はたぶん10％以下であり，AMES（AMES Reserch Center，Calif．）のILLIAC Ⅳの場合にはおよそ1000～1500万ドルのうちの40万ドル分程度の使用であろう。多くのネットワークサイトではまだ低使用度の段階だが徐々にさらに多くのトラフィックが使用されるようになってきている。

質問5

ネットワークに集中フローコントロールがないと，その使用料をユーザーに請求するのは難しいのではないか。又，ARPAに類似した商用ネットワークの場合のユーザーへの使用料請求はどのように行われるか。

答

商業用ネットワークについては話すことはできないと思うが基本的にはネットワークは各HOST

に入ってくるトラフィック量に関する情報を絶えず得ている。このデータの収集は，ＩＭＰの中のＮＣＣレポートというプログラムによるものである。HOSTにパケットが入ってくるたびに，ＮＣＣレポート用のテーブルに記録をしていき，１分毎或いは数分毎にネットワーク・コントロール・センターにメッセージを送る。そこでは全てのメッセージが蓄えられ，製表・印刷が行われる。

従ってHOSTに入ってくるものはそのHOSTに対するネットワークのチャージングによって製表が行われる。これらのチャージ（料金）をどう分配するかを決めるのはHOSTによるわけである。合計のチャージが平等に分けられるか，或いはユーザーが送るパケット量によって決めるかいずれにしても，ネットワークにとってはこの分配法に対する条件のようなものは何もないのである。１つのＩＭＰに複数のHOSTが結合されている場合も，IMPは夫々のHOST用にわけて記録をとる。商用のネットワークに於ても恐らく似た様な方法をとると思うが，接続料（パケットの伝送料）に相当するようなものも加えるかもしれない。

質問６

コア・ストレージの図（Chart 23）でスーパバイザが書かれていないが，このプログラム間或いはタスク間の流れのコントロールを行うものは何か。

答

たしかにスーパバイザはない。プログラムは，完全に外部媒体によってコントロールされているから非常に処理速度が速くなっている。

プログラムコントロールの構造はそのオリジナルの当時の方法と全体的に大きく変ってはいない。スーパバイザを持たぬというのは最初からの思想である。

それぞれのプログラムはハードウェアのインタラプトによって起動される。またそのインタラプトの優先度も正しく処理される。インタラプトによる処理が終ると，レジスター類が復帰されバックグラウンドのプログラムが続行する。

質問７

ＮＣＰをＩＢＭコンピュータにも作ることはできるのか。もし可能なら，プログラムの構造は？またＯＳ内のインターフェイスはどうなのか。TENEXに於て，ＮＣＰは，モニターの初期設定，およびユーザプログラムの実行の取消し等を行うことができるか。

答

ＮＣＰは種々のコンピューターに対して使われてきた。ＩＢＭの360－91，360－75，360－55，360－67等，これらは異った形で使用されているが，ネットワークの図をみればこれらＩＢＭ機を有する場所が書かれている。

プログラムの構造はかなり複雑であり，これについては後述するがかなり大きな負荷になってい

るようである。

TENEXに於て，ＮＣＰはモニターの初期設定或いはユーザーのプログラムの実行取消し等の作業は行わない。ＮＣＰの機能は，ユーザーのプログラムとネットワーク間のインターフェイスとして働くものであって，データを取り入れたり，取り出したりすること或いは，ユーザーのプログラムのリクエストに応えたり，他端のユーザ・プログラムから入って来るリクエストに応えたりすることができるものである。接続故障等のネットワークに欠陥が生じたと思われる時に応答する以外にそれ自身で動作を起すことができないものである。

質問 8

ARPAとＢＢＮの関係について。ARPAネットワークでのＢＢＮの役割について。

答

ARPAは政府機関であり，ＢＢＮは民間の会社で1948年頃，ＭＩＴ出身の音響学の教授３名から出発したもので，まだ規模は小さいが，非常に高度な技術を持ち特に音響の分野では優れている。コンピュータ及びコミュニケーションに関する専門家はほんの数人しかいないがＢＢＮ社は歴史的にも今日に至る迄，音響分野の専門会社であるということで最もよく知られている会社であろう。

彼らは合衆国内の大部分の新しい音響ホールを設計している。

ARPAネットワークは商業用ネットワークではなく，研究開発用ネットワークであり，現在の形では商業用に使用されることはないし，できないものである。使用されるのは，一般にARPAが研究の契約をしている米国政府研究機関に限られているが，その他の政府機関も他の手段を使って行う場合よりも経済的である場合にはその使用が認められている。

ＢＢＮの場合は主たる仕事としてネットワーク開発を行っているたぶん10数或いは20数機関のうちの１会社である。

ネットワークに関して我々が1966年から72年にかけて特にやってきたことはサブネットの通信の部分の設計とその後のインプリメンテーションを行うことであり，この仕事は継続的なものであってARPAからの委託の仕事であった。

この仕事をＢＢＮか他の業者等にまかせることになっており，ＢＢＮとの関係を続けたのは，彼らが非常にいい仕事をしてきたからである。

質問 9

TYMSHAREのネットがARPAに接続されるということを聞いたことがある。

またARPAネットワークの地図上にもその名前が書かれてあるが，これも商業用のネットワークであり，どういう目的でこの接続は行われたのだろうか。

答

図上にあるTYMSHAREと呼ばれるこの接続は，実際はTYMSHAREネットワークではない。

ＳＲＩがＮＬＳというエディティングシステムを開発したがＳＲＩは，ＰＤＰ－10にこれを

適用したオペレーティングサービスに対して，入札するよう米国内の各種私企業にあたっていたがその時，TYMSHARE社は，これに対し最も低い付け値で入札した。そこで同社は，ＰＤＰ－10を運転し，ARPAネットワークとの接続のあるサービスをとり扱うことになった。

しかし，このサービスは彼らがTYMSHAREのネットワークで使用している機械とは全く別の異ったものである。従って，これはTYMSHAREのネットワークとは全然関係のないものである。電気的にも接続は行われていない。

TYMSHAREのネットワークとの接続については過去にその可能性について話し合ったことがあり，その時実験的に1972年10月，ワシントンＤＣでのＩＣＣＣ会議に於いて，３日間接続してみようということが決められた。しかし会議が終ったその日にこの接続は，はずされ以来再接続されることはなかった。実際，国際的な接続に関するノルウェー及び英国との協定にもとづき，我々は外国の承認なくしてARPAネットをTYMSHARE に接続することができないということにもなっているし，もしこれを無視すれば，現在我々が既に有する外国との接続を失うという危険を犯すことも覚悟しなければならない。業者は全ての国際線接続に非常な関心を持っており，新たに接続する時はいかなる場合でも電気的に接続された国際回線の全てを含む関係者の合同の了承を得なければならないという非常にこみいった問題があるのである。

質問１０

ARPAネットワークはロンドン，ノルウェーまで拡がっているが，その社会的理由は何か。また，このネットを使用している機関及びその目的について。

答

ノルウェーへの接続はケラーにある地震研究所のＩＢＭマシンと結びノルウェーの地震のデーターをアクセスしニュークレア・ディテクション（Neuclear detection）に使用されている。

ロンドンに延長されたのは，大体これと同じ理由によるが，数年前に米英両国の間で地震データに関する協力ということで締結された協定の１部分として，このデータベースを英国と共有することとなっている。従ってこの全体の結合の主たる目的は，地震データに関する情報交換のためである。同時に，英国との間では相互にデータをコミュニケートするネットワーク間のプロトコルについての研究も進められている。これが社会的理由といって十分なものかどうか確信はないが，これが結合の主な目的であるといえる。

質問１１

異種のコンピュータ間のデーターベースを共有するというプランはあるか。またそれに関してどのようにしてデータ間の変換（Conversion）を行なうのか。

答

現在の最大のプランとしては，世界の多くの場所に於いて存在する地震のデータ・ベースを共有することがある。

現在，接続されている数ケ所では夫々自身のデータ・ベースを有している。また地球の気象関係のデータ・ベースとしてそれぞれの場所でのローカルなデータをもとに分散型のものを作ろうというプランもたてられている。その一部はAMESのILLIAC及び，CCA（Computer Corpora-tion of America）にそれぞれ大量のデータの蓄積が行われている。他にもARPAに接続されてきたものの中にデータベースを持ったものもあったが，それらは分散型のものではない。例えばケミカル・アブストラクト・データベースやMEDLARS等があるが，これらはTIPからや，やゝ低速のTSSで使用されている。

また変換に関しては私の知っている限り，データベースについての変換は現在の時点では行われていない。

質問12

メッセージのルートはどのようにして見出すのか。

答

基本的にはルーティング・アルゴリズムは最小時間の推定という仕事をやっているがもう少し詳細は後述することにする。メッセージに対するルートは内部的に作り上げられるものでユーザーはどういうルートを通って来たかは知る必要はない。

ネットワークの感じとしては，入力したメッセージが他端からポンと飛び出てくるという感じなので，ネットワーク中でメッセージがうろうろさまよっているといったような感じは微塵もない。出てくるか出て来ないかということだけが問題でこれは信号がラインの一方から入って他方から出て来るのと何等変らない。

質問13

トポロジーの研究について及びその関係文献について。

答

トポロジー研究の主要文献は配布した資料の中の「Computer Communication Network Design」と題する論文の最後の参考文献のリストに書かれているが，その内から取りあげてみよう。

№8は，トポロジーの参考になる。トポロジー研究にはルーティング・コンピュテーションが必要であり，№8の「Routing in Computer networks」には"フローの伝送網への割り当て"が取り扱われるので有用であろう。これはネットワークのルーティングに使われている実際のアルゴリズムではなく，フローの伝送網割当て法についての文献である。

№9の「Cost & throughput in Computer Communication networks」は直接，トポロジー研究に関係するものである。

№12の「Topological Considerations in the design of the ARPA Computer network」は，たぶんトポロジーデザインに関する文献中最良のものであろう。

№13は,ルーティング技術の習得法の紹介という意味でいくらか関係のあるものである。

同じ雑誌中の論文に「Resource Sharing in Computer Communication Networks」に関するものがあるが，この中に「トポロジー・デザイン・クライテリア」に関する参考文献のリストがのっている。実際，全部トポロジーについては触れているものだが，Ref．34～Ref．59 の文献は全部，トポロジー・デザインについての参考文献である。

トポロジー研究の目的は，ネットワーク・トポロジーのデザインのための技術開発にあるが，これは，伝送網速度，伝送網及びノードの信頼性とコスト，インタフェイスのコスト等に対しての広範囲の仮定に基づいている。

その主要パラメータはディレイ，スループット，コストのスループットに対する比，等々である。

トポロジー研究の詳細については，Network Analysis Associationが提出した論文にも書かれている。

このAssociationは,ビジネスとしてサービス提供を行っていると思うが，本来，ＰＣＩとの特別契約をしていて，開発した最新技術はＰＣＩに提供され，それらは公表されていないと思う。

彼らはブランチ交換技術等について公表したことはある。彼らが努力を傾注したのはネットワークのプログラムデザインに於ては，プログラムのランタイムの最適化に関するものであった。

質問 14

RFNMはＡＣＫを受けないのか。

答

RFNMは，メッセージを先方へ送ると返ってくる応答メッセージである。RFNMはＡＣＫ を受けない。

各ノード対の間では各メッセージがチェックされＡＣＫを受けるがその場合は，RFNM も同様に承認を受ける。しかしRFNMが返ってくるとそれに対して，再び逆方向に返っていくメッセージは他にはない。

質問 15

ＩＢＭ機の使用が少なくＰＤＰが多いのは何故か。

答

ＩＢＭ機は実際には少ないということはない。私の知る限りネットには少くとも５機あるし，これはおそらく全体の１０％にあたるものである。コストにおいては他を圧倒しているだろう。多くのＰＤＰをネットに使用しているのは，普通我々は機械の選別に関しては夫々のユーザーに任せていることによる。

ARPAネットではどの機械を買ってどれを買わないことにするといったことの決定はしていないし，ユーザーも自分で探してコスト的にも最適のものを選ぼうとする傾向がある。

そして彼らがＰＤＰの方がＩＢＭ機より購入に適していると判断した結果そうなったのであろう

と思う。これは何もＩＢＭ機の方が機械として優れていないということではない。ＰＤＰを買った方が価格が安いという判断であろう。これは私個人の考えである。機械の購入の多くは3年或いはそれ以上前のことであり，最近買われたものは殆んどない。

質問 16

"ＴＩＰ user dialogue (Chart 17)" のスライドで@Ｌ23↙の↙記号はエコーによってプリントされるのか？ また@ GET FILE のアンダーラインの部分はリモートHOSTから送られてくるのか？

答

↙は単なる復帰改行記号である。エコーによるプリントではない。またETとＩＬＥ はリモートHOSTからネットワークを通して送られてくる。この場合，GとFは赤字，ETとＩＬＥ は黒字でプリントされる。

質問 17

遠隔のHOSTからエコーが送られる場合，1つのキャラクタ・パケットがつくり出されるのか，もしそうであれば，経済性はどうであろうか。

答

この質問で聞いていることは，1キャラクターづつがそれぞれ1パケットとして送り返されるのかどうかということと，もう1つは，その場合の最も経済的にそれを送り返す方法についての質問である。

これはおそらくあらゆる場合，3キャラクタパケットの形で送り返されるものである。この場合にはたぶん " ET Space " が1つのパケット " IEL Space " が1つのパケットという具合に送り返される。

もしシステムが本当に良好なプロトコルを持っていれば，ET Space とＩＬＥ Space は即座に返ってきて，プロトコルはそれをうまく順序よく並べるであろう。

即ち，GとFが送られる方向と逆方向に " ET Space " と " ILE Space " が送られてくるわけであるが，プロトコルはGのエコーが " ET Space " であることを心得ており，そのエコーが帰ってくるまではFをプリントするようなことはしない。例えばG F ETなどをいう順序にはならない。

これは複雑なTELNETのプロトコルであり，このネットワークでごく最近まで使われていたものだが現在では使用されていない。現在のは " GF " を送ると，" GET Space FILE Space " がエコーとして返ってくるようになっている。

質問 18

VDHの場合，メッセージのリアセンブリーが行われるのはどこか。HOSTそれとも，IMPで行われるのか。

答

メッセージのリアセンブリは現在常にネットワークで行われておりVDHは単に，1度に1パケットの割合で，リアセンブリされたメッセージをHOSTに送るだけである。

質問 19

HOSTのインタフェイスサーキットはBBNの設計・製作によるものか。

答

そうではない。一部，たとえばPDP−10のインタフェイスはBBNの設計・製作であり，IBMの360−91のインタフェイスもそうであるが，一般的に各場所に於いては各々の機械に対して彼ら自身のインタフェイスを製作している。

質問 20

衛星チャンネルのコストは安いものか。

答

非常にコストは高い。しかし，高いといっても例えば50キロビットチャンネルのコストは全く同じヨーロッパのものと比べて4倍，日本に比べて5倍程安いと思う。同じ回路，同じ装置についての比較である。

質問 21

TIPに於いてターミナルへのメッセージ伝送の完了後，TIPがRFNM を返すのは何時か。

答

TIPがRFNMを返すことはない。IMPがRFNMを返すのであり，それは，メッセージの最初のパケットをHOSTに送った後に行われる。RFNMの目的は本来,主として，フロー・コントロールのためであり，その意図は伝送の開始を確認することにあり，これによって全てのパケットが到達したことがわかり，そのシーケンスを保つことができる。

質問 22

VDH（Very Distant HOST）インタフェイスについて。VDHインタフェイスのユーザーによる評価或いは他にユーザーからの要求等はどうか。

答

これはいい質問だと思う。VDHインタフェイスはプログラミングを要するので開発が比較的難しいものである。一方，エラーチェックができるのでたいそう信頼性のあるものである。VDH インタフェイスにはいろいろと関係するものがあって複雑だが，一面エラーチェック機構があるという有利さもある。同期回路による結合がそれ程要求されない場合にはVDHインタフェイスをむしろ製作しない方がよいというユーザーが殆んどである。

VDHインタフェイスの製作は他のどのインタフェイスの製作よりも高くつくので，もし選択が可能な場合にはいつでも直接継続の方が選ばれる。

質問 23

Distributed Computation について将来,大容量のコンピュータたとえばILLIAC−4 のようなものが必要になってくると思うか。たとえば非常に多くのミニコンピュータを結合し大量の計算を同時に行なえる様になった場合も必要か。

答

この推定はめんどうな問題である。ILLIAC−4 についてはその代りに多くのミニ・コンピューターがあればよいと云えるかもしれないが超大型機の主たる難点は，その製作そのものではなく，そのためのソフトウェアの開発にあり，現行のバージョンに対して大きな投資をしてきたわけであるが，このようなものを早急に再製作するということはするつもりはまずないであろう。

しかし問題は，機械の大きさに重点があるのではなく，その容量が十分あるかどうかということであろうし，もし十分でなければ，もっと容量を増すにはどうすればよいかということだと思う。

質問 24

映画の中でNPLのDavis がARPANETの批判をしていることについて述べてほしい。

答

彼は別に批判しているのではなく支持していると私は信じている。ARPANET がスタートした時は多くの批判があったが彼はその１人ではなかった。

質問 25

Fig 5のネットワーク・経済性の計算式とそのカーブについて説明して欲しい。

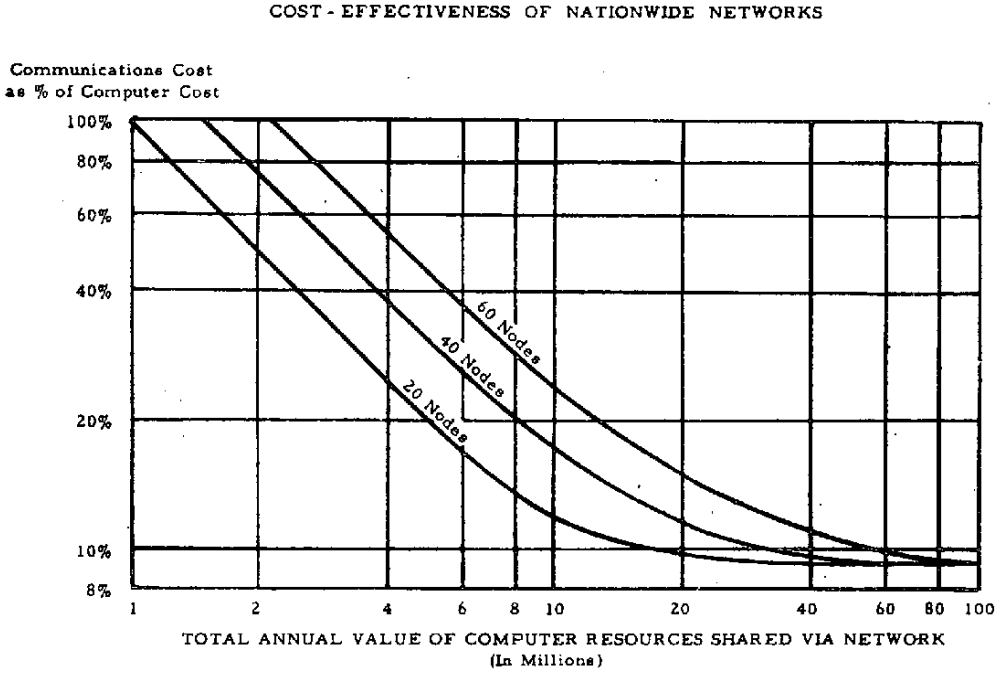


Fig 5

答

Fig 5の図は式にもとづいてつくったものではなく，Network Analysis Corp. が種々のノードの数とそのコストについての数字から作ったものである。これは，計算のリソースの相当多くがネットワークを通して使用されているという仮定に基づいた資料である。

ネットワークで計算を遂行する時にトラフィック量のおよその価格の見当をつけるのには多くの計算方法があるが，この図は，種々のノードがどれ程のトラフィックを受け取るかを見積もって，このノードの関数としての種々ネットワークのコストをプロットして曲線を適当に描いたものである。

質問 26

現行の種々のコンピュータ・ネットワークについて技術的に分類してほしい。その各ネットワークの利点及び欠点は？

答

後述すると思うが，ここで少しばかり説明しよう。おそらく現在2ダース位のネットワークがあるであろう。

ARPAネットワークは米国に於けるパケット・ネットワークの内で最大のものである。

私の知っている米国の他のネットについては小さなものでミシガンにあるMERITネットワークがある。これは大部分はARPAネットをモデルにしたものだが，方法に於いては多くの点で異っている。PDP−11を使用し，ウェイン州やミシガン州のミシガン大学が接続されている。彼らは dial-up を用いているか或いは少くとも電話線を使っているがそれは全て低スピードである。私の知る限りでは，それはプロトコルの開発において遅れをとってしまった。現在まだ非常に有効に使っているとはいえないのではないか。

また，ハワイ大学で開発された小規模のネットワークがあるがこれはALOHAシステムと呼ばれていて，これはARPAに支持されたパケットの技術にもとづくものであるが，基本的には回路にパケットを送るサーキット・スイッチ・ネットワークである。従ってこれは低スピード回線をつかうハイブリッドシステムの一種であり，その最適化はアイボールによって行われる。必要な時に他の回路を加えるが，その接続を最適化するという計画はされていない。

信頼性上の欠点としては，たとえば回路の故障があると接続されないままの状態になって再度ダイヤルインしなければならない。この対策として適用されるべきルーティングの計画はないようである。

ベルシステムは内部的には種々のパケットネットワークに注力していた。しかし私の知る限りでは正式のサービスを彼らが提案したということはなかった。彼らは非常に強くPCIやTelnetを支持している。

IBMは非常に熱心にコンピュータ・ネットワーク研究に力を注いできたが，それは主としてリ

ソースのシェアリングという点に関してであり，実際にネットワークを建設するという計画をもっていたとは思えない。

Irvineにあるカリフォルニア大学に於いて，パケット技術に基づいた小規模のループ・ネットワークが開発されたが，これは非常な成功を収めた。

イギリスに於いては，EPSSと呼ばれるネットワークがあり，これは実験的パケット・スイッチングシステムとして，英国逓信省が開発中のものである。ノード間48キロビットの，3ノード・ネットワークで種々のノード間のコールを設定するもので，これは回路とパケットスイッチングのハイブリットシステムであるが彼らはこれをパケット・スイッチング・システムと呼んでいる。

NPLは，研究所内にHoneywel 516を修正して多数のターミナルが接続された形の小規模なパケット・スイッチ・ネットワークを持っている。現在まだ他のネットワークとの接続はないと思うが，彼らはこれに，Node-to-node packet switch capability（ノード対ノードのパケットスイッチ機能）をもたせようということで拡張の計画をしており，プロトコルを熱心に研究している。

フランスには現在開発されているネットワークがあり，最初のノードはもう間近に，たぶん1ケ月或いはそれ以内に使用できる状態になるであろう。これはCYCLADES と呼ばれており，このサブネットはCIGALEと呼ばれる。これは非常に将来的なネットワークでIRIAのルーイ・プザン（Louis Puzan）の指導下に開発が行われている。

フランスのPTTもまたパケットを伝送するサーキュットスイッチネットを開発している。これはCYCLADES とは全く異っているもので，パケット・ネットというよりサーキュットスイッチネットのようなものである。

カナダの研究者達はパケット・スイッチングに興味をもっていて，いくつか異った観点からアプローチしたが，そのうちの1つとしてCANUNETというネットワークがあって，これはカナダの大学間を結合している。これはオンタリオ・プロビンスで始ったものと思う。他にも現在進行中のものがあると思う。

スペインには，5～6ノードを有する稼動中のパケットスイッチングシステムがあると思う。これはおどろくべきことだった。私たちの手許にはそれに関するデータはあまりないが，これは現在設置され稼動されていることはまちがいないと思う。

この他多くのメッセージスイッチングシステムがあるが，こゝでは省略する。

質問27

IMPのプログラムがリロード（reload）される場合，そのIMP個有のコントロールテーブルはどのようにして再設されるか，これらはプロテクトメモリーに入っているのか。

答

IMPのプログラムはすべて同一である。とくにコントロールテーブルの様なものがあるわけではなく，またこれらがプロテクトメモリーに入っている必要もない。たゞキーパラメタはプロテクト

されており，それらはリロードの際，セットし直される。

質問 28

コスト対ディレイに関して。コスト対ディレイの比較表によってARPAネット が最良であるということであったが，トラフィックについてはどのように仮定しているのか。

答

この計算はトラフィックのスタートからストップ迄，設備の連続使用を行った場合のものである。たとえば 1000～10000 ビットの大量のブロックを送りたい場合，トラフィックは普通，一時的に送られてから少しの停止が行われる。従って，ダイアルアップ回路を使っている場合，トラフィックが送られ，回路が通じてから断たれるその時に，次のメッセージが送られる。通常，こうしたやり方に対して，リースの回線のようなものを使う場合には回線をその都度切ることはないだろう。実際には，いつ次の伝送が起るか予測がつかないのだが，ここでは実際のコストを計算することができるように，仮定としてこうした形で伝送のシーケンスが行われるということにしておこう。たとえば，1日当たりのパケットという表わし方が実際には意味のないものだとすると，パケットを送るのにダイアル・アップ伝送に於ける伝送当りのコストについて考えることができる。電話のシステムでは，電話をダイアルアップする限り通話がどれだけの長さでも 3分間コールの料金は請求されるわけである。もし 20 ミリ秒でも受話器をとっていて速座にこれを戻したとしても料金は 3分間として請求される。こうした特別の形のトラフィックの伝送に対する費用は計算することができる。

ARPAネットワークにおける計算法は，ネットワークユーザから得るトラフィック見積り量に基づくものであって，これは比較のために 1972年初頭にネットワークで採用されていたものである。

その頃我々はまだ16ノードのネットワークをもっていなかったので，その規模でのネットワークに対する数字は全く補外法によるものである。

質問 29

コンピュータ・ユーザーの実際の要求を考えに入れると，ARPAネットワークのようなコンピュータネットワークのオペレーションは，クローズドネットワークで行われなければならないものか。

或いは，現在多数の国々で研究開発されているディジタルデータネットワークのような，パブリック・スイッチ・ネットワークを使って将来，うまく運転できるようになるだろうか。

答

今日ＡＲＰＡネットワーク のランに於ける拘束は，それが商業用ネットワークではないということである。これは米国政府の研究用に使われ，政府の後援のもとに，大学と民間会社の研究用であり，商業用のものではない。

ゼロックスがカスタマーとのビジネスの為に使おうと思ってもネットワークは使えないが，ネッ

トワークに関する研究であれば使うことができる。我々の期待は，パブリック・スイッチ・パケット・ネットワークが米国で実用化されることであるが，その時には研究目的用としてパブリックネットからARPAネットへのアクセスが可能になると思う。

同様に，電話ネットからARPAネットへの接続も可能である。

しかし，料金に関しては現行の料金ではARPAネットを，商業用のものへと移行して使用することはできないだろう。現在，料金はかなり安くなってきているようである。AT&TはDDS設備に関して値下げしてきているし，これは我々が毎月，50キロビット／秒のものにマイル毎に5ドル程度の料金をTelpackに払っているものよりも安い料金になってきている。

料金の問題が解決すれば商業用のもの非商業用のものも共に接続できるようになるであろう。

質問30

Lockheed Sueを使った新しい高速IMPについて説明して欲しい。（本文参照）

答

Lockheed Sueについてはマニュアルがあるので参考にしていただければ良い。我々もLock-heed Sueのランを早くする為にハードウェアの内部を再設計しなおしたりして手を加える必要がある。技術的には，マシンの速度が遅くても良ければ問題はない。部品の関係で生産も大幅におちてしまった。従って我々の予定では1975年から使用することになっている。

質問31

TIPのレスポンスが遅く，使用するのにあまり便利でないということが言われている。この点についての所見を述べて欲しい。

答

TIPの問題点というのは，TELNET protocolであると思う。Youエコーや I エコーといった非常に簡単な考え方でエコーをリモートで行ったりローカルに行なったりする。仮に，TIPを衛星通信上で使うとすれば，TELNETコントロールのより複雑な型が必要となる。それは私が昨日説明したように，ローカルエコーやリモートエコーに必要な，"GET FILE" を用意しなければならないということである。もう一つの点は，TIPを設ける方法である。すなわち，IMPと同じCPUを使えば，IMPのプロセスのためにTIPが待機しなければならない状態によく落ち入ることである。別の問題としては大きなキャラクターサイズのファイル，例えば詳細な航空診断図（Aero diagnostics）のようなものを扱う場合のより大きなバッファーの必要条件についてである。ただ経済的な理由の為にこのようなデータを記憶させることがあり，こういう場合TIPでは記憶及びハードウェアーの面に関して非常に経費のかからないものにすることができる。しかし，ユーザーにとっては大へん使い易いというよりはむしろ非常に実用的なものというべきで，例えばユーザーが"L"を打ち込んでも，システムが，その"OGIN" をユーザーに打ち返してこない。というようなこともあって，このシステムは非常に倹約したシステムである。Recovery

procedures はこれ自体，全体的な問題であり，後程，NCC（Network Control Center）について述べるので，その時実際に詳細な問題を理解していただけると思う。

質問 32

計画中のユーザレベルのプロトコルとはどんなものであるか。

答

ユーザレベルのプロトコルは多数ありグラフィック・プロトコルは人々が今までしばらくの間作業を行なって来ているが，まだ正式なプロトコル（official protocol）にはなっていない。ファイル伝送プロトコルはここ数ケ月ないし数週間の間に変わっている。私自身その変化についての書類に目を通していないので，そのファイル伝送プロトコルについて今述べることはできない。

質問 33

現在のＩＭＰを新らしいＩＭＰに代えた時，ＩＭＰソフトウェアの変化は主としてどんな点であるか。

答

誠に面白い質問である。もともとこの高速モジュラーIMPの計画は，BBNが従来のIMPを通じてのリモートリロードの問題その他の多くの，修正を加えてゆく段階で考えられたものである。今では新らしいIMPを新しいネットワークに連結して，ゲートウエイコネクションの形で扱うことにほぼ決定したのである。いくつかのLockheed Sueを連結することができるが，あるものはゲートウエイ連結用で，あるものはHOST連結になるものとに分かれる。しかし本質的にはゲートウエイ連結によって，２つの別々のネットワークを連結することになる。この問題も1975年の予定にとり入れており，確実なことは言えない。一応現在の考え方を言ったまでである。

質問 34

ARPAのようなネットワークは，オンライン　ビジネスオリエント　システムとして使えるのか。例えばTIPのようなものを介して使用する。

答

返答に困る質問である。今まで私はビジネス界に必要な経済的な研究をあまり行なったことがなく，特にそれにかゝわる信頼性の問題などについては研究したことがない。一応私の想像を言わしていただこう。この種の業務はタイムシェアリングの形には適合するが，今まで高速コンピュータ同志のインタラクションというようなものを彼等は考えたことはなかった。私の考えでは，すべてのコンピューターを一ケ所に集中させるのは，とうてい不可能である。どんなことをしても，ビジネス中心のシステムで，すべてのマシンが１ケ所に集中することは出来ないと思う。しかし，お互いに内部連結をするという特徴をもたせたネットワークであればいくつも考えられるのであろう。そして経済性の点からいえば，連結による利用の方が，より有利であるとの意見を持っているが，この問題に関してはあまりデータを持っていない。

質問35

こうしたシステムでHOSTが近くに置かれる場合，星形ネットのシステムと比較してどうであろうか。

答

詳細については残念ながら答えられないが，ただ，距離はファクターではないということがわかったということは言える。

つまり，もし，ARPAネットが1つの都市に全て集中しているという場合，たとえばWashing-ton DCに集中していると考えてみよう。この場合，ARPAの 技術を使ってのネット建設のコストは，事実上米国内全土にわたってネットが敷かれる場合と同じである。つまりネットがある規模に達した後は，回線のコストはその距離の長さによるコストではないのである。

星形のネットワークに関して云えば，まずその信頼性はどうか，またつぎに，星形ネットワークに於ける回線数はARPAネットの場合と殆んど同じでなければならないといったようなことがあると思う。

星形ネットワークは基本的には各ノードにつき1つの全二重回線を持っており，ARPAの技術に於いても同様であるが，ARPA技術の方がたぶん信頼性が高いであろう。また星形ネットワークの方が全部をデュプレックスにしている場合には信頼性に於いては同等かもしれないが今度は経済的にARPAの方がすぐれていると云えると思う。しかしこの2つの技術に関する徹底的な比較を試みたことはない。スループットよりもむしろ信頼性の方が重要だと思う。

またHOST機に入ってくる回線上の多重データについての話があるがこれには多数の利点がある。というのは大型機や大量データを使用したいユーザーにとって彼のコンピュータシステムでのインターフェイスが大へん高くなるということがある。そしてもし，百の異ったポートエントリーを与える方式と，1個の多重ポートエントリーを与える方式とどちらを選ぶかという時には，後者の1個の多重ポートエントリー（multiplex port entry）をとった方がよりスピードのおそいエントリー（外部で多重が行われる）よりもはるかに安い。たとえばILLIACではそのシステムの各接合点につき，およそ5万ドルで実際にはソフトウェアのコストを考えに入れるともう少し高くはなるが，どこのシステムからでもよいから，エントリーの1点を持つことができればこれはただ経済的な観点から云って非常にのぞましいことである。

質問36

新しいノード或いは加入者（Subscriber）がARPA ネットワークに加わった場合には，どのような修正がIMPおよびHOSTソフトウェアに施こされなければならないであろうか。

答

HOST システム内に於ては簡単にいえば，修正を要する部分はただその名前をシステムにつけることだけである。これはHOST システムのユーザーは番号ではなく名前でその行先を表わす

からである。ユーザーは例えばMITに結合したいという様に指示するであろう。したがってシステムはMITの存在を確認し且つマッピングをしておくことが必要である。これはちょうど電話のシステムで，実際にコンピューターに電話番号調べをさせるようなもので，だれもが名前でリストアップされているが電話番号まで憶えることはないわけである。このようなマッピングが自動的におこなわれるが，場所の名前は入れておかねばならない。名前さえわかればだれでもその場所の番号を使うことができるのである。

IMPで変更しなければならない部分は，定期的に，たとえば毎年一回，テーブルの大きさをエントリーに対してさらに多くの収容場所をつくるために少しづつ大きくしていく必要があるということだけである。たとえばこれまで，説明してきたルーティング・テーブルは64のエントリーを収容できる場所をもっており，まだ何年間かはたぶんネットワーク操作上不都合なことはないと思われるが，ある時点でエントリーが64に近い数になったところで，この数を80或いはおそらく100にしたいときがくるだろう。その時には，ネットワークに対して何らかの内部変更が行われるであろう。ある日突然，内部にさらに多くのテーブル・スペースを持つ新しいシステムが発表され，だれもそれに気づかないだろう。つまりユーザーにとってはそれは全く眼に見えないものであるからだ。現在のネットワークの構造はどうなっているかといえば，ノードは7つまでコネクションを持っており，HOSTや回線に接続されている。新しい回線を機械につなぐと，機械の方で新しい回線の接続を自ら確認しており，新しい回線が接続されたということをわざわざ知らせなくてもいいわけである。

また回線をはずすときも同様であって，ネットワークにそのことを知らせる必要はない。

そこでもしAT&Tが何かのわけがあって，ある回線を使う必要のある場合，通常それらの1つの回線をとりはずすという知らせがあればよい。しかしその時には新しい同期の手続きが必要となる。回線はいつでもすきな時に切ることができるし，ネットワークはただ1秒，内至1／2秒位の間に適応化される。

IMPに接続される新しい回線に対してはインターフェイスの設置が必要となってくる。この装置は約5000ドルである。

この装置は7回線まで収容可能で，その範囲内なら追加は簡単である。

質問37

パスワードのチェック或いは会計目的の為に，どのように加入者（Subscriber）がARPAネットワークに登録されるのであろうか。

答

現在は必要な登録はHOSTシステムの中に入れるものだけであり，パケットが最初に入ってくるときARPAネットワークの入口でユーザーのチェックを行うことはしない。ARPAネットにおけるその目的がデータ通信そのものにあるのではなく，むしろコンピュータシステムをネットワーク

を通してアクセスすることにあるのだから，コンピュータシステムの中に入ってくる時にチェックが行われればよい。ネットワークにユーザーが入ってきて，そのデータが終日流れているとすれば，それを抑制するようなことは何もしないで，流れているトラフィックの量を記録しておくのである。もしそのトラフィックの入ってくる場所が有料ならば，支払請求が行われる。

ユーザーのデータは，HOSTシステム内に入ってくる時にその名前を明らかにしなければならない。Logプロセデュアを経て入って来るが，その時にはパスワードを確認する必要がある。他のコンピュータを使いたい時にはそのコンピュータに対するパスワードを持たなければならない。従ってネットワークに入ってきて何かをやろうとする場合，ユーザーは，たぶん会計目的の為に全体のシステムに対するパスワードを持とうとするだろう。パスワードのチェックは少量のメッセージがあるコンピュータに入っていき，ユーザーの確認信号が返ってくるわけであるが，もしユーザーが公認されていないものであった場合は，このメッセージは少しの間そのままの状態におかれたままで，ユーザーはシステム内に30秒間程度は入っていられるかもしれないがおそらくどの設備も使用することはできないだろう。

ユーザーを公認するには，どのシステムを使うべきなのかをユーザーに告げ，ネットワークその他についての30秒間分のニュースを提供することになるだろう。その間，ユーザーは何もできず，もし正式ユーザーの公認がされない場合は，その使用ができないようにするというものである。このような機構はまだネットワークがこれを必要とする程大きくないことから実際に存在しているものではないが，この機構を導入する考えであるし，民間のネットワークでも必要であることは明らかであろう。

質問38

回線交換型IPC（Inter Process Communications）と比較して，ウォルデン（Walden）氏論文によるメッセージ交換型IPCについてどのように考えるか。このメッセージ交換型IPCを実施する計画はあるか。

答

ネットワークの開発及びその応用について見てきたが，殆んど全ての面で初期の開発においてどの部分が長所であり欠点であったかということを見てとることができる。どの場合にも最初の頃はむしろ非常に複雑な設備をつくるよりも簡単にできるものを手がけていった。精巧な保護機構ではなく簡単に本当に必要なものだと見做したものだけでスタートしたのであった。

HOST プロトコルに関しても同様に改良の可能性のある領域はあるし，現在のHOSTプロトコルがどんなところで相互のネットワーク構成と関係しているのだろうかということが非常に真剣に注目されてきている現況である。将来どのようなプロトコルが実施されるかが基本的に決定されようとしていると私は思っている。私の今，書き終えようとしている論文があるが，ハワイ会議でそのアブストラクトを出したのだがこの論文の中で，種々のパケット交換型ネットワークが各ネッ

ト間で相互接続できるようなプロトコルについて述べている。

しかしこのプロトコルは，ネットワーク間のパケットレベルと同様HOSTレベルにあるプロトコルである。基本的には互いに交信している両者の間のHOSTシステムの問題を扱わずにパケット・ネットワークを相互連結するという問題を取り扱うことは基本的にはできないようである。その理由は，パケット・ネットワークが異った内部構造を持つことを可能にするためであって，ネットワーク全体を，ある固定されたパラメータ設定のもとに標準化するということのためではない。これは基本的なことで1つの最初の事例であると思う。

非常に便利ではあるが，パケット・ネットワークを相互に連結するということは，これがHOSTにとって有用性があることが必要であり，そうでなければ，それらのインフォメーションを全部もう一度適当に順序づけして返すということはできないであろう。

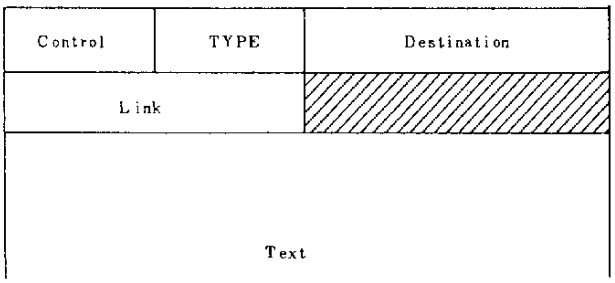
質問 39

IMP間或いはHOST間の通信中に何か故障が起った場合または，メッセージ，パケットに何らかのフォマット・エラーが検出された場合に，IMPからHOSTへのエラー・メッセージをどのようにリポートするのか。

答

問題点を2つに分けて見てみよう。1つはユーザーに関係するもの，もう1つはネットワークそのものに関するものである。

HOSTとIMPがあった場合に両者の間にハードウェア・インタフェースがあるということについては先述した。通常のHOSTのインタフェース，ディスタントHOSTのインタフェース，VDHのインタフェース等々。それらの間に12本程の回線があって，メッセージが送られる。



TYPE： 0 – Regular (H→I , I→H)
– RFNM (I→H)
– Incomplete Transmission (I→H)
– Error (H→I , I→H)
– IMP going down
– HOST going down

IMP/HOST protocol で処理

Fig 21

メッセージのフォーマットはFig 21で示した。この中の8ビットのリンク情報は基本的にはネットワークで使われるのではなく，HOSTで使われる。またタイプを表わす4ビットと，コントロール用の4ビットがある。タイプを示すビットのうちタイプ0は通常のメッセージで，あらゆる通常の伝送に使われ，HOST間を送られるメッセージである。従ってHOSTが通常のメッセージを送る場合，基本的に最初の8ビット全部が0のメッセージとする。

他のタイプのものにRFNMがあるが，通常のメッセージがHOSTとIMPの間を往復できるのに対してこのRFNMはIMPからHOSTに向う一方向にしか伝送できない。もう一つ別のタイプは不完全な伝送である。これもまたIMPからHOSTへの一方通行しかできないものである。別のメッセージとしては，一連のエラー・メッセージがあるが，これは両方向の通行ができるものである。また，IMP或いはHOSTがダウンしようとするのを知らせるメッセージがあるが，これのデータ・フィールドはその周囲の状況と時間の状態についてのインフォメーションを与えることができるものである。

ネットワークのオリジナル・バージョンに於いてはリンクフィールドに関係するメッセージがたくさんあった。たとえば，リンクメッセージがブロックされたときにそれを知らせるメッセージが返ってくるが，そのリンクテーブル（サイズは64である）が満員であると知らせるメッセージもあるが今では使用されていない。基本的にはこれはIMPとHOST間を結んで往き来する基本的に大事なものであり，種々のタイプのメッセージによってその役割が果たされている。

これはソフトウェア・プロトコルであり，一般にIMP－HOSTプロトコル或いはHOST－IMPプロトコルとして知られている。多重プロトコルのうち最も低いレベルのもので，ネットワーク内，基本的にはHOST内部に存在するものである。

質問 40

ソケット番号について説明して戴きたい。

答

ソケット番号というのは特別のコネクションがネットワークによって指定される方法であり各HOSTが，その独自の方法で指定することができるようになっている。TENEXに於いては，JOB番号に関連して付けられている。あるいは，それを指定する内部の識別子といっても良いと思う。又，恐らくある種のファイルハンドルに関連しているのであろう。どのように指定するかというのは大切なことではなく，データが送られて来た時，そのデータに関連して独特の方法で指定できるということが大切なのである。ある場合には，運用中のプログラムに対し，内部表現があり，NCPが内部表現を適当な長さのソケット番号に変換するのである。また，NCPが，モニターからソケット番号を得ることもある。これはマシンによってやり方が異る。ソケット番号は，言わゆるどちらかの端に於てデータに対してその識別の役目を果すものである。従って一方の端と他方の端に一つずつソケットが設けられてある。

質問 41

通信衛星の transponder の信頼性について少々疑問視しているのだが，これは active なのかそれとも passive なのか？

答

基本的には，ただtransponder といった方が良く，aetive あるいはpassive と付ける必要はない。すなわち，一つの周波数の信号を取り入れ，違った周波数に変換して，それからは違った周波数バンドで正確に信号を送るのである。この意味からすれば，passive である。しかし，増副器と他の装置があり，信号を送るのに強化するわけでこの意味からすれば，active だと言える。

質問 42

通信衛星 transponder を使えば，ネットワークのすべての形状が星形になってしまって当初の段階に戻ってしまうのではないか？

答

確かに中央部があって，すべての結合がその中央部に集中すると言う意味では，スターネットワークによく似ている。通信衛星の信頼性は今まで非常に良好であった。私が今までみてきた各種のコミュニケーションシステムのどんなものよりもその信頼性は，はるかにすぐれていた。ただ通信衛星 transponder に関して失敗が１つあったのには気付くことができなかった。例えばハワイへ送る回線上で，一時アンテナが正常に働かなかったことがあり，再送をしなければならなかった。そこで，我々が実際に考えたのが，マルチプル通信衛星チャンネルを使うことでしかも費用が安いということである。２～３のチャンネルが，アンテナを通じてサービスを供与するわけである。仮に，例えばアンテナが30フィート以下のものであれば，大きなアンテナ程，ひんぱんにトラックを行なう必要はない。トラックを行なう回数が少ない程良いのである。しかし最終的には，我々としてはこの方法にかけるということはしない。ある意味では，容量がこの方法によって大きくなったことは事実である。容量の小さいものは，おそらくまだ，地上での連結によって行なわれるであろう。恐らく，1.5 Mb の連結，50 Kb の連結であろう。従って総データー容量も効果的なものではない。そこで，地上での容量と通信衛星の容量との間の最高容量を調整する場合がある。費用の面でも，地上回線よりも通信衛星回線の方がはるかに安い。また我々の経験からして，通信衛星リンクに対する不信感は地上回線から出てくるものである。商業サービスでは，良きにしろ悪きにしろ，どうなるのかを見守らなければならないと思う。

質問 43

ネットワークの使い方で，ユーザーの観点からの典型的な使用方法はどのようなものであるか。

答

ネットワークを使用することで，まず主となるのは，バッチ処理の環境でプログラムを作成したり，そのプログラムを実行させたりすることである。私が今まで使ってきたどのタイム・シェアリ

ングマシンも，離れて業務が行なえるRJSという能力を備えている。従って，ユーザーはターミナルに居て，タイム・シェアリング・マシンを使ってプログラムを書き，あるいはプログラムを編集したり，つなぎ合わせることができるのである。そして次にRJS能力を利用して，どのマシンにプログラムを送りたいのかを指示すればバッチ操作としてそのプログラムが送られ，その返答が戻ってくる。それを今度はどこかでファイルにまとめたり，コンソール上にプリントしたり，プリンタに出したり，さらに次の処理をする為に，次のマシンに送ったりすれば良いわけである。

異なった設備を，ある方法で非常に経済的に使ってきたユーザーがいる。特にすぐれた例は，プログラムを減らしたものである。この方法はユタ州のTony Heron氏が行なっていたものである。それはmathematical simple manipulation prog．であり，プログラムを違ったマシンへかけられるようにしたものであり，IBM360−91の下で作ったようである。また，LISP instructionを他のLISP instructionに変えるものもある。PDP−10ではこれが可能である。LISP言語で書いたプログラムを他の違ったマシンに送り，実行時間および経済性などの点で，最適のマシンを選び実行させるわけである。天候データーを扱っているユーザーはILLIACを使用して，かなり多くの天候データーを処理しているケースもある。ここではかなり複雑な変換タイプの計算が扱われる。

私自身はネットワークを計算よりも経営用として使おうとしている。ネットワーク上で我々のファイルをすべて扱って経営面に役だたせるわけである。これもビジネス界でネットワークを運用する全く違った方法である。これには終始オン・ラインのエクスチェンジが必要である。最初の連絡から1時間の間に起こってくるのは，まず，文書を読解し，寸評を作成し，訂正があれば行ない，再び提出し，また提出するといった具合である。我々の場合，ネットワーク上で文書を作成すれば良いわけである。よくFrank氏とKleinrock氏と共に文書を作成したものである。2〜3週間ばかり行なったがその間にはかなり頻繁なやりとりがあった。Kleinrock　氏はカリフォルニアに，私は東岸にいて，同氏が，ある文書を書き，私がその文書をネットワークを通して読んだのである。時々，同氏が文書を書いている時，私はすわって彼が書く文書をネットワークを通じてみていた。そして私が気にくわないからと言って訂正を加えていったものである。そうすると彼の方は，"変えるから戻して欲しい，君が変えたものは気に食わないから。"と云ってくるのであった。ファイルを作り出すのに実際，オンラインの相互作用がみられる。方程式を書いたり，それを操っているのと同じようなものである。長い文書の場合，郵便よりも伝送した方が早い。しかし実際には，まだ一度も伝送したことがないのである。文書はかなり長文であったし，一台のマシンで書くという状況であったからである。時間内にその文書を郵便で出すのも高くつくし，ネットワークを通じて，bitで送るのも高いわけである。そこで私はその文書を電子的にスキャンし，変化している部分を読み，そのパラグラフを抜き出し，それをみることにした。そうすれば時間内で全部処理することができる。今やネットワークを通じて我々のもつメッセージのトラフィックをすべて扱える

わけである。私の事務所で行なう連絡の90％がカリフォルニア経由でネットワークを通してやっているものであると言える。面白い例としては，私のところから20分程離れたところにいる友人に"昼食を一諸にとりたい"というメッセージをカリフォルニア経由で彼に送ることがある。そして彼の返事が"ＯＫ"とでたなら私は彼の事務所を訪ね，昼食に出かけるのである。

ある種のトラフィックは，ネットワークのない場合には決してつくりだされることがないものである。事務管理用のトラフィックを扱うのにネットワークを使ってやると，他の機構を使って行う場合よりはるかに便利であるということはしばしば経験するものである。

私自身，朝，出社するとふつう30部程度の事務書類があり，私はそれをネットワーク内で処理するわけだが，それが終るまでにさらに15部置かれてあり，こんどはそれをすませるうちにさらに5部，そしてそれが終るとランチタイムとなる。ランチから帰るとさらに20部の書類が置かれてあるというわけである。

もう1つ非常にいいことは，どこで旅行していてもちゃんと仕事をすることができるということである。

日本はその形状においてかなり細長い距離をもっているが，その通信網についてはどうだろうか。おそらく端から端までかなりうまく配置されていて，どこにいてもその現在位置が明確にされるのは我々が米国のどこかにいても通信の届かないようなところがないのと同じようなものであろう。

ネットワークが切れている場合，ネットワークと対話しようとすると一種の無効の頭文字である"dead"か，"HOST・デッド"の信号が返ってくる。それでも，ある人がどこにいるにしても彼はターミナル或いは何か他の機構を通してネットワークと連絡をとることはできると考えられる。もしネットワークの応答があまり速くないときはこのネットワークでは応答活動がないと云ってもいいだろう。

我々は米国内にいる限り，国内を移動している場合，どこの地点からでもネットワークを使えるということは非常に便利なことである。どこにいても質問ができるし，事務的な問題も処理できる。どこにいようとあらゆる一連のファイルを持っているということになる。たとえばファイルに収められている技術的問題に関する質問がでてきた場合，すぐにそのファイルに目を通して，ハワイ，カリフォルニア，メイン，フロリダ等どこにいても，その解答を与えることができるというわけである。

すべて同じことで，私は自分のすきなところへどこにでも事務所を移すことができる。ちがうのは気候ぐらいのもので他のちがいは全くない。従ってこれは我々に対して大きなインパクトでありこのことに非常に力を入れて研究しているわけであるが，たとえ我々が懸命に働いて，非常にお金をかけてやってもまだまだこれは芽の出かかった初期の段階であり，実際に堅実な電子的基盤を築きあげる迄には相当の仕事が必要であることを痛感するのである。ひとたびこれをやれば，実態は，はるかに偉大な能力の発展に向って進むだろうし，現在非常に実行が困難であるとされているある

種の機能の向上も行われることだろう。

同様にいろいろとお話できるような他の例もたくさんあると思う。そのうちのいくつかについてはすでにのべた。

種々の場所にファイルを送ったり送り返したりするという点で他の例をあげると，種々の機械の研究者によって書かれている多くの種々のプログラムの試みがあるのだが，言語認識の分野の努力がなされてきている。そこでは全体の問題にわたって取り組むには専門技術がまだ不足している。種々のグループが，この問題を部分的に研究し，お互いに他のグループの開発している部分との関連づけを試みているので，このプロジェクトが終るにはこれらの部分研究がすべてネットワークに関する共同作業となり1つのシステムとなることについては，たとえそれが現在全ての部分に対して責任をもつグループあるいは，理解するグループが全然ないとしても，希望のもてることである。従って各部分の研究がよく理解されておらずまた普及もされていないが，全体のシステムは活用的なこの種の研究プロジェクトをさらに我々の手で進めていくことを私は期待している。

COST vs. DELAY

FOR POTENTIAL 20 NODE NETWORK DESIGNS

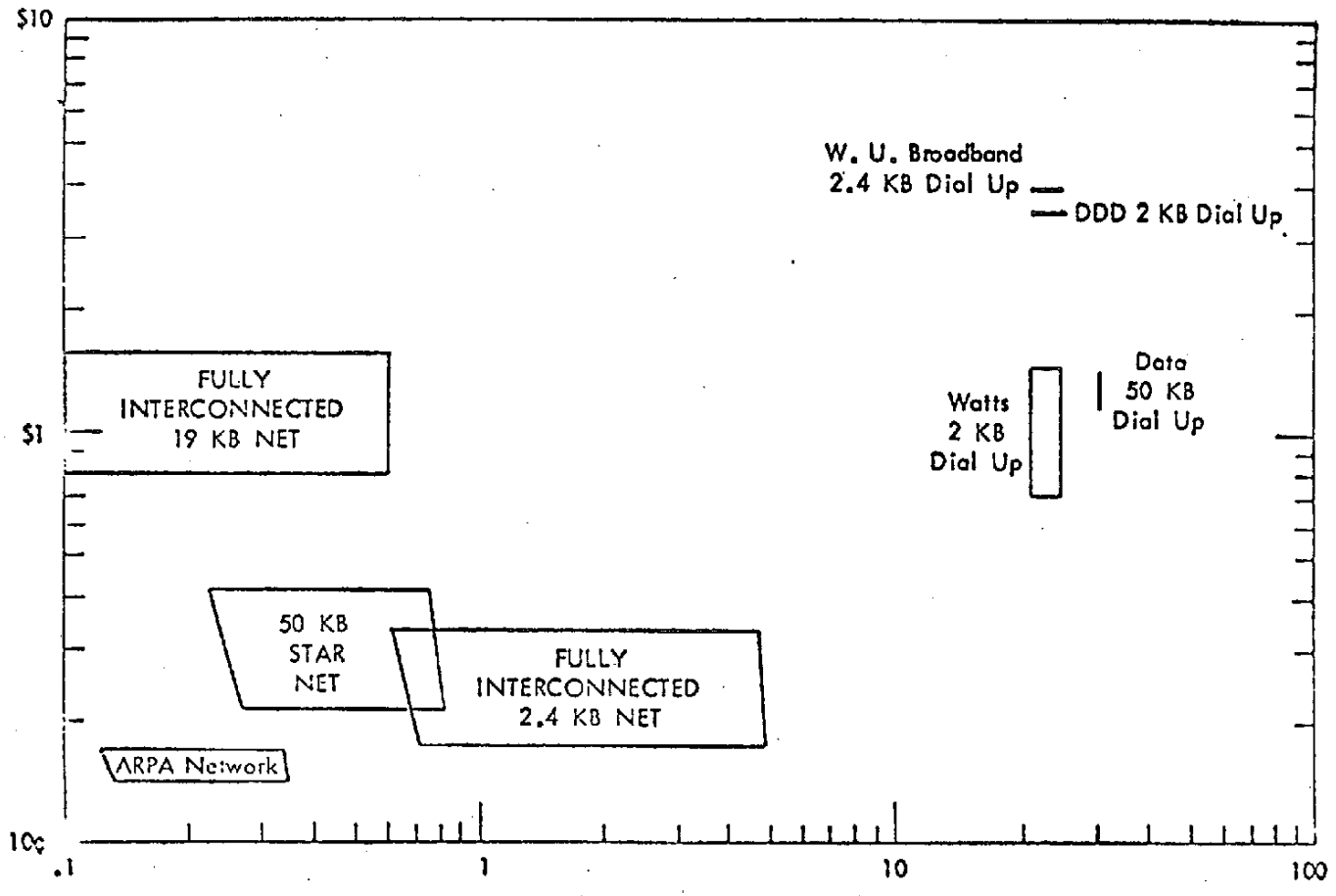


Chart 1

EFFECTIVE BANDWIDTH vs. BLOCK SIZE

TWENTY NODE NETWORK

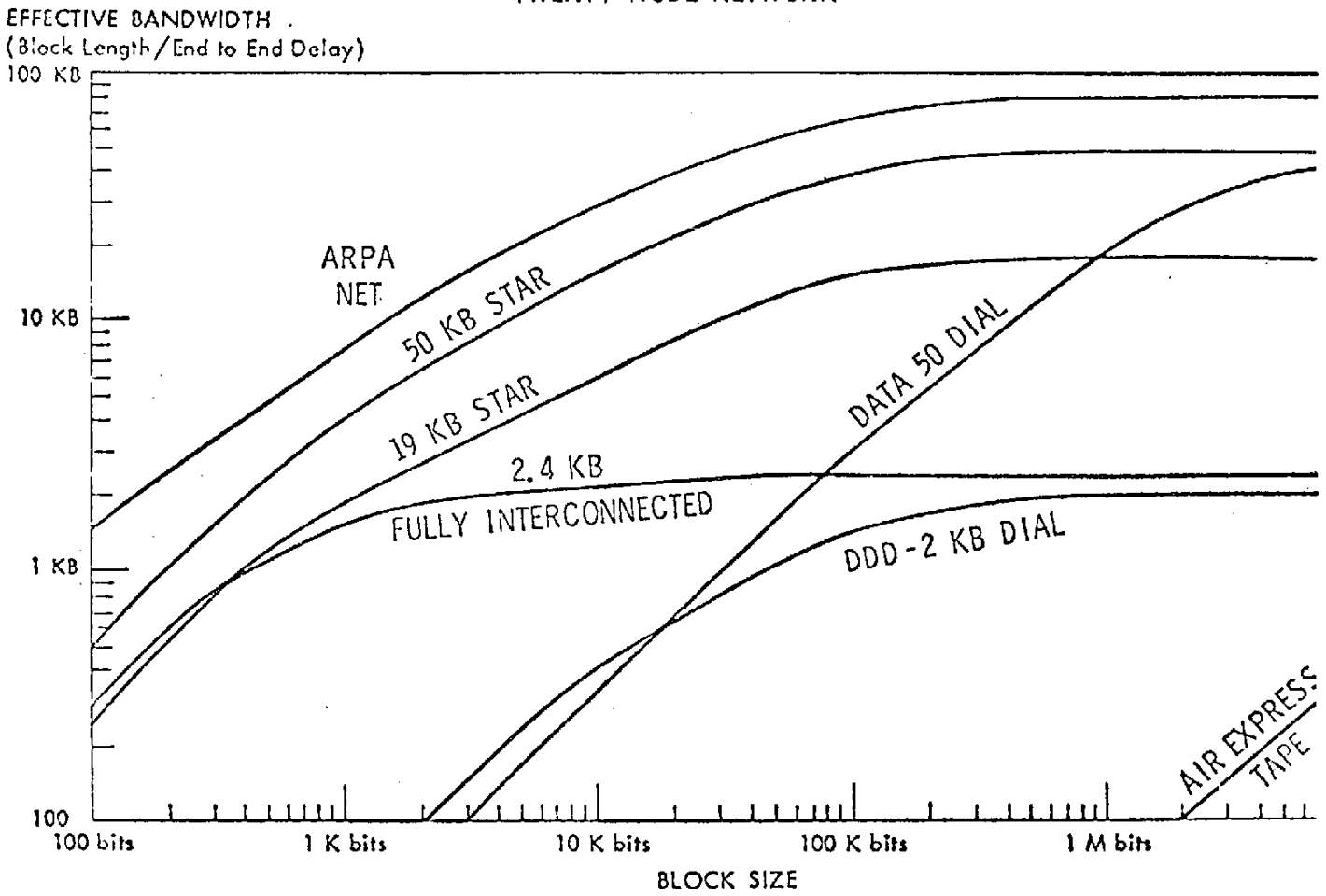


Chart 2

OVERVIEW OF NETWORK

Over 30 geographically distributed and inhomogeneous computers.

Interconnected via a wideband packet switching network.

Resource Sharing		Allows the full set of network resources such as software, storage, etc. to be effectively shared by the whole network community.
Cost Effective	-	$.30/megabit estimated communications cost.
Reliable	-	Requires multiple circuit failures to disconnect.
Error Control	-	Undetected packet error rate of one in 10^{12} or less.
Responsive	-	Average packet transit time under 0.2 seconds.

Chart 3

ARPANET MESSAGE FLOW

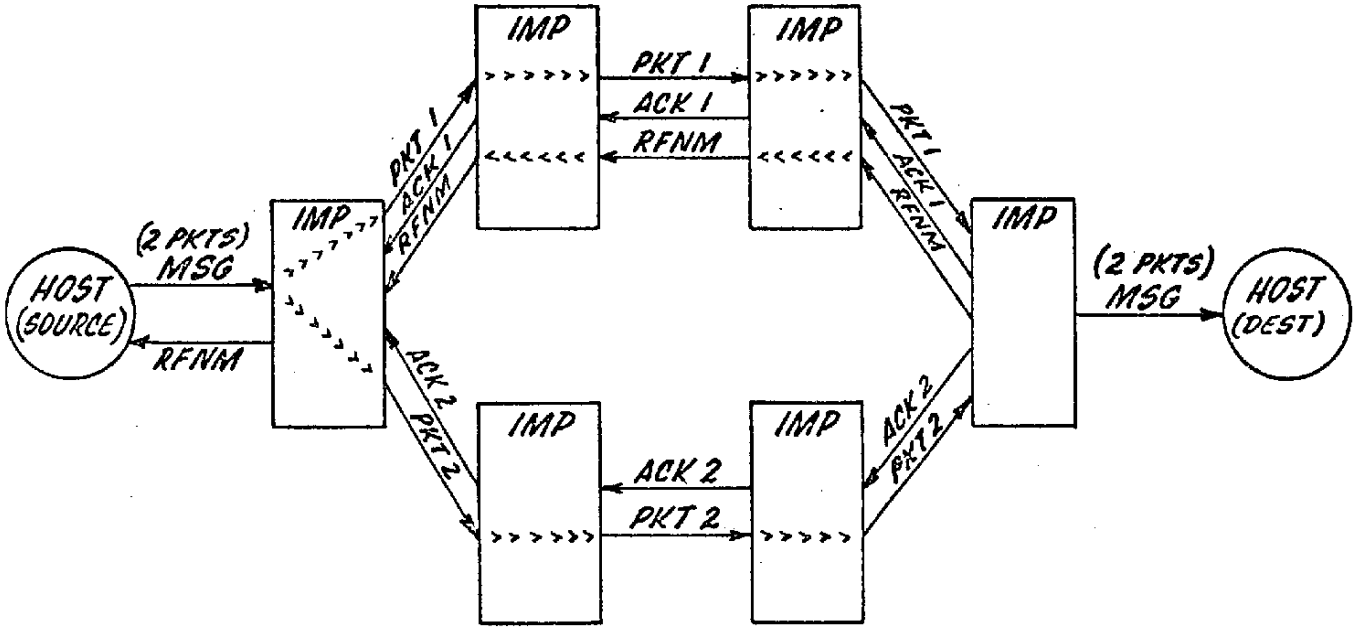


Chart 4

ARPANET MESSAGE FORMAT

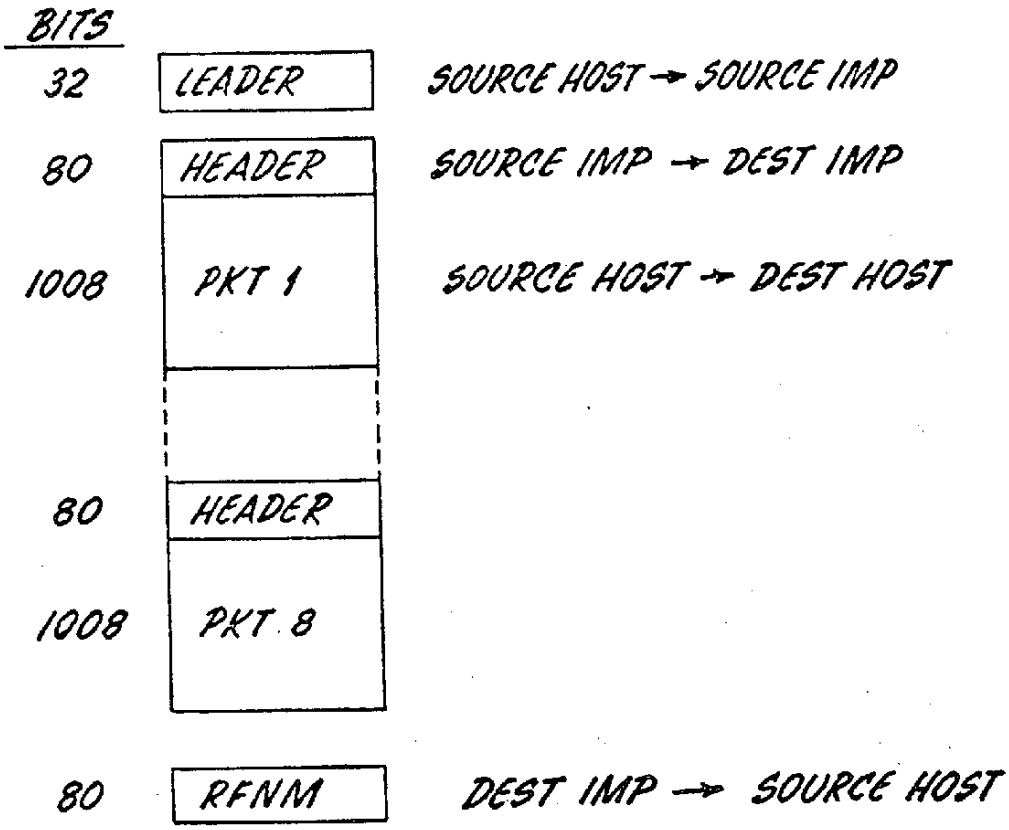


Chart 5

EVOLUTION OF THE ARPANET

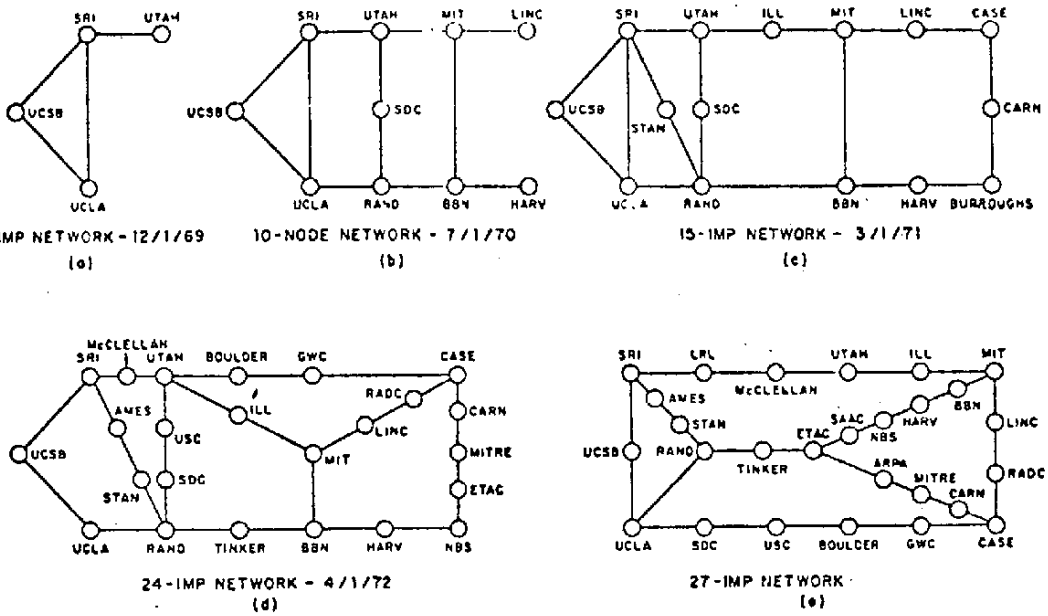
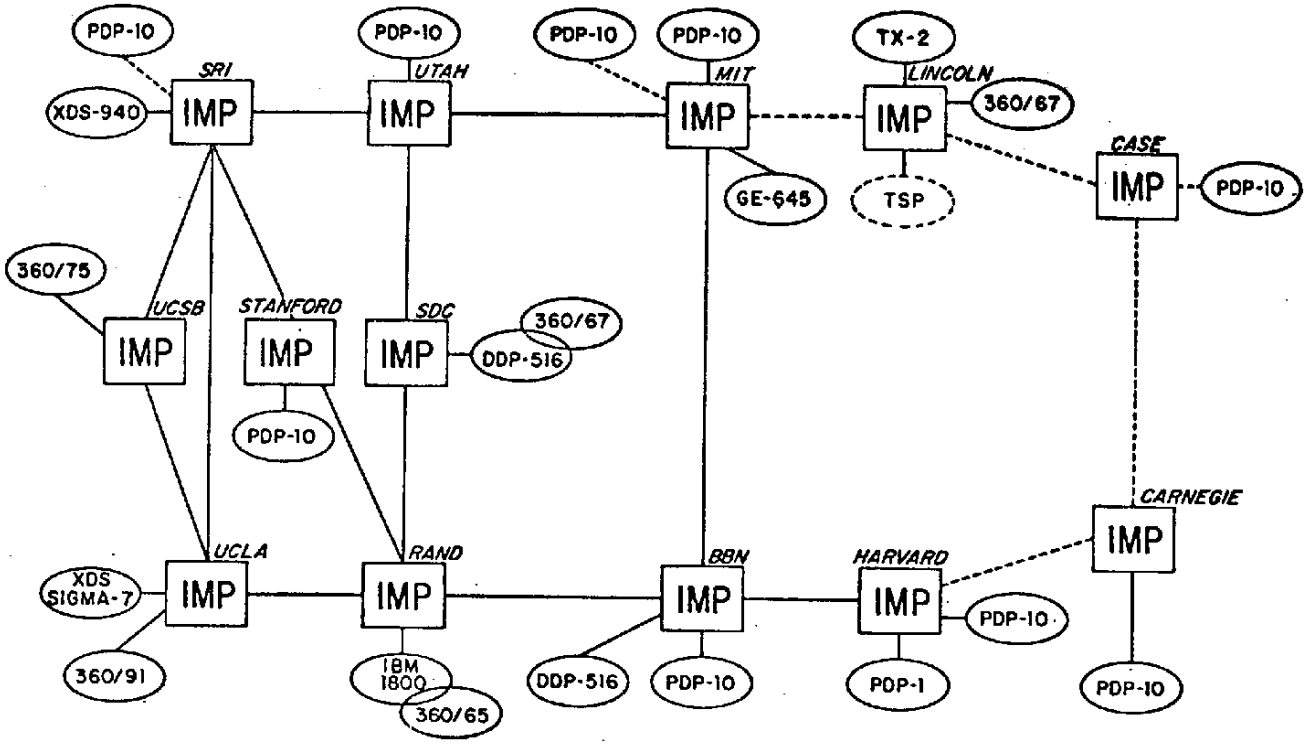


Chart 6



ARPA NET, DECEMBER 1970

Chart 7

ARPA NETWORK - JAN 1973

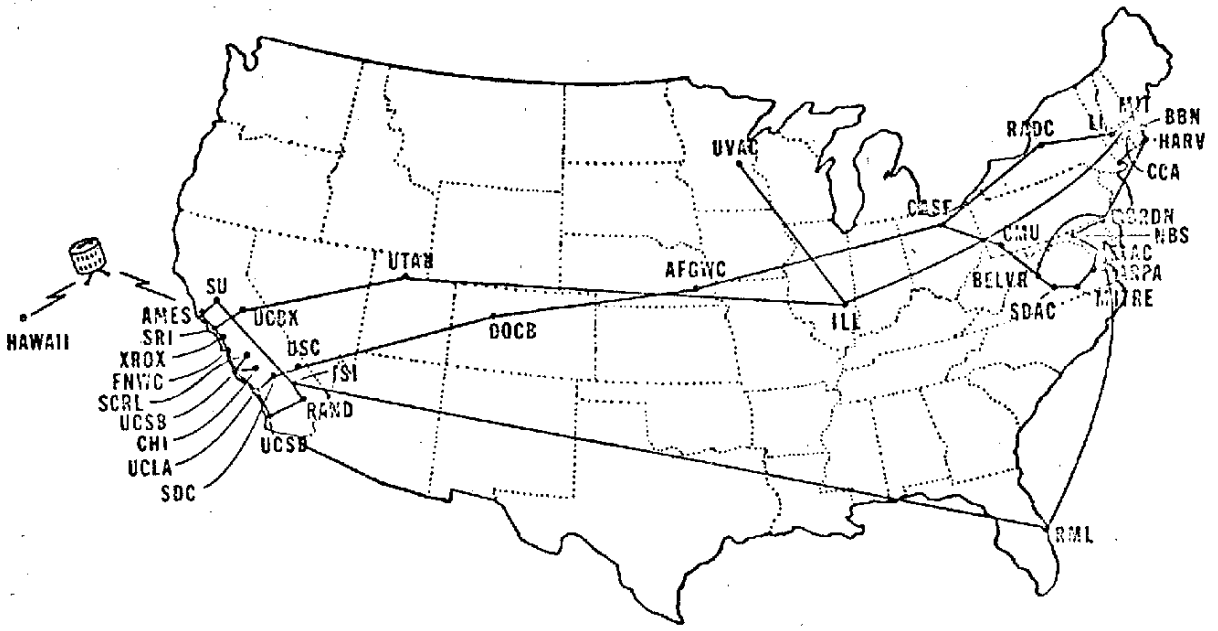


Chart 8

ARPA NETWORK, LOGICAL MAP, SEPTEMBER 1973

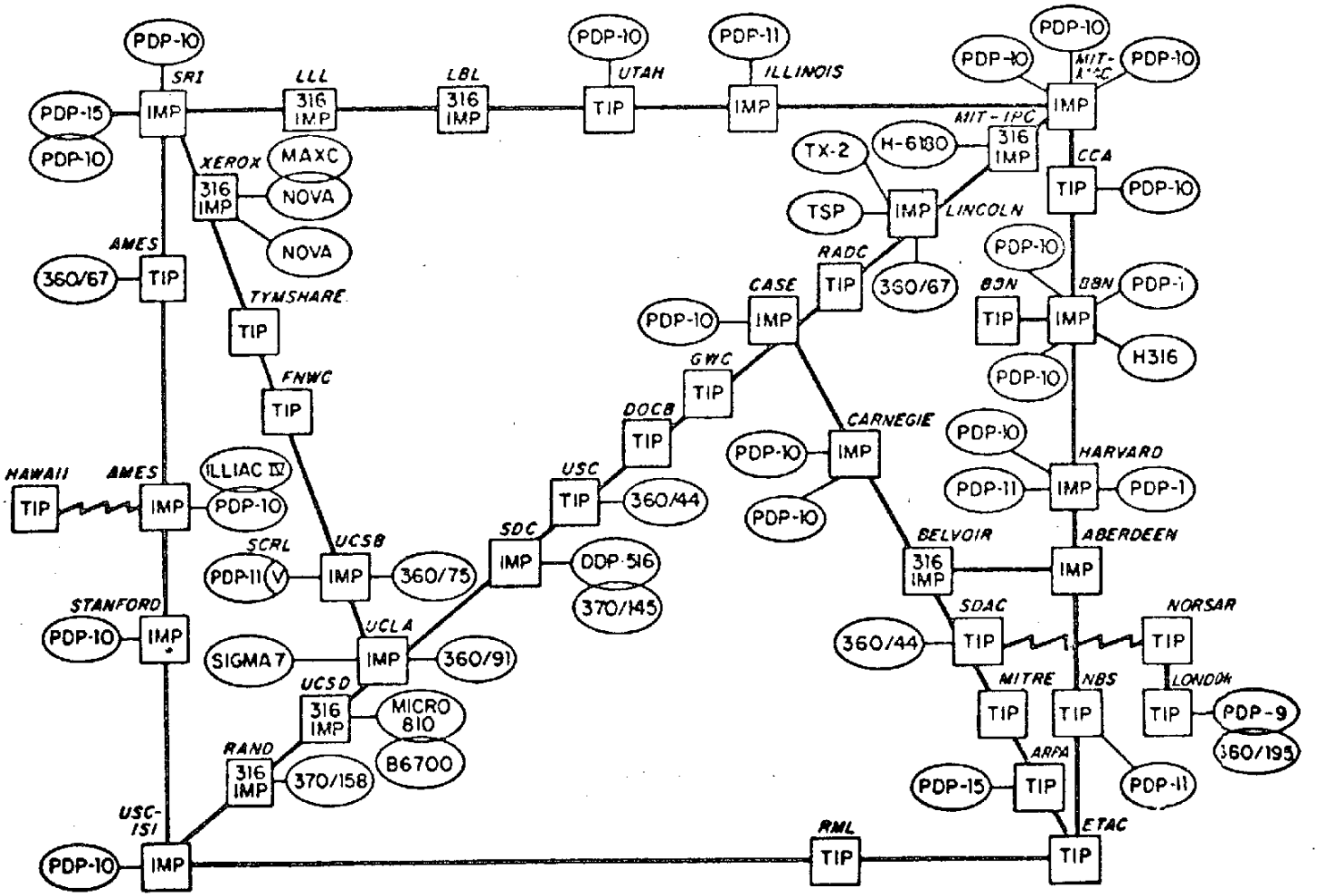


Chart 9

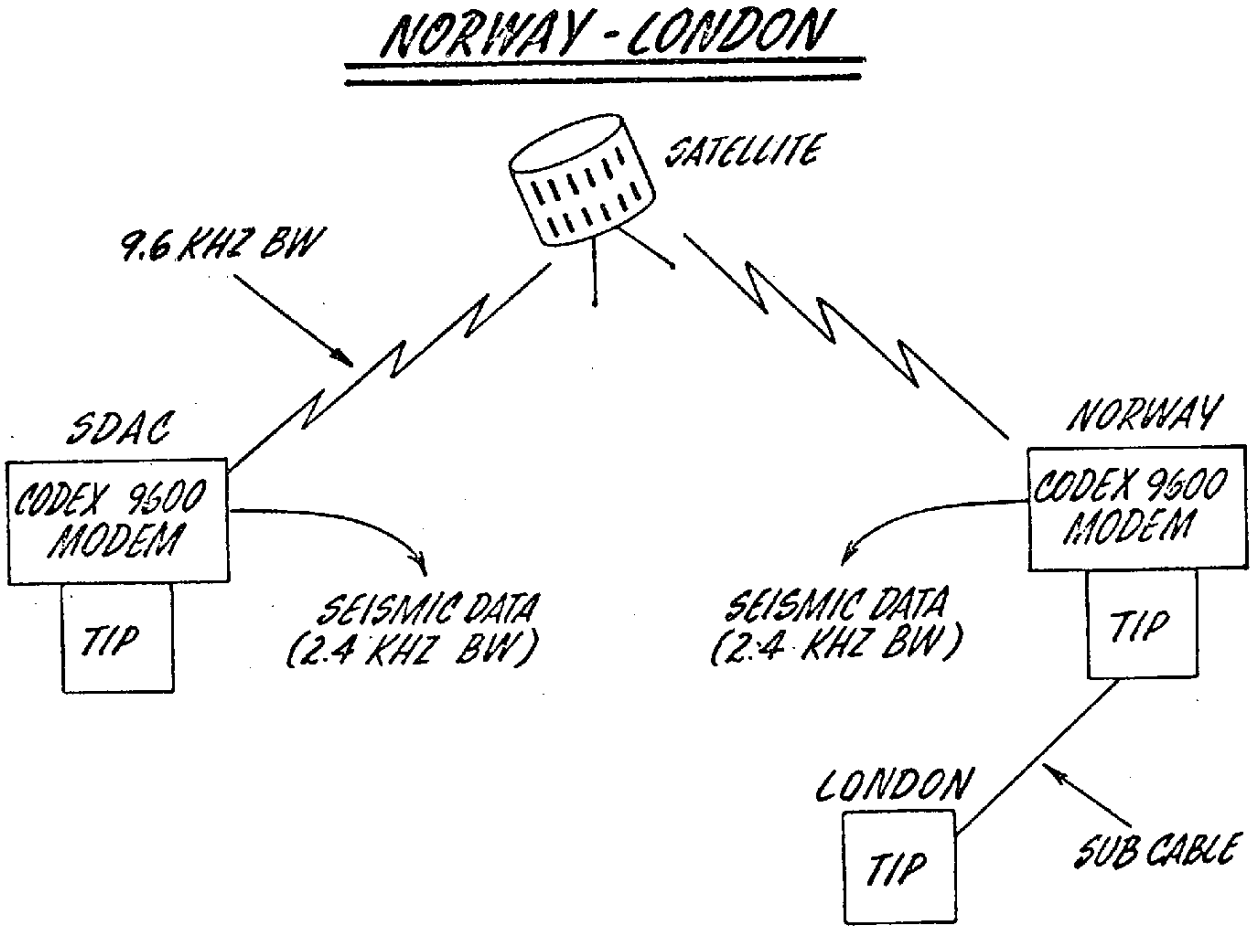

NORWAY - LONDON
SATELLITE
9.6 KHZ BW
SDAC
CODEX 9500 MODEM
TIP
SEISMIC DATA (2.4 KHZ BW)
SEISMIC DATA (2.4 KHZ BW)
NORWAY
CODEX 9500 MODEM
TIP
LONDON
TIP
SUB CABLE

Chart 10

COMPUTER RESOURCE USAGE WITHIN ARPANET

Annual Remote Computer Usage Cost Based on March 1973 Data

User Organization	Activity	Remote Usage ($ in thousands)	Projected Cost for Local Replacement
University of Illinois	Parallel processing research	360	1100
NASA Ames	Air foil design and ILLIAC	328	570
Rand Corporation	Numerical climate modelling	210	650
Applied Data Research	ILLIAC IV compiler development	151	470
Lawrence Livermore Lab	Dev of TENSOR code on ILLIAC	94	370
Stanford University	Artificial intelligence research	91	180
Rome Air Dev Center	Text manipulation and resource evaluation	81	450
ARPA	On-line management	77	370
Seismic Array Analysis Ctr	Seismic data processing	76	300
Mitre Corporation	Distributed file network research	60	240
Natl Bureau of Standards	Network research	58	200
Bolt Beranek & Newman	TENEX system support	55	80
Xerox Parc	Computer science research	47	100
USC-IPL	Picture processing research	35	70
UCLA	Network measurement	28	90
Systems Control, Inc.	Signal processing research	23	70
UCSB	Network research	22	70
Range Measurements Lab	ARPANET management	17	60
Institute for the Future	Teleconferencing research	13	40
Miscellaneous	Computer research	192	580
	Total	2018	6060

Chart 11

TRANSIT TIME & MESSAGE RATES

	Minimum	*Maximum*
SINGLE WORD MESSAGE		
Transit Time	5 msec	50 msec
Round-trip	10 msec	100 msec
Max. Message Rate/Link	100/sec	10/sec
SINGLE FULL PACKET MESSAGE		
Transit Time	45 msec	140 msec
Round-trip	50 msec	190 msec
Max. Message Rate/Link	20/sec	5/sec
8-PACKET MESSAGE		
Transit Time	265 msec	360 msec
Round-trip	195 msec	320 msec
Max. Message Rate/Link	5/sec	3/sec

Chart 12

MIN. ROUND TRIP DELAY VS. MESSAGE LENGTH

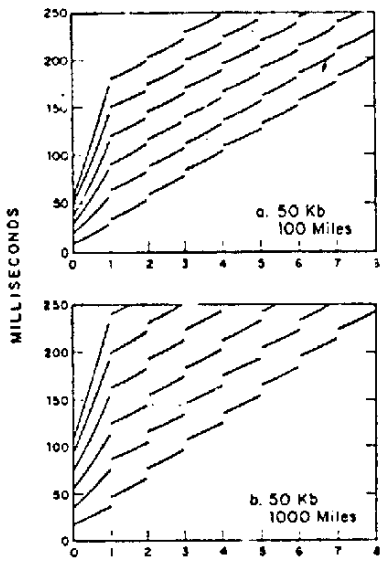


Chart 13

BENEFITS TO ARPA OF THE NETWORK

OVERALL PERFORMANCE OF SERVERS IS BETTER
- Service is more visible
- Network users free to select other systems

INCREASED INTERCHANGEABILITY OF SOFTWARE, STORAGE, ETC.
- Results from more uniform, standardized installations - due to network interactions

IMPROVED SYSTEM DEVELOPMENT FACILITIES
- The best machine can be selected without regard to geography

REAL TIME DISTRIBUTION OF NEW SYSTEMS AND MODIFICATIONS
- Avoids delays and errors - convenient distribution via the Net

ABILITY TO USE SYSTEMS EVEN WHILE UNDER DEVELOPMENT
- for research purposes only - can dramatically reduce the time needed to transfer new concepts

ONLY PRACTICAL INSTRUMENT FOR SHAPING UP COMPUTER CENTERS

REDUCED REQUIREMENT FOR SYSTEM PERSONNEL
- Increased cooperation among sites saves at least one person per TENEX site
- Cooperation provides increased overall leverage in solving system problems

Chart 14

NETWORK SUMMARY

MONTH	LINE OUTAGE	IMP DOWN ALL* CAUSES	IMP DOWN HARDWARE/SOFTWARE PERCENT	IMP DOWN HARDWARE/SOFTWARE MTBF	IMP DOWN HARDWARE/SOFTWARE MTTR	AVERAGE HOST TRAFFIC NODES	AVERAGE HOST TRAFFIC PACKETS/DAY INTERNODE	AVERAGE HOST TRAFFIC PACKETS/DAY INTRANODE
SEP 71	.59%	3.27%				18	51,386	55,451
OCT	1.66%	1.77%				18	95,930	267,560
NOV	1.65%	5.50%				18	116,515	254,578
DEC	3.21%	3.95%				19	107,896	577,619
JAN 72	1.02%	1.92%				19	172,037	4,375,642
FEB	1.23%	2.75%				19	224,668	339,081
MAR	1.36%	4.00%				23	240,144	435,016
APR	.88%	2.86%				25	362,064	407,638
MAY	1.11%	2.57%				25	505,639	177,860
		(BEGIN FULL OPERATOR COVERAGE)						
JUN	1.27%	.97%	.48	325	1:34	29	807,164	199,873
JUL	2.69%	5.56%	2.34	320	7:29	29	501,896	460,844
AUG	.66%	1.79%	.66	434	2:51	29	682,502	287,953
SEP	1.17%	1.44%	.64	684	4:23	30	828,917	252,988
OCT	1.44%	1.92%	1.14	491	5:35	32	1,357,048	452,972
NOV	1.81%	1.68%	1.06	444	4:43	32	1,184,062	241,884
DEC	2.63%	3.40%	3.12	422	13.11	34	1,139,590	173,573
JAN 73	1.28%	2.55%	1.77	335	5:55	36	1,455,325	337,308

*INCLUDES P.M., SITE ENVIRONMENTAL PROBLEMS, MACHINE HIT BY LIGHTNING, ETC.

Chart 15

23 NODE 28 LINK ARPA

Number of Circuits Failed	Number of Combinations to be Examined		Number of Cutsets
28	1		1
27	28		28
26	378		378
25	3276		3276
24	20475		20475
23	98280		98280
22	376740		376740
21	1184040		1184040
20	3108105		3108105
19	6906900		6906900
18	13123110	same	13123110
17	21474180		21474180
16	30421755		30421755
15	37442160		37442160
14	40116600		40116600
13	37442160		37442160
12	30421755		30421755
11	21474180		21474180
10	13123110		13123110
9	6906900		6906900
8	3108108		3108108
7	1184040		1184040
6	376740		340618
5	98280		≈70347
4	20475		≈9852
3	3276		827
2	378		30
1	28		0

Chart 16

TIP USER DIALOGUE

```
E
HELLO 306
@ L23↓
TR OPEN

ARC TENEX XXXXX, WELCOME
@ LOG↵
USER JONES↵
PASSWORD SECRET↵
ACCOUNT # 1000↵

JOB #6, TTY #2, 10 JAN 73, 0812
@ GET FILE <WORK EASY>↵
@ PRINT BRANCH 3 ↑↑

@ LOGOUT↵

JOB #6, TTY #2, USED 0.02 IN 10.31,
 10 JAN 73, 0822

@ C↓
TR CLOSED
```

Chart 17

A TIP IN THE NETWORK

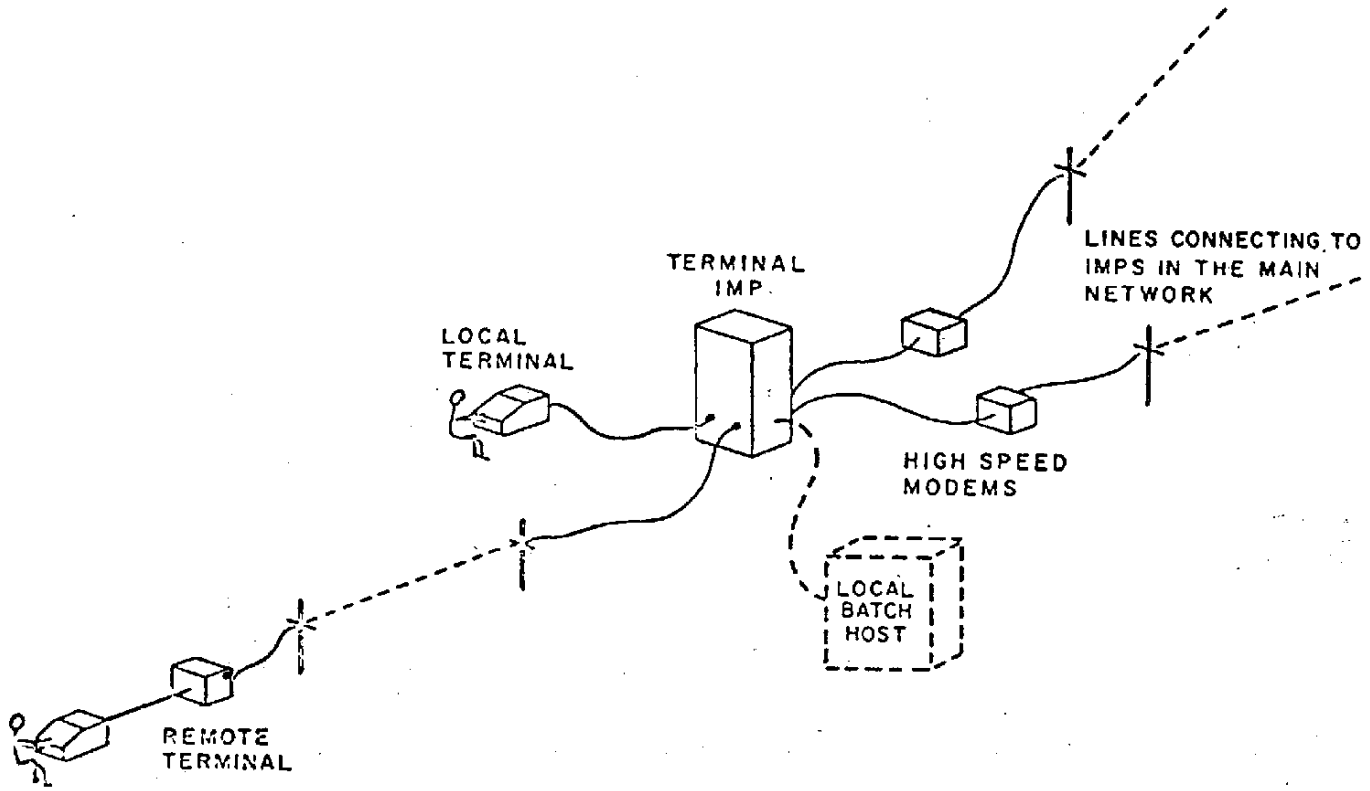


Chart 18

DATA FLOW FOR REMOTE INTERACTIVE USE

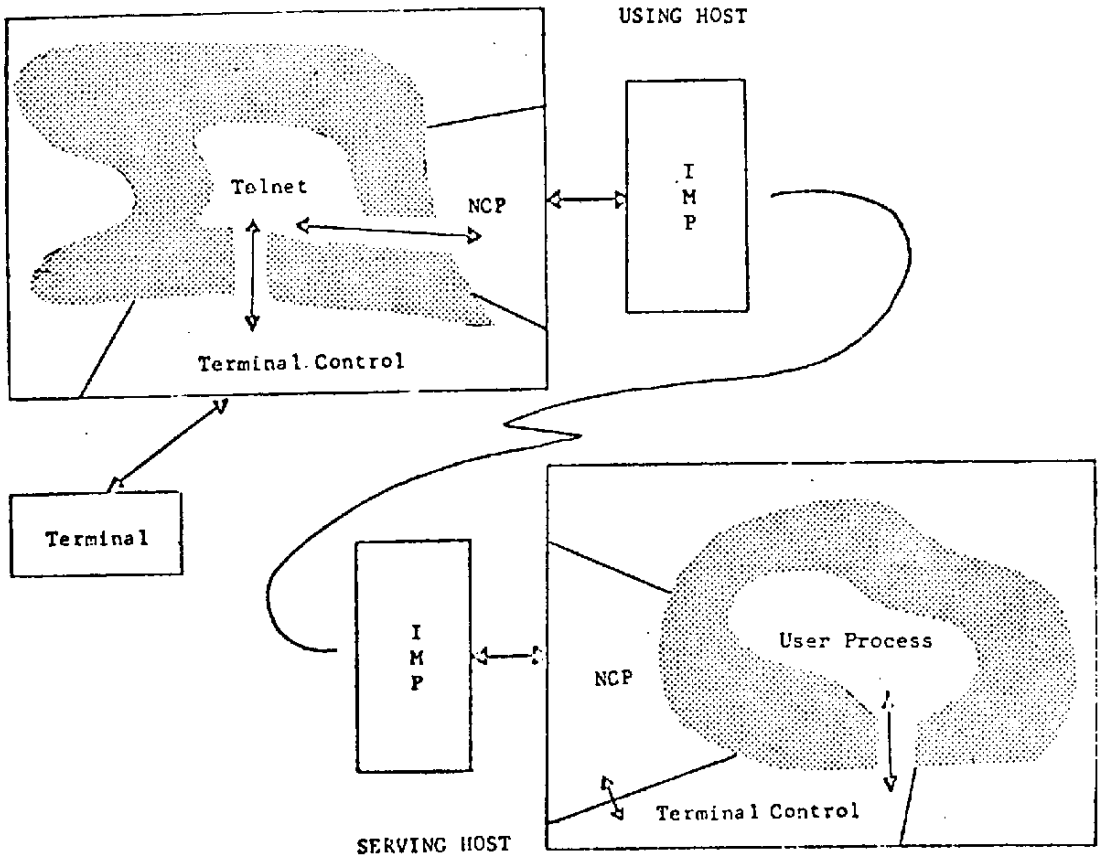


Chart 19

HOST/IMP INTERFACE SIMPLIFIED

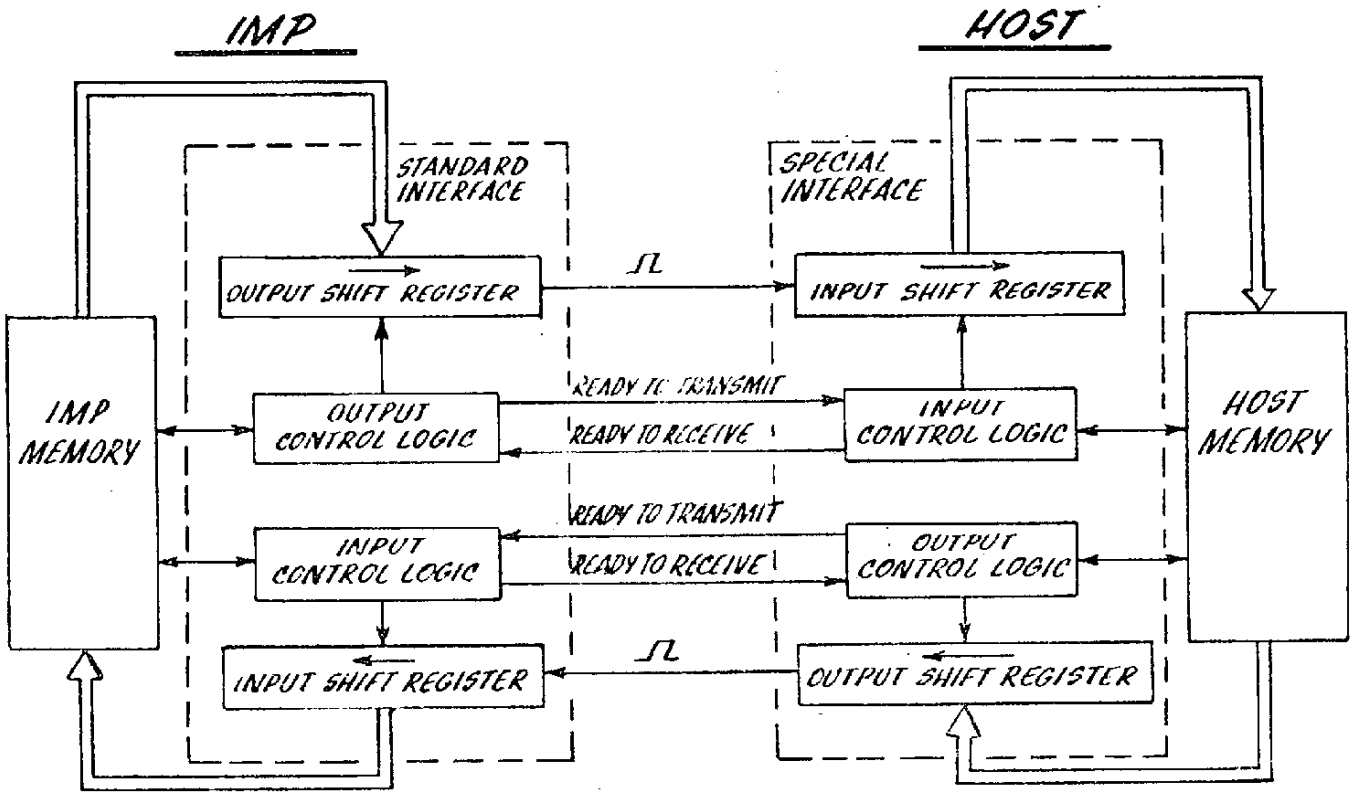


Chart 20

IMP/HOST CONNECTION FOR VERY DISTANT HOST

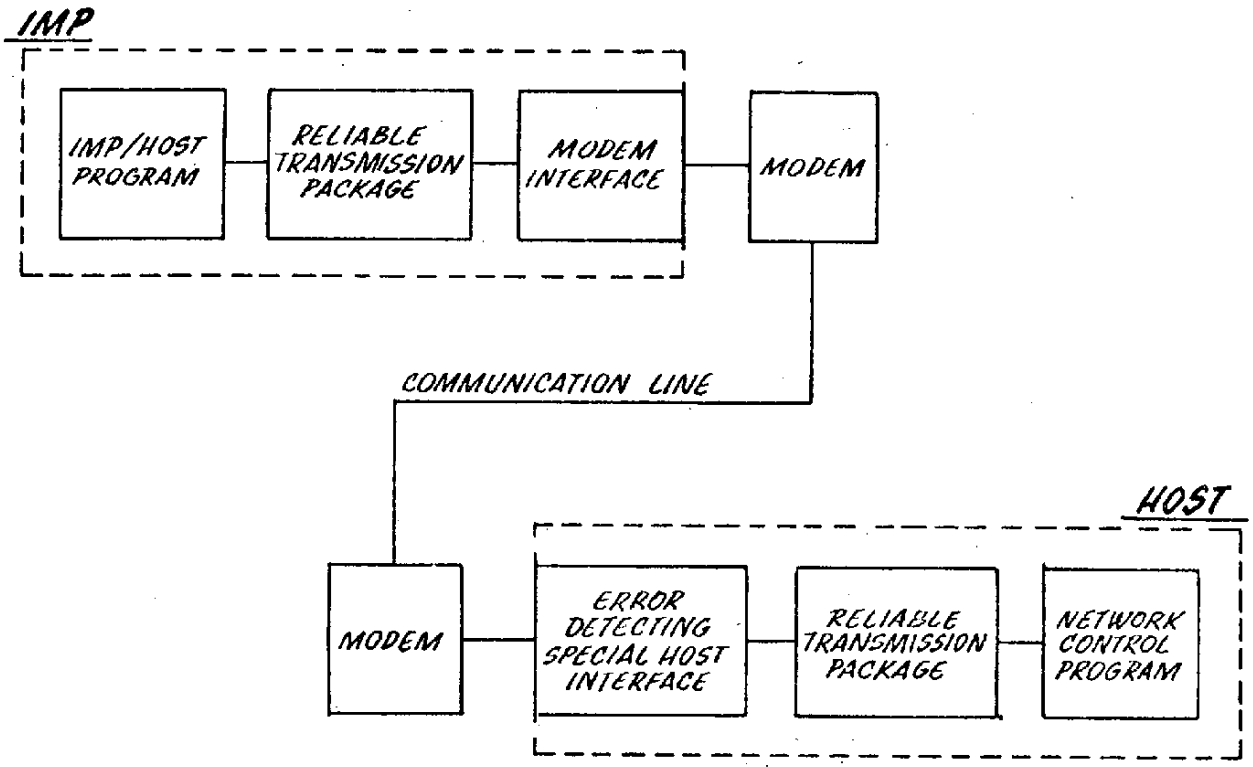


Chart 21

NUMBER OF BUFFERS FOR FULL LINE UTILIZATION

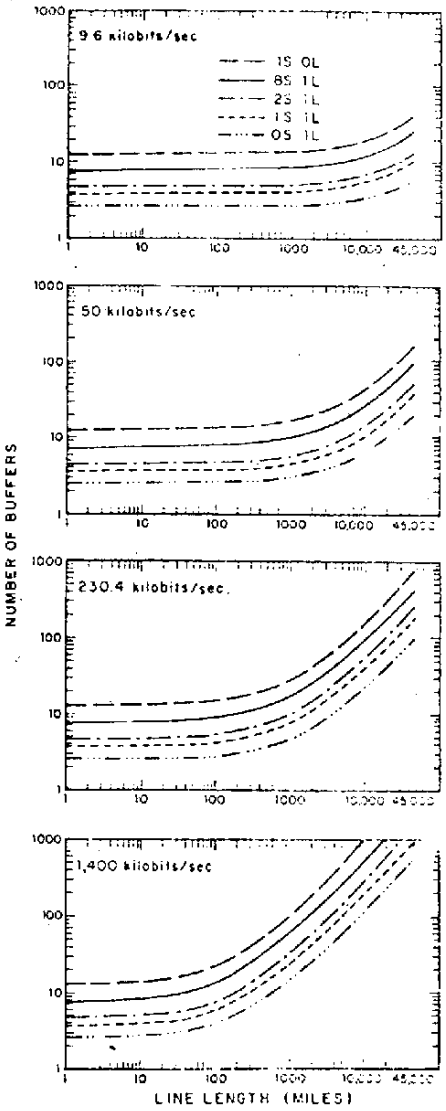


Chart 22

MAP OF CORE STORAGE

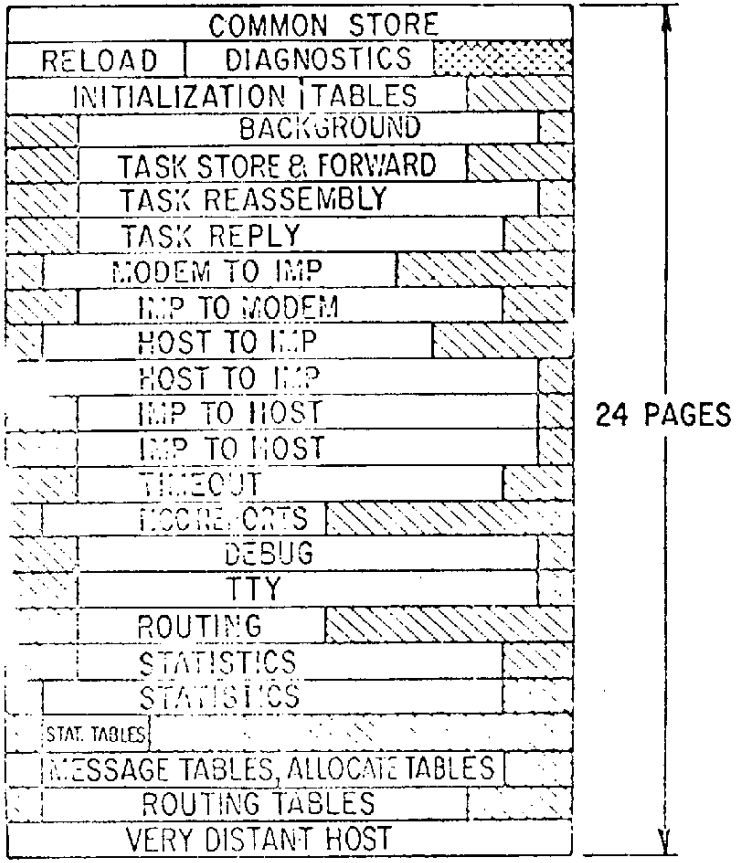


Chart 23

REASSEMBLY LOCKUP

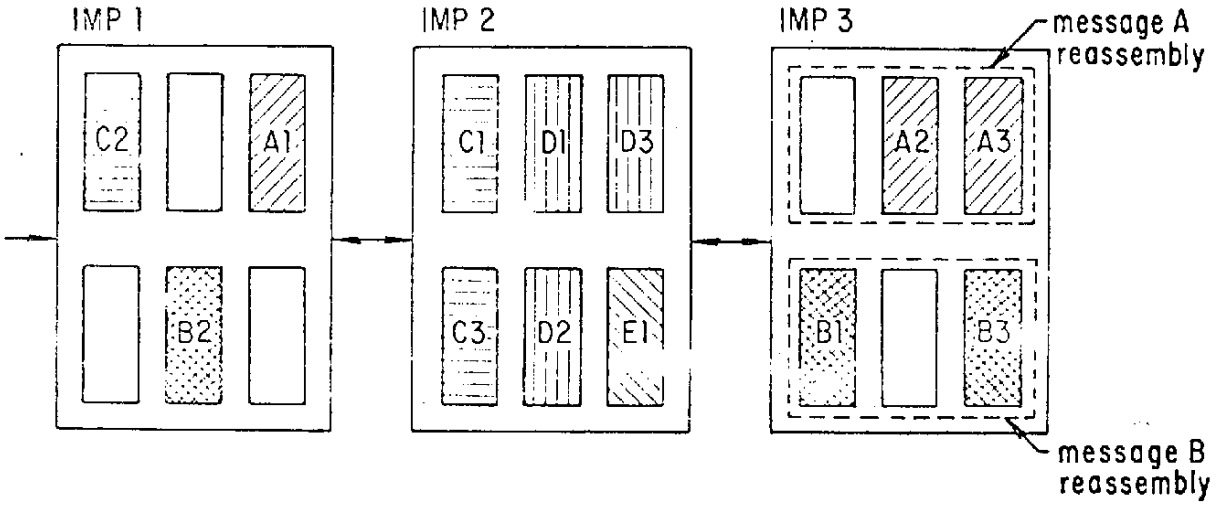
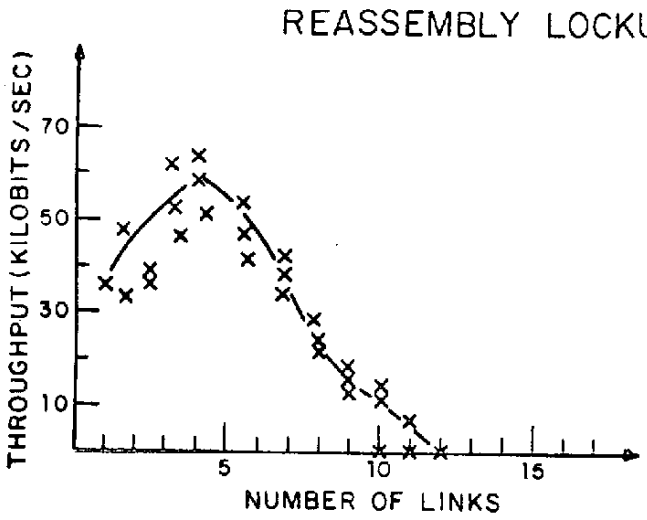


Chart 24



21 S/F BUFFERS
32 REASSEMBLY BUFFERS
8000 BIT MESSAGES

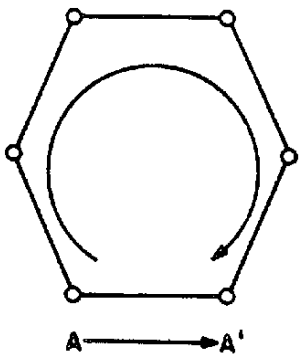


Chart 25

DIRECT STORE AND FORWARD LOCKUP

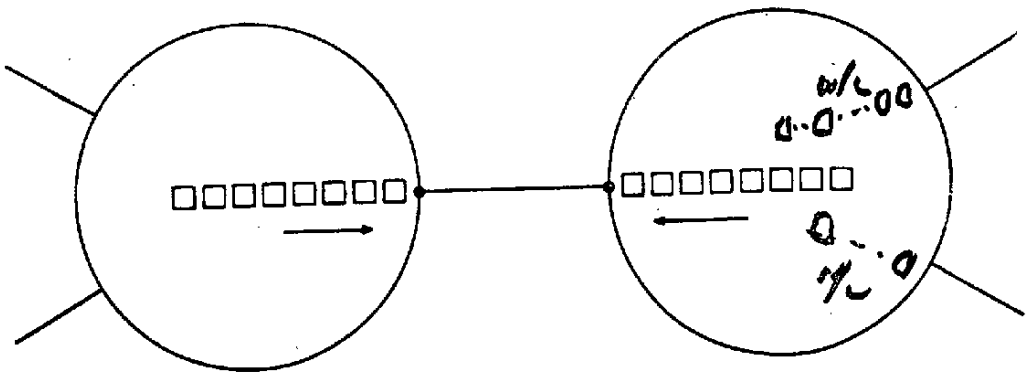
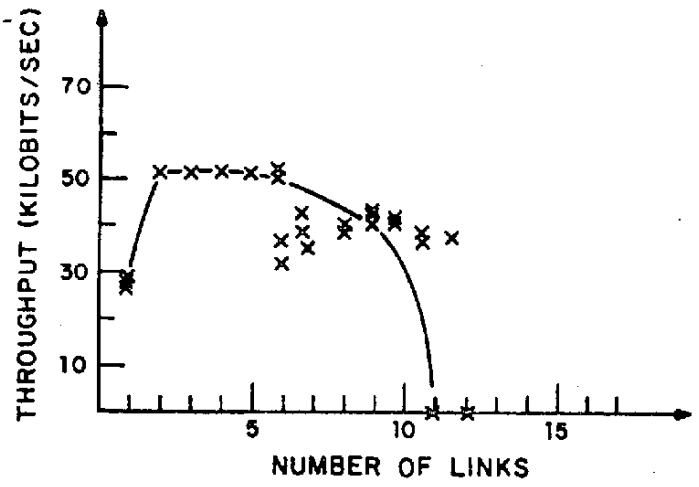


Chart 26

DIRECT STORE-AND-FORWARD LOCKUP



21 S/F BUFFERS
∞ REASSEMBLY BUFFERS
1000 BIT MESSAGES

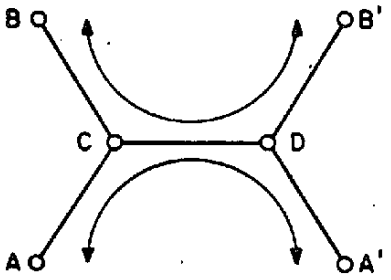


Chart 27

SPACE RESERVATION

- KEEP COPY IN SOURCE IMP
- IF SPACE, ACCEPT ORIGINAL AT DESTINATION, DISCARD COPY AT SOURCE WHEN RFNM RETURNS
- IF NO SPACE, DISCARD ORIGINAL, ASK FOR COPY FROM SOURCE WHEN SPACE OCCURS
- SEND SEPARATE SPACE REQUEST AHEAD FOR MULTI-PACKET MESSAGES

Chart 28

ALLOCATION OF IMP TABLE STORAGE

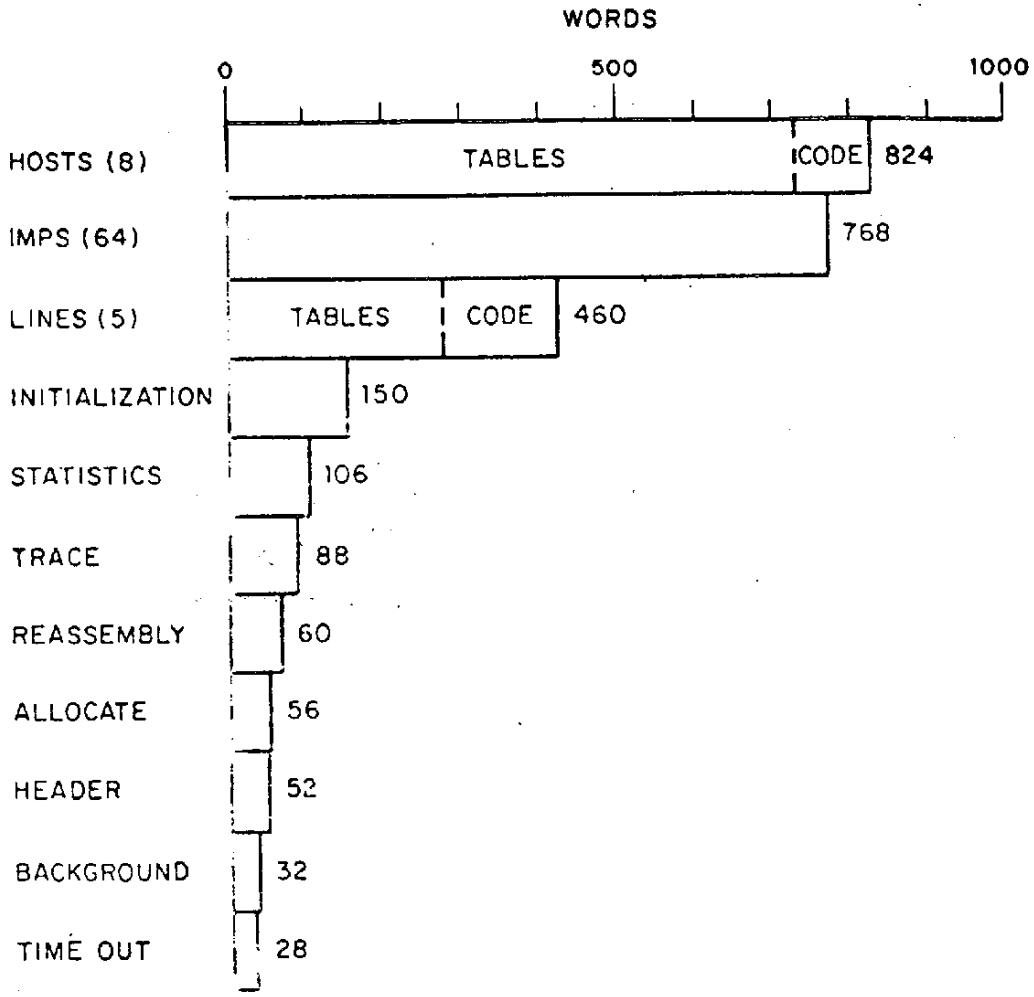


Chart 29

For the original IMP program, the program processing time per packet consisted of the following:

Modem Output	100 cycles	Send out packet
Modem Input	100 cycles	Receive packet at other IMP
Task	150 cycles	Process it (route onto an output line)
Modem Output	100 cycles	Send back an acknowledgment
Modem Input	100 cycles	Receive acknowledgment at first IMP
Task	150 cycles	Process acknowledgment
	700 cycles	Program processing time per packet

For the modified IMP program, the program processing time consists of:

Modem Output	150 cycles	Send out packet and piggyback acks
Modem Input	150 cycles	Receive packet and process acks
Task	250 cycles	Process packet
	550 cycles	Program processing time per packet

Chart 30

BLOCK DIAGRAM OF TIP PROGRAM

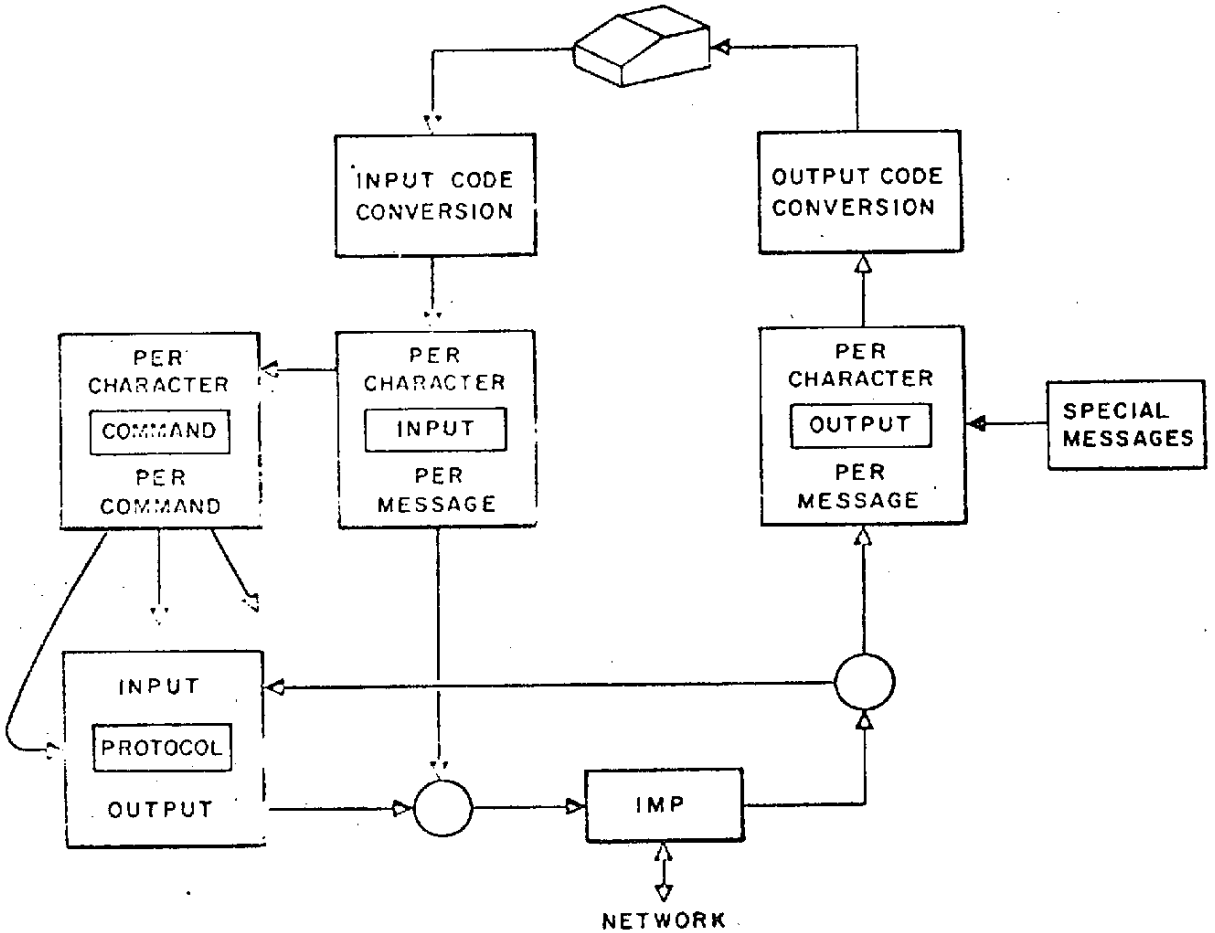


Chart 31