

WIDE 上の X.25 機能の設計と実装

本田 和弘 加瀬 直樹 尾上 淳 中島 達夫 所 真理雄
慶應義塾大学

村井 純
東京大学

ネットワークの広域化にともない、信頼性があり、同時に複数の相手と通信が可能であるパケット交換網を利用した通信の重要性が増加してきている。X.25 を用いたパケット交換網の利用は、広域分散環境を実現する上で重要な方式の一つである。これを異種プロトコルのネットワーク間接続に用いるためにはインターネットワークプロトコル (TCP/IP) 技術を考慮する必要がある。

本論文では、TCP/IP 下で X.25 を柔軟かつ高速に利用する方式を設計、提案する。これは、汎用プロトコルのデータリンク機能として X.25 パケット交換網を利用する際の効率化に有効である。

The Design and Implementation of X.25 functions on WIDE

Kazuhiro Honda Naoki Kase Atsushi Onoe
Tatsuo Nakajima Mario Tokoro
Keio University

Jun Murai
University of Tokyo

Utilization of X.25 packet switching network as a data-link entity of existing network protocol requires different technology than which of permanent link. Because of its reliability and functions, achievement of this technology is important in terms of construction of large-scale distributed computing environment.

In this paper, we propose a design and an implementation of efficient way to utilize X.25 packet switching network under TCP/IP. This approach is effective to use X.25 packet switching network as the data-link for the general protocols.

1 はじめに

WIDE(Widely Integrated Distributed Environment)プロジェクト[1]¹は、東京大学、慶應義塾大学、東京工業大学、大阪大学などを中心に、広域分散ネットワークを構築するにあたっての問題点を検討し、広域分散環境に必要な新たな技術の確立を目的としている。主な研究テーマは、広域分散環境における名前空間、ユーザの認証機構の研究、経路制御問題、広域通信におけるプロトコルの研究などが挙げられる。従来のネットワーク OS から分散 OS への移行期にある現在、これらの研究は非常に重要な位置をしめるものである。本論文では、WIDEにおける広域ネットワーク、特にX.25を用いた広域ネットワークの役割について検討する。

広域ネットワークを考えた場合、いかに速くデータを送信できるかによって、そのネットワークの質が決まると言える。しかし、現実には、物理的に回線速度が決まっているネットワークを規格以上に高速化することは不可能である。

そこで、スピードアップするためには、通信する2つのマシン間で論理的な回線を複数本張ることが考えられる。ここで、スピードを左右する要因として考えなければならないことは、回線を選ぶアルゴリズム、パケットのサイズ、回線の本数などが挙げられる。本研究では、パケットを複数の論理回線のうち、どの回線上に流せば最も効率が良いかについて検討する。

今回、この実験環境として、公衆パケット交換用プロトコルであるX.25を採用する。X.25は、マシン間で簡単に論理回線を複数張れ、今回の実験に非常に適していると思われるからである。また、X.25には、ウィンドウサイズやパケットサイズなどかなり細かな点までネゴシエーションができる機能が備わっているため、マルチリンクの実験をする上で最適な環境である。これにより、X.25の高速化も試みることができる。

また、従来のアプリケーションを変更せずに利用できるようにするため、トランスポートレイヤよりも上位のプロトコルは変更しない。すなわち、トランスポートプロトコルの標準であるTCP/IPの下位レイヤに、改良したX.25を構築する。

具体的には、ある二点間の通信において、論理回線を複数本張り、IPパケットを並列に転送することで回線の高速化を実現する。そのため、回線を選択するアルゴリズムを2つ提案し、どのような

場合に、これらのアルゴリズムを使い分けていくのが良いかを考察する。

2 広域分散環境における広域ネットワーク

既存のネットワーク同士を論理的に結合し、新たに大規模なネットワークを構築するインターネットワーキングが世界中でさかんに行なわれている。現在、アメリカのARPANET, CSNET, NSFnet, BITNET、日本ではJUNET²などの各国のネットワークが相互接続されている。

ネットワークの広域化による最大の利点は、計算機環境の地理的、物理的拡大により、遠隔地にあるリソースを共有したり、数多くの人々の間でコミュニケーションを可能にしたりすることである。特に、電子メールは、不特定多数の相手と比較的短時間にやりとりができるため、計算機研究者の間で活発な意見交換が可能である。アメリカなどでは、以前から電子メールを用いた議論が数多くなされ、その結果として、計算機科学はまれに見る急速な発展をとげている。その意味で、電子メールの果たした役割は非常に大きいと言える。

しかしながら、電子メールやニュースシステムなどの情報交換だけではなく、遠隔サイトにあるデータベースなどの計算機資源を対話形式で利用したいという要求がある。この遠隔データアクセスは、次世代OSである分散OSを構築するうえで、なくてはならない基本要素でもある。この広域分散環境下においては、遠隔地にある資源にアクセスするアプリケーションが、ローカルエリアネットワークと同等のスピードで透過的に実行されなければならない。そのためには、かなり高速な広域ネットワークの実現が必要とされる。

広域ネットワークを構築するにあたっては、基本的に次の三通りの方法がある。

1. 電話回線
2. 専用回線
3. パケット交換網

第一の電話回線を用いる方法は、回線契約者が多いため比較的簡単な方法でネットワークを確立することができる。これは、従来のJUNETで広く用いられている方法である。しかし、従来の電話回線によるアクセスでは、相手とコネクションを確

¹本研究は電気通信基金普及財団昭和63年度研究助成金並びにWIDEプロジェクト共同研究各社の助成と協力によって行なわれている。

²日本でも最近、東京大学をゲートウェイとしてNSFnetに接続された。

立するためにダイヤルするという操作を必要とする。この時間的制約があるため、回線交換方式による通信では、対話形式にアクセスことは極めて困難であると言える。この制約を排除するためには、コネクションを一度確立したら二度と回線を切断しないという方法も考えられるが、経済的に実用的ではない。また、回線交換方式では、一度に一つのサイトとしか通信できないので、分散環境のように、いついかなる時でも任意の相手と通信する必要があるときは、この方式は使えない。また、相手が話し中で、回線がふさがっていたときには通信する手段がないというのもこの方式の欠点である。

第二の方法は、専用線を近いサイト間で張っていき、そのつなぎ合わせで広域ネットワークを構築する方法である。この方法は、かなり実用的であり、かつ、高速通信を期待できる。しかし、各サイト間で専用線を維持するために、莫大な運用資金を要する。また、専用線はあくまで個人的な利用目的のために構築されるため、そのサイトにとって無関係なパケットまで転送するかどうかは、そのサイト間の取り決めによる。従って、それを基礎に公衆な広域ネットワークを構築することは、各サイト間でよほどの合意がない限り困難であると考えべきであろう。

そこで、第三の方法として考えられるのが、パケット交換網を利用する方法である。パケット交換とは、データのある一定の規則によりパケットという形に区切り、それを回線上に流すことによって通信する方法である。これは point to point である回線交換方式とは異なり、各パケットが多重化されているために、同時に複数のサイトと通信を行なうことが可能である。この機能により一人が回線を占有することがなくなる。また、データパケットは、相手に対して確実に配達されることを網が保証しているために、タイムアウトによる再送などを考慮する必要がない。このため、かなり信頼性があると言うことができる。課金に関しては、通信したデータの量によって計算されるので、たとえば回線をアイドル状態にしたままでもいっこうに料金はかからないという利点がある。また、通信距離による料金の格差がほとんどないのも特徴のひとつである。

以上の様に、広域ネットワークを構築するに当たっては、通過する情報量とそれにかかる費用を十分考慮した上で、メディアを選択しなければならない。その大まかな概念図を図1に示しておく。この図からも明らかなように、広域通信においては、パケット交換網が、コストパフォーマンスの点で優れてい

る。このパケット交換網を利用するためのプロトコルとして広く使われているものが、CCITT 勧告の X.25 であり、アメリカの CSNET の X25NET など採用されている。日本における X.25 ネットワークには、NTT の DDX、学術情報センタの NACSIS などがある。また ISDN もパケット交換をサポートしている。

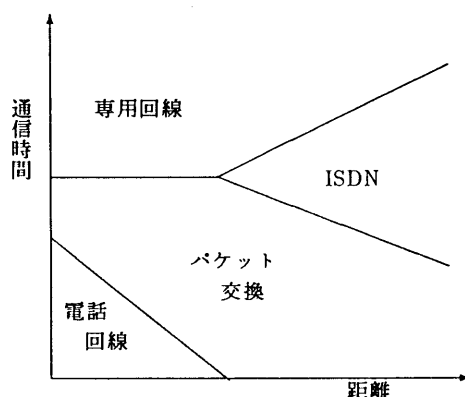


図 1: 通信回線の適用領域

しかし、パケット交換方式には上述のような利点がある反面、通信にあたって遵守しなければならない制限がかなり細かく規定されている。その一例として、

- 網内に転送できるパケットの個数の制限 (window size)
- 1パケットの大きさ (packet size)
- コネクションの確立、切断に関する規則
- タイムアウト
- ネゴシエーション

などが挙げられる。X.25 においては、スライディングウィンドウと呼ばれる手法により、網内に転送できるパケットの個数があらかじめ決められているので、その制限を越えてパケットを送信することは許されない。これは、パケット交換網が混雑しないようにフロー制御を行なっているためである。しかしながら、この方法では、網が混雑していないときにもフロー制御がかかってしまう。そこで、網が混んでいない時には同一相手に対して論理回線を複数本ひけば、プロトコルを変更することなく、

網内に数多くのパケットを転送することができるようになる。このため、公衆パケット交換網において高速通信を実現することが可能となる。

さらに今後、遠隔サイトにある音声や画像といったマルチメディアデータを扱うために、大容量のファイル転送が頻繁に行なわれることが予想されるが、そのための予備技術としても、論理回線のマルチリンク化の技術は重要である。

3 WIDE/X.25 のアーキテクチャ

WIDE/X.25 のアーキテクチャは、大きく 4 つの部分 (HDLC, X.25, Multilink, IP Interface) に分けることができる。

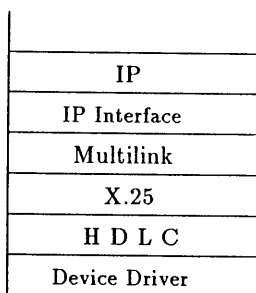


図 2: マルチリンク化された X.25 のレイヤ構造

最下位のレイヤは、HDLC モジュールと呼ばれ、伝送制御手順を規定している部分である。HDLC では基本機能として、回線の接続、切断、データフレームの送信を行なう。回線の設定など HDLC のコントロールに関しては、2 ウェイハンドシェイクを行ない、回線の状態遷移に矛盾の生じないようにしている。その他、通信の信頼性を高めるためフレーム毎にシーケンス番号を振り、この番号の入った ack をやり取りすることで、順序誤りやフレームの消失などのエラーから回復できるようにしている。また、DTE - DCE 間で輻輳を抑えるため、スライディングウィンドウというアルゴリズムを用いてフロー制御をしている。

その上のレイヤは、X.25 モジュールで、通信するマシン間のコネクションを end to end で管理する。すなわち、このレイヤでは、通信したい相手に対して 1 対 1 で論理回線を張ることになる。このため、論理回線が設立されていない相手に対しては、通信に先だって呼の設定が必要となる。この時、相手に対して、ウィンドウのサイズ、スループット、

その他のファシリティなどをネゴシエーションすることができる。また、HDLC と同様にスライディングウィンドウを用いたフロー制御を各回線毎に行なっている。ここで、HDLC と異なるのは、X.25 レイヤがパケットの再送を行なわないということである。なぜなら、DTE - DCE 間は、HDLC がフレームを再送することによって通信の信頼性を保証しており、DCE - DCE 間は、網自身が信頼性を保証しているので、結局、end to end でパケットが相手に到達することが保証されることになるからである。

その上位レイヤにマルチリンクモジュールを設ける。このモジュールは、一つの通信相手に対するコネクションを複数本管理する役割を持つ。すなわち、上位レイヤに複数の X.25 の論理回線を 1 本の回線であるかのように見せる。従来の X.25 では、各通信相手に対して論理回線は 1 本であったが、これを複数本確立する。そして、この部分であるマシンに向かう IP パケットを効率良く振り分けることによって、X.25 の高速化を実現する。また、回線の使用状況を見て、動的に論理回線の本数を増減する機能を持つ。これにより、バンド幅を広げ、スループットを向上させることが可能となる。ここで、注意しなければならないのは、IP レイヤから見るとただひとつの論理回線が設立されているように見えることである。つまり上位層からは、回線が太くなったように見える。

最上位のモジュールは、IP とのインタフェースをつかさどる部分である。このレイヤでは、IP アドレスから DTE アドレスへの変換、その逆変換を受け持つ。そのための情報をテーブルとして持っており、ユーザレベルから追加、削除が可能である。また、IP リンクを統括的に管理するため、論理回線の接続、切断などの制御を指示する。

4 設計と実装

この章では、各レイヤ毎に基本設計と実装について述べる。

4.1 HDLC モジュール

HDLC モジュールは、大きく分けて 3 つに分類できる。DTE から DCE に対してフレームを送受信するための送信モジュール、フレーム受信モジュール、そして、一定時間毎に起動されるタイマモジュールである。さらに、回線の状態遷移を保持するための HDLC コントロールブロック (hdlccb) と呼ばれる構造体がある。これは、3 つのモジュールのど

れからもアクセス可能で、HDLC モジュール内にただひとつ存在する。各レイヤ間で渡されるデータは、メモリバッファ(mbuf) を単位として受渡しされる。

送信モジュールは、論理回線を確立するための SABM や受信フレームに対する ack である RR などの呼の制御に関するフレームを送信する他、X.25 のレイヤから渡されたデータパケットをフレームに加工して送信する働きをする。呼の制御に関するフレームを送信する際は、受信モジュールが適当なフレームを送信するための関数を直接呼び出す。また、X.25 のレイヤからデータパケットを送信する場合は、`hdlc_output()` という関数に mbuf を渡す事で必要なヘッダを付加した後、送信される。ただし、この時に、データフレームの再送要求がくることを考慮して、データフレームのコピーがなされる。これは、そのフレームに対する ack が来るまで保存される。

受信モジュールは、フレームが到着することによって起動される。フレームは予め決められたキューに入れられているので、受信モジュールは先頭のフレームから順番に処理をしていく。まず、最初の処理としては、フレームのコントロール部を解析することである。この解析結果によって、フレームの再送、再送用のコピーの破棄、ウィンドウなどの状態遷移の更新を行なう。また、受信フレームがデータフレームであった場合は、HDLC のヘッダ部をはずし、残りのデータ部を `x25_input()` の引数にすることによって、上位レイヤに渡す。

タイマモジュールは、OS のタイマ割り込みによって 0.5 秒に一回起動される。このモジュールの機能は、タイムアウト変数を更新することと、フロー制御などの理由によりキューに溜ったままになっているフレームを取り出して、送信モジュールまたは、受信モジュールに渡してやることである。もし、タイマ変数が 0 になった場合は、網を制御するために適切なフレームを送信モジュールに渡す役割をする。

なお、HDLC リンクの確立は、OS の起動時になされるため、上位レイヤは関係しない。また、リンク切断後の回復は HDLC 内で自動的に処理されるため、上位レイヤには全く知らされない。

これにより上位レイヤでタイムアウトを起こさない限り、再び回線を設定しなおす必要はなくなり、各レイヤごとに全く独立に回線の管理を行なうことができる。

4.2 X.25 モジュール

X.25 レイヤでは、HDLC と異なり通信相手毎に論理回線を確立するため、状態を保持するためのコントロールブロック (x25cb) を論理回線数だけ必要とする。一ヶ所の DTE から設定できる回線の本数は決められているので、この x25cb の個数も一意に決まる。そのため、マルチリンク手法を導入した際、約 80% が常に使われることを予想し、初期設定時にこれらのコントロールブロックを確保している。今回の実装では、NTT の DDX-P を基準にしているため、90 個を目安にしている。X.25 レイヤも基本的には HDLC と同様に、送信モジュール、受信モジュール、タイマモジュールの 3 つの部分に分けることができる。

論理回線の確立、切断などの回線のコントロールは、すべて上位レイヤであるマルチリンクモジュールが直接、X.25 の関数を呼び出すため、送信モジュールは、データパケットの送信のみを受け持つ。このとき、HDLC と同様に、x25cb 内のウィンドウの状態により、直ちに送信するかキューに入れるかを決定する。送信する場合は、`hdlc_output()` という関数を呼んで、下位レイヤにパケットを渡す。

受信モジュールは、受信パケットのヘッダを解析し、チャンネルグループ番号とチャンネル番号から、回線の x25cb を決定する。以下、このコントロールブロックを基準としてパケットを処理する。すなわち、受信モジュールはパケットタイプを見て、適当な処理を行なう関数を呼ぶが、x25cb を引数として渡すことにより、論理回線の制御を行なっている。たとえば、受信したパケットがデータパケットであった場合には、マルチリンクレイヤに回線の状態を知らせるために、データパケットに付随して x25cb も渡すという操作をしている。

タイマモジュールは、0.3 秒毎に起動される `fasttimo()` と 0.5 秒毎に起動される `slowtimo()` というモジュールから成り立っている。X.25 レイヤでは、連続した受信データパケットに対して、ack を即座に返すことはせず、いくつか受信パケットが溜った段階で、ack を一つだけ返す方法を取っている。これは、送信されるパケット数を減らし、効率の良いデータ転送を行なうためである。これらの受信パケットに対する ack を返すための関数が、`fasttimo()` である。 `slowtimo()` は、各コントロールブロック毎にタイマ変数の更新、キューに溜っているデータの送信などを受け持つ。

4.3 マルチリンク モジュール

X.25 のレイヤでは、各論理回線毎にコントロールブロックが存在したのに対し、このレイヤでは、通信する相手毎に一つのコントロールブロック (xmlcb) を持つものとする。ここで、一つの xmlcb が複数の x25cb を管理することにより、ある通信相手に対して複数のチャネルを持つことが可能になる。この結果、バンド幅が広がり、スループットの向上した通信が可能となる。

マルチリンクモジュールには、X.25 モジュールと同様、送信モジュールと受信モジュールが存在する。

送信モジュールでは、論理回線の設定・切断等の回線のコントロール、およびインタフェースモジュールから渡された IP パケットの送出を行なう。論理回線のコントロールのために、このモジュールでは X.25 モジュールの各関数を呼び出し、必要なパケットを送出している。また IP パケットが X.25 パケットより大きい場合このモジュール内でパケットの分割を行なっている。

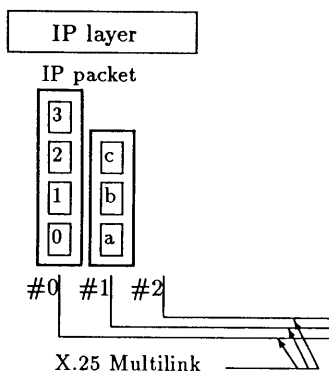


図 3: たてうなぎ方式

一つの相手に対して論理回線を複数本張るために、ここではマルチリンクに用いるリンク数のネゴシエーションを行なっている。これは、一般に発呼側と着呼側で許すことのできるリンク数が異なっているためであり、少ない方を用いてマルチリンク回線を張る必要があるからである。これには、まず発呼側から必要とする最大本数のデータを便乗させた呼設定用の CR パケットを一つ送出する。着呼側では、これに対する応答用 CA パケットに最大本数のデータを便乗させて返す。このパケットが到着すれば、発呼側であと何本の論理回線を張れ

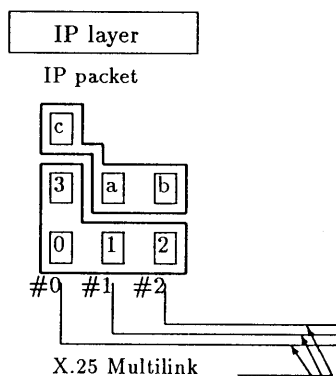


図 4: よこうなぎ方式

ばいいのかが判断できるので残りの論理回線を張る。一方、着呼側にネゴシエーションのデータに乗せたパケットが到着した時点で、着呼側ではマルチリンクに用いる回線数を判断することができる。したがって、残りの回線は着呼側から張るようにする。これによって、リンクの設定時間を減少することができる。

複数の論理回線で接続された一つの通信相手に対して IP パケットを分割して送信するアルゴリズムとしては 2 通りの方法が考えられる。一つは、IP パケット毎に送信に使用する論理回線を選択し、その中でパケットを分割して送り込む方式である。この方式では、IP パケット単位でパラレルに送信されることになる。これを「たてうなぎ方式」と名付ける。これに対し、送信する IP パケットをまず分割し、分割されたパケット毎に回線を選択する方法をとることもできる。この方法では、分割された断片がパラレルに流し込まれることになる。これを「よこうなぎ方式」と呼ぶことにする。

「たてうなぎ方式」の場合には、パケットの到着状況によっては回線の使用率にばらつきが出ることがあるが、「よこうなぎ方式」では常に平等に回線を使用していくため、回線の使用効率の点で優れていると考えることができる。つまりパケットの到着が少ない場合、「たてうなぎ方式」ではある回線では待ちが生じているにも関わらず別の回線は空いているという状況が生じてしまう可能性があるが、「よこうなぎ方式」では、必ず全ての回線を使用することになる。したがって、X.25 の高速化を考えた場合、リンク間の平等性の高い「よこうなぎ方式」を採用するのが妥当である。

受信モジュールでは、回線設定要求の受理と、断片化されたパケットの再構成を行なう。パケットを組立てるためには、どのような順番でパケットが送られてきているのかを判断できなければならない。このために、パケットに順序番号を付加することも考えられるが、処理が複雑になりそのオーバーヘッドを考えると実用的でない。ここでは、使用できる回線に順序番号を割り振ってその順番に従ってパケットを流すことにする。受信モジュールでは、この順番通りにパケットを取り出してつなげていくことで、パケットの再構成を行なう。

4.4 インタフェースモジュール

インタフェースモジュールは X.25 の機能全体を他のハードウェアインタフェースと同様に扱うためのモジュールである。論理回線そのものは必要に応じて設定、または切断されるが、IP レイヤからは常に接続されているように見える必要がある。従ってこのレイヤでは主に論理回線の管理を行なう。

インタフェースモジュールにおいてもマルチリンクレイヤ同様、通信相手毎のコントロールブロックを作成する。このコントロールブロックはマルチリンクモジュールのコントロールブロックと 1 対 1 に対応しており、必要に応じて動的に確保及び解放される。

まず IP レイヤとのやりとりは IP アドレスにより行われ、これ以下のレイヤでは DTE アドレスによって行われるため、そのアドレス間のマッピングをしなければならない。このマッピングテーブルの管理は静的にファイルまたはコマンドにより行う方法、及び動的に問い合わせることによって行う方法などが考えられる。

公衆パケット網のような広域分散環境を考えた場合、アドレスの変化などに対応するためには動的に管理される方が望ましい。しかし一方比較的低速のネットワークであるために問い合わせのオーバーヘッドによる効率の低下も無視できない。従って現状では、アドレスマッピングは静的に行われ、ユーザコマンドから呼び出される ioctl によって登録及び削除を行う。またテーブルの参照にはキャッシュを用いている。

パケットの送信時には、論理回線の設定の有無を調べ、回線が設定されていない場合はコントロールブロックの作成及びマルチリンクモジュールに対する回線設定の要求を行なう。また回線が既に設定されている場合には、下位レイヤの状態を見て直接あるいはキューを用いてマルチリンクモジュールに渡す。網には無限にパケットを送り込めるわけではないのでフロー制御が必要になってくる。このモ

ジュールでもそれに対応する必要があり、マルチリンクモジュールからの指示に従って制御を行うが、キューが溢れたときにはそのパケットは消失する。

パケット受信時にはマルチリンクモジュールから渡されたデータをキューにセットして IP レイヤに割込みによって渡す。

次に論理回線の管理であるが、回線は送信あるいは受信のパケットがあったときに限り設定される。しかし論理回線として割り当てられるのは有限であるため、不要なものは切断する必要がある。また網によっては回線を設定しておくことそのものに対して課金される可能性がある。そのため一定時間アイドルであった回線はマルチリンクの場合でも、その全てを切断する。

この場合でも各回線に対してアイドル時間が短い場合が考えられる。このようなときに新たな接続要求があった場合、とる手段としては以下の二つが考えられる。一つは LRU, FIFO などのアルゴリズムを用いて使用中の回線を切断し、要求のあった接続を行うものである。またもう一つは既に回線の空きがないということで新たな要求を fail させてしまうというものである。

前者は必ず相手にパケットが届くが、同程度に使用されている回線がいくつか存在した場合、回線の取り合いが起これば転送効率が著しく低下する可能性がある。それに対し後者の方法では効率の低下は比較的小さいと思われるが、場合によっては回線自身は混んでいなくても接続先が多いか、マルチリンクを多用したために新たな接続ができなくなる可能性がある。この問題はマルチリンクの動的な変更、切断時間の適切な設定により軽減される見込みはあるが、現状では前者の方法を採り、LRU アルゴリズムにより回線を切断し、新たな接続を行っている。

5 現状

本論文で、提案されたマルチリンク化された X.25 を UNIX 4.3 BSD 上に実装、デバッグ中である。実装は、C 言語によってなされ、HDLC 部約 2000 行、X.25 部約 2000 行、マルチリンク部約 500 行、インタフェース部約 500 行の合計 5000 行からなる。

6 結論

ネットワークの拡大により LAN と WAN の間にデータ転送の速度にかなりな開きがでてきた。一例として、ローカルエリアネットワークで使用され

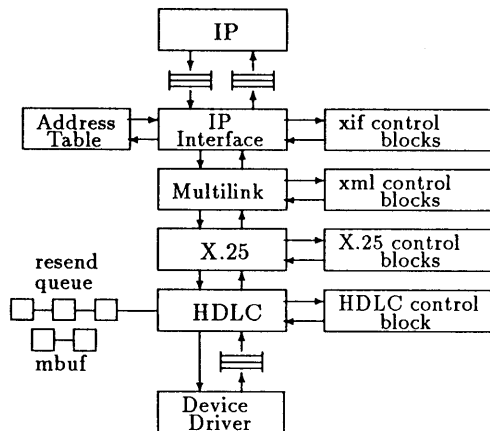


図 5: ソフトウェア構造

ているイーサネットと広域通信で使用されている X.25 とでは、その速度差が 1000 倍以上違う。今後、分散環境を実現していく上で、ローカルアクセスとリモートアクセスの速度的差異を小さくする必要がある。そこで、広域回線においてマルチリンク化の手法を用いることで、データ転送を高速化を実現する。

今回の研究により、マルチリンク方式によるデータ転送が、従来のシングルリンク転送に比べ、有効であることが確認された。また、上述の様に、マルチリンク化された X.25 の論理回線に対して、分割された IP パケットを並列に挿入する方式には 2 つのアルゴリズムが考えられるが、一つのパケットを「よこなぎ方式」で挿入する方が、「たてうなぎ方式」で挿入するよりも勝っていることが明らかになった。

上述のようにマルチリンクの手法が、広域回線を高速化するのに有効な技術であることが分かった。しかしながら、今後マルチメディアデータを扱う場合、取り扱うデータ量は、現在とは比較にならないくらい膨大な量となることが分かっている。また、情報自体の特性から厳密な信頼性を追求するよりも、高速通信の実現を必要としている。今回は、パケット交換のみに的を絞ったが、今後 ISDN などパケット交換と回線交換が同時に使える環境や複数の通信手段が存在する場合、いかにしてパケットを効率良く振り分けるためのアルゴリズムを見つげ出すかが今後の課題である。

謝辞 本研究に関する多くの議論と助言を頂いた WIDE 研究会のメンバーに感謝する。

References

- [1] 村井純 (東大), 加藤朗 (東工大), 佐藤智満 (慶大), and 楠本 博之 (電総研). 大規模広域分散環境 WIDE の構築. 1989.
- [2] 中島達夫 and 所真理雄. X.25 プロトコルを用いた Unix 上のネットワークシステムの設計と実装. 電子情報通信学会論文誌, July 1897.
- [3] CCITT. *Yellow Book X.25*. 1980.
- [4] Douglas Comer. *Internetworking with TCP/IP*. 1988.
- [5] Jun Murai and Akira Kato. Researches in Network Development of JUNET. In *Proceedings of SIGCOMM '87 Workshop*, 1987.
- [6] Jon Postel. *Internet Protocol - Darpa Internet Program Protocol Specification*. September 1981. RFC 791.
- [7] Jon Postel. *Transmission Control Protocol*. September 1981. RFC 793.