報告番号 乙第2948号

分散型問題解決システムの構成法に関する研究

山崎　晴明

＜目次＞

第　1　章

序　論

１．１　分散型問題解決システム構成のための諸要素技術

今日の情報化社会は、コンピュータと通信とが融合したシステムを前提として成り立っているといっても過言ではない。このようなシステム形態は又、広義の分散処理システムであるとみることもできる。つまり、こうしたシステムは、概念的には、バスやデータ通信網を介して多数の処理装置が結合しそれらの装置間で自由なデータの交換を可能とする形態としてモデル化することができる。

分散処理システムの最初の動向は、1960年代中期において、大型計算機のフロントエンドＩ／Ｏプロセッサにミニコンピュータを使用したことに始まるとされる。その後、タイムシェアリングシステムやコンピュータネットワークの普及と発展により分散処理の概念は増々有望視されるようになった。

一方、1970年代に入ってからの、情報処理産業に対する最大のインパクトは、デバイス技術の進歩によるＬＳＩの登場であった。これによりマイクロプロセッサの誕生をみ、コンピュータ処理のコストを大巾に引き下げることになり、分散システムというアプローチは増々魅力あるものとなった。特に従来の集中管理型情報処理システムに比し、そのシステム構築における柔軟性や経済性に優位点を見出すことができよう。

なお、今日の分散処理システムは、その地理的広がりや、データ通信網の速度等により、以下の様な３つのシステム形態に分類できる。

第１は広域コンピュータネットワークによる分散システムB6,10）である。これは、地理的に広く分散した処理サイト間を、広域ネットワーク網や公衆通信網を介して結合させたシステム形態をとり、既存の装置やソフトウェア、データ等のリソース共有に主眼がおかれる。

第２は、ローカルエリアネットワークB12,13）による分散システムである。これは、近距離に散在する処理サイト間を比較的高速のデータ交換網で結合した形態であり、サイトの独立性を保ちつつ処理を分割実行したり、装置間の機能分担を行ったりすることが中心となる。

次に、第３はインハウスのマルチプロセッサA1~6)による分散システムである。これは、高速の内部バスを介してインハウスの装置が結合した形態をとり、機能分担や並列処理、高信頼性の保証等が主目的となる。

このような分散処理によるアプローチは、柔軟で経済的なシステムの構築法を提供するという意味で、情報処理という技術分野において極めて大きな意義を持つこととなった。そして今日このアプローチに込められる期待は増々大きなものとなりつつある。

一方、最近になって、情報処理の概念そのものを拡張し、従来のシステムでは扱えなかった高度の判断や処理にもコンピュータを適用しようとする試みが注目されるようになった。これは、知識情報処理とか、知識工学とか呼ばれる。[D6-10] 知識工学は人工知能研究から派生したひとつの実用技術であり、情報処理システムの質的向上を目指した、今日極めて大きな期待が込められた技術である。

知識工学は、専門家の知識をコンピュータに蓄え、利用し、高度の問題解決を支援するシステム（エキスパートシステム[D8,10] と呼ばれる）の構築を主目的とした技術分野といえる。ここでは、このような知的作業に対し、コンピュータの果し得る役割を再考してみよう。

たとえば、大須賀は、[D17] 設計作業のような高度の知的作業において人間が用いている知的機能として、以下の11の機能をあげている；すなわち、直観、創造、発想、帰納、認識、学習、連想、合成、演繹、検索、演算、といった諸機能である。

通常、問題解決の最終フェーズにおいては、反復計算や検索のような単純な知的作業が必要となる。このような煩しい作業から人間を解放したいという要求が、パスカルやライプニッツに始まる初期の計算機械を誕生させたといえる。たとえば、機械式計算機考察の動機として、ライプニッツは、"天文学者達を複雑な計算のために費やす時間と忍耐から救うため"という意図を持っていたといわれている。[E7] 今日のディジタルコンピュータは、スーパーコンピュータをみるまでもなく、単に計算するという機能だけについてみれば、人間の能力、忍耐力を大きく越えた作業能力を達成しており、この目的については正に完璧にその役割を果しているといえよう。

これに対し、問題をモデル化したり、解を求めるための方程式をたてるといった問題解決の初期のフェーズにおいては、人間は前述の直観から演算までのすべての機能を総動員

する。このうち特に、直観とか創造、発想といった機能は、全く未知であり、このような機能を用いる人間の知的行動自体もほとんど解明されていないといってよい。このようなメカニズムの解明には、心理学、認知科学、生理学、情報科学、哲学といった広い学問分野を統合した研究の成果を待つ必要があろう。一方、帰納、認識、学習、連想、合成、演繹といった機能は人工知能研究やソフトウェア科学、知識情報処理の格好の研究テーマとなっており、現在、様々なアプローチによる研究が進行しつつある。[D12,16~19)] 但し、今迄の所、新しくコンピュータによる機械的な実行が可能となった機能は、演繹だけである。その他の機能については解明されない部分も多く未だ研究段階といえる。

したがって演算、検索といった単純な機能の他に、新しくコンピュータの果し得る役割として追加された機能は演繹（推論）機能だけということになり、実用面を重視する知識工学においては、現在の所、この機能を中心としたシステム構築が主流となっている。

さて本研究を始めるに到った動機は、このような人間の行う知的作業のうち、問題解決と呼ばれるようなまだ解を求めるアルゴリズムが定まっていない高度の判断を要する作業、特に設計や計画策定、意志決定などのような知的作業を支援する情報処理システムの実現にあった。さらに、ＬＳＩ技術やマイクロプロセッサ技術等の動向をも考慮し、経済的にも意味のあるシステム構築を目指した。このため、既存の大型計算機や、専用マシンによるアプローチではなく、汎用のマイクロプロセッサによる分散処理アーキテクチャが研究の前提となった。

また実用的な問題解決支援システムの構築を目指すためには大量のデータを扱うことのできるシステム構造が不可欠と考え、知識情報処理技術の中でも特に、データベースと演繹推論機構とを結合したいわゆる演繹データベース技術[D1~3,20,21)] に研究の焦点がおかれることとなった。

本研究は、システム構築の手段としての分散処理と問題解決支援という目的を達成する手段としての知識情報処理とを同時に指向したものであり、これら２つの技術分野の接点として位置づけることができる。

一般に、高度のシステムを構築する際には、従来とは異なる新しい発想や、既存の技術を統合する新しいシステム化技術の確立が不可欠とされる。本研究は、このような観点からは、コンピュータネットワーク、分散処理、知識情報処理といった通信と情報処理における既存の諸技術分野を統合する新しいシステムの構築のためのひとつの試みでもある。

１．２　本研究の概説

本研究では、ＳＤ^3(System for Distributed Database with Deductive search mechanism) システムと呼ぶ、分散データベースに述語論理に基づく演繹推論機構を持たせたシステムのプロトタイプ（以降本システムと呼ぶ）の開発が行われたD22~24)。

本システムは、図１．１に示すような三層構造をとっている。

図１．１において、ネットワークサブシステムは、同軸ケーブルを通信媒体としたローカルネットワークの通信プロトコルを実行することにより、ノード間でのデータの交換や並列処理の実行を可能とする部分システムである。このネットワークサブシステムは、後述するように、ＩＳＯの参照モデルやＳＮＡのような標準ネットワークアーキテクチャと同様にレイヤ構成をとっており、下位二層はEthernetに準じたものに、上位層は分散データ処理を特に指向した独自アーキテクチャになっているB15,19,20)。

分散データベースサブシステムは、各ノードの保持する固有データや共有データを一貫性を維持させながら更新したり、ネットワークに分散したデータの統合的な検索を行ったりする部分システムである。

ＳＤ^{3}における分散データベースサブシステムは、トランザクションクラスの前解析による効率の良い同時実行制御、同報型ネットワークの利点を充分に生かしたデータベースアクセスプロトコルによるパフォーマンスの向上といった特長を持っているC29~33)。

分散型推論サブシステムは、問題解決のための演繹推論を分散型で実行する部分システムである。後述するようにこのサブシステムの特徴は、述語論理に基づく演繹推論機構をデータベースに組み入れたこと、コントロールのあいまいさを解決するためのノード間協調プロトコルを導入し、推論の並列実行を行っていること、推論の中途結果をトリガ機構の導入によりすべてデータベースに蓄積できるようにしたこと等である。

最後に応用プログラムは、特定応用向きに記述された手続きであり、本システムにおいては応用プログラムに対し演繹検索と単純検索という２種のインタフェースを提供しており、問題解決の段階に応じて推論サブシステムを起動させることも（演繹検索）、分散データベースサブシステムを起動させることも（単純検索）可能であるという特長を持っている。

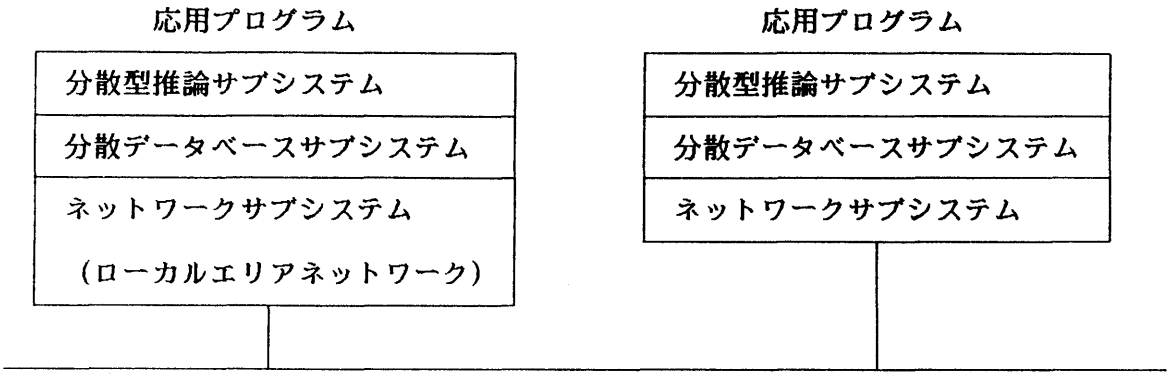


図１．１　ＳＤ^{3}システムの構造

次に本論文各章の概説を行う。第２章では、コンピュータネットワークについて、その背景、技術動向を述べ、本研究で開発したネットワークシステムの位置づけ、基本アーキテクチャについて議論する。第３章では、本研究における中核的な構成要素である分散データベースについて、その概念と特徴、主要な動向と技術課題について概説し、本研究における分散データベースシステムの位置づけとアーキテクチャについて考察する。次に第４章では、本研究により開発された推論システムの技術的背景としての知識情報処理について述べ、ついで演繹データベースシステムの主要な動向、分散型推論システムの研究事例等との対比により、本分散型推論システムの位置づけを試み、さらにその基本アーキテクチャについて考察する。

第５章では、本研究において応用まで含めてプロトタイプとして開発したシステムの詳細を述べ、本研究の主要課題であるデータベースと演繹推論機構との結合方式について、より具体的に議論する。

最後に第６章では、本研究により得られた要点を総括し、あわせて、今後の研究課題について論ずる。

第　2　章

コンピュータネットワークの
プロトコルと制御方式

本章では、SD^{3}システムにおけるネットワークサブシステム開発の背景となったコンピュータネットワーク技術（特にローカルエリアネットワーク）について、その歴史、近年の技術動向を概説し、あわせて本ネットワークサブシステムの位置づけについて論ずる。次に本ネットワークサブシステムの基本アーキテクチャについて考察を行う。

２．１　背景

広域に分散した多数の端末装置を、通信回線を介して中央のコンピュータに結合させ、通信とデータ処理を統合的に行うデータ通信システムは、1950年代に到って盛んとなった。

このデータ通信システムの発展過程において特に注目すべき動きが２つあった。ひとつは、1959年Ｃ．Ｓｔｒａｃｈｅｙにより提唱されたタイムシェアリングシステムで、遠隔地にある複数の端末装置と中央の汎用コンピュータとを通信回線でスター状に結合し、多数の利用者があたかも１つのコンピュータ専有しているかのように、同時利用する形態である。もうひとつは、この形態がさらに発展し、コンピュータの応用分野の多様化と拡大によって、データ通信システムの基本であったコンピュータ・端末装置間通信から、コンピュータ間通信へと発展した形態で、今日、コンピュータネットワークと呼ばれるものである。

このような形態のシステムは、1970年代に構築されたＡＲＰＡネットワークB6）を原点とみることができるが、その普及と発展は極めて迅速であり、今日多数のコンピュータネットワークが実現し、また国際間のコンピュータネットワークに関する標準化の動きも活発となっているB1~10)。

この国際標準コンピュータネットワークのアーキテクチャとして現在、日、米、欧で認められつつあるのが、ＩＳＯの Open System Interconnection（ＯＳＩ）参照モデルであるB10)。ＯＳＩ参照モデルは、各国で開発されたオペレーティングシステムやコード、プログラミング環境等の異なるコンピュータやシステム間の相互接続を可能とするため、アドレス体系やプロトコルの国際標準を目指して開発されたものである。

このモデルは図２．１に示すように現在７つの機能別に階層化した構成をとっており、そのそれぞれの層間でのプロトコルとヘッダ形式とが提唱されている（現時点では、トランスポートレベルまでが決定されている）。

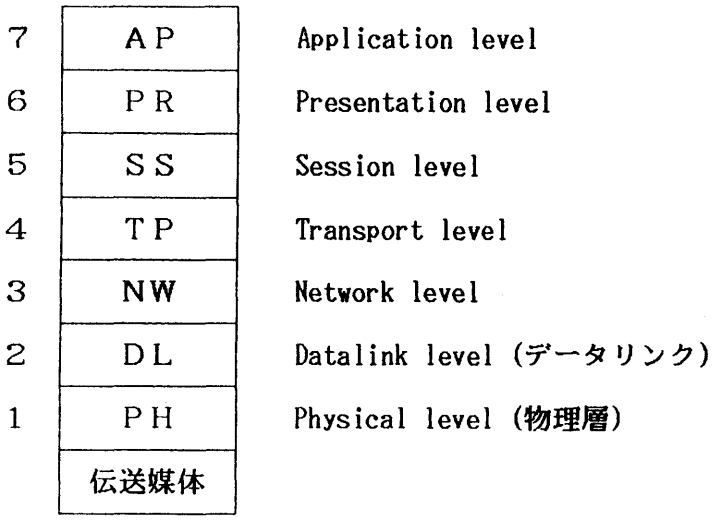


図2．1　OSI参照モデル

ここで、物理層は、例えば、コネクタの形状、電圧、信号線の使い方等、伝送媒体を接続するための電気的、機械的インターフェース条件を定めたものである。またデータリンク層は、リンク毎のフレームの送受信を行い、通信回線の伝送誤り制御等の手順を実行するものである。一方ネットワーク層は、複数の通信回線を経由して送受信を行うための、ルーチング方式等の標準を定めたものである。トランスポート層は、エンドノード間でのパケットの送受信を行うためのプロトコルの実行をB14,18)、セッション層は、エンドユーザ間でのコネクションの設定や解放及び送受信の管理を、プレゼンテーション層では、端末装置、ファイル等の差異を統合し、同一のものに見せるための仮想端末／仮想ファイルプロトコルB16,17) やコード変換、暗号化等の機能を実現するための層である。最後のアプリケーション層は、特定ユーザ向きの相互通信のためのプロトコルを実行する層となっている。

既開発のコンピュータネットワークアーキテクチャとして有力なものに、ＩＢＭ社のＳＮＡ B5)、ＮＴＴのＤＣＮＡ B9)等があるが、いずれもこれらコンピュータネットワークの目的は、遠隔地間のコンピュータ相互の通信を可能とさせ、かつ、距離にかかわりなくハードウェア、ソフトウェア、データ等のリソースを共有することを主眼としたものとなっている。

一方、このような遠隔地のリソース共有を目的としたコンピュータネットワーク（広域ネットワークと呼ばれる）に対して、比較的近距離のコンピュータやリソースを統合する閉域内でのコンピュータネットワーク、つまりローカルエリアネットワーク（ＬＡＮ）に対する関心が、近年特に高まりつつある。

このローカルエリアネットワーク（ＬＡＮ）は、広域のコンピュータネットワークによって提示されたアーキテクチャ、プロトコル等に基礎をおきつつも、広域ネットワークよりも一層高速で簡易な通信を閉地域内に於て実現しようとするもので、最近盛んに導入されつつあるオフィスオートメーション機器の統合に際して極めて大きな効果を発揮するものとして期待されている。

現在開発されているローカルエリアネットワークは、以下に示すように伝送媒体、形状、交換方式、アクセス方式の４つの観点から分類することができる。

２．１．１　伝送媒体

ローカルエリアネットワークで現在使用されている伝送媒体には、より線、シールド付より線、同軸ケーブル、光ファイバケーブルがあり、それぞれ表２．１に示すような特徴をもっている。

媒体	伝送速度	雑音の影響	特徴
より線対	1M b/s	大	安価、容易な配線工事
ｼｰﾙﾄﾞ付より線対	1M~10M b/s	小	より線対に比し外部雑音の影響小
同軸ｹｰﾌﾞﾙ	1M~50M b/s	小	ﾍﾞｰｽﾊﾞﾝﾄﾞ伝送、Ethernet等で使用
	(300 MHZ)		ﾌﾞﾛｰﾄﾞﾊﾞﾝﾄﾞ伝送
光ﾌｧｲﾊﾞｹｰﾌﾞﾙ	1M~100M b/s	なし	低伝送損失、分岐挿入困難

表２．１　ローカルエリアネットワークの伝送媒体とその特徴

２．１．２　ネットワーク形状（トポロジー）

ローカルエリアネットワークの形状には、スター形、ループ形、バス形の３種類があるが、ループ形ネットワークにおいては、送信ノードから受信ノードに直接データを送信できるリング形と、一旦コントローラを経由してから通信を行うループ形とを分けて、４種類とする場合もある。それぞれの方式の特徴および長所、短所を表２．２に示す。

形状	方式の特徴	長所	短所
スター	各ノードと中央のコントローラ間での通信。	大規模システムでは端末当りのコスト小。	初期コストが大（増設等を考慮した設備の要あり）。中央コントローラの障害によりシステムダウンとなる。
ループ	ループコントローラ(LC)経由で目的ノードと通信。	光ファイバの適用が容易・長距離も可能。	初期コストが大。LC又はノード障害でシステムダウンとなる。ノードの増設撤去等・システム変更が難。
リング	各ノード間で直接通信。中央コントローラなしの完全分散制御。	長距離も可能。光ファイバの適用が容易。	ノード障害でシステムダウンとなる。システム変更が難。
バス	中央コントローラなしの完全分散。各ノード間で直接通信。	コストが規模に比例(経済的システム構築)。1つのノード障害が全体に波及することはない。	光ファイバの適用が難。

表２．２　ローカルエリアネットワークの形状とその特徴

２．１．３　交換方式

大別して回線交換方式とパケット交換方式およびこれらを組合せたハイブリッド交換方式がある。２つの方式の特徴および、それらの方式の細分を表２．３に示す。

方式	方式の特徴	方式の細分	特徴
回線交換	通信の必要が生じたﾉｰﾄﾞ間にその都度回線を設定する。設定時間に比し通信時間の長い場合に適する。	空間分割方式	通信路、接点等を物理的に切分ける。
		時分割方式	同一通信路を時間帯を分けて使用する。
パケット交換方式	ﾒｯｾｰｼﾞを複数のﾌﾞﾛｯｸ(ﾊﾟｹｯﾄ)に分解、ﾊﾟｹｯﾄの宛先情報に従って着側ﾉｰﾄﾞまで届け、そこで元のﾒｯｾｰｼﾞを復元し、ﾕｰｻﾞｰに配送する。ﾒｯｾｰｼﾞ配送の実時間性が比較的ゆるい場合、多重化によって高効率の通信を実現できる。	データグラム	個々のﾊﾟｹｯﾄが独立した形で送られ、ﾌﾛｰ制御、ﾊﾟｹｯﾄの到着順制御は行わない。
		バーチャルサーキット	送受ﾉｰﾄﾞ間で仮想的な回線を設定する。ﾌﾛｰ制御、ﾊﾟｹｯﾄの到着順制御を行うため、ﾃﾞｰﾀｸﾞﾗﾑより信頼性は高いが仮想回線の設定、切断の為のｵｰﾊﾞﾍｯﾄﾞがある。

表２．３　ローカルエリアネットワークの交換方式

２．１．４　アクセス方式

ローカルエリアネットワークにおいて、複数のノードが同一の伝送路を使用して通信を行おうとする際、伝送要求の衝突を回避する為、何らかの制御が必要となってくる。このとき、送信を要求するノードが送信権を取得する方法は、中央コントローラにより送信ノードを決定する集中型と、そのようなコントローラのない分散型とに大別される。

分散型アクセス方式は、衝突の発生の有無により、図２．２に示すように解決方式とコンテンション方式とに大別され、コンテンション方式は、複雑な制御を行わない為、経済的であり、伝送路の容量に比べトラフィック量が少ない場合に適したものとなっている。

なお、個々の方式の詳細については、本研究の主題とは離れる為、本稿では論じない。

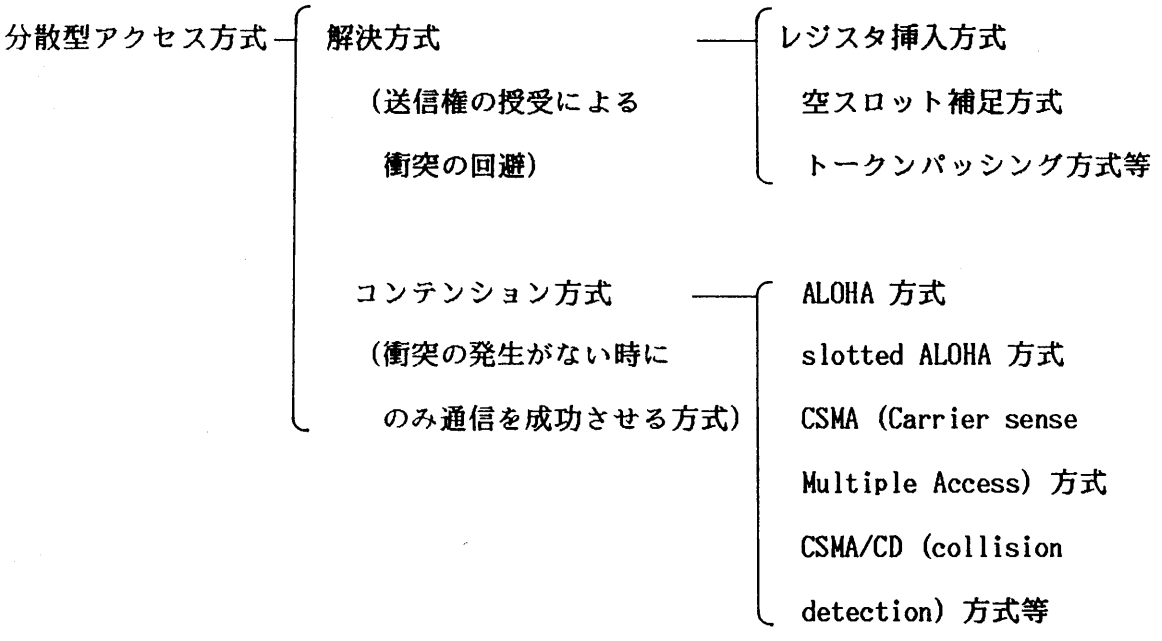


図２．２　分散型アクセス方式の分類

２．１．５　ローカルネットワークの標準化

広域ネットワークと同様ローカルネットワークに対しても、いくつかの委員会により標準化が試みられている。その主なものはＩＥＥＥ８０２委員会、ＥＣＭＡ（欧州電子計算機工業会）等で試みられているものである。

図２．３に示すように、ＩＥＥＥ８０２委員会では、ＯＳＩモデルの下位２層に相当する部分の標準化が行われている。対象となっている方式は、ＣＳＭＡ／ＣＤ、トークン・バス（アクセス方式がトークンパッシングで形状はバス）、トークン・リング（アクセス方式がトークンパッシングで形状はリング）である。さらに将来には、広帯域同軸方式も対象となる予定である。

一方、ＥＣＭＡにおける標準化は、第一層の物理レベルから第四層トランスポートまでを対象としている。

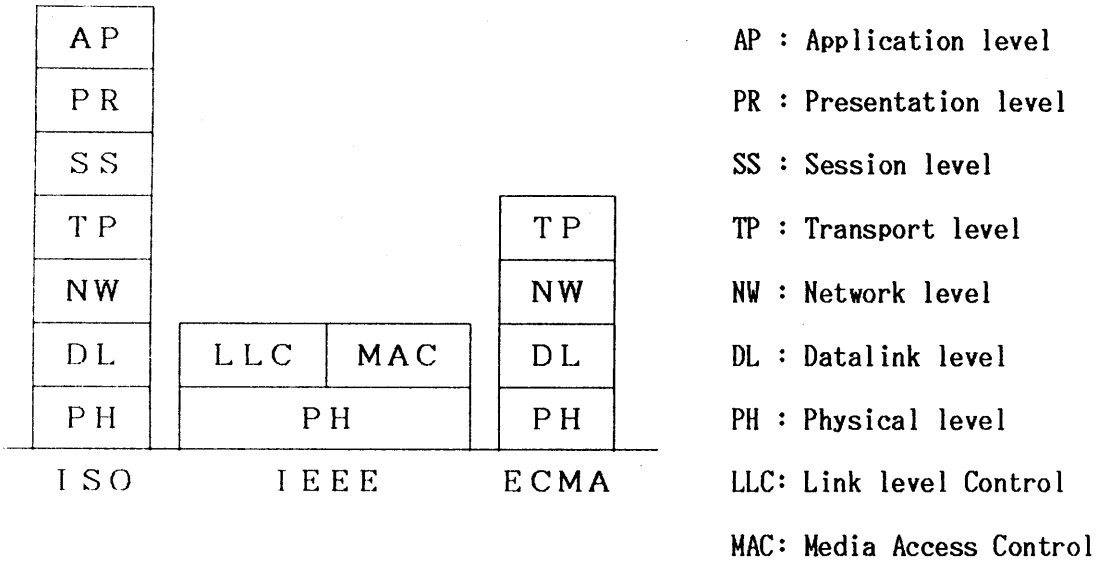


図２．３　ローカルネットワークの標準化

一般にローカルネットワークの標準化は、広域ネットワークのそれ程、作業が進行しておらず、特に高位レベルのプロトコルについては、各メーカが各々独自の使用を実現しているのが現状である。

その理由の１つは、ローカルネットワークに於ては、異機種コンピュータ間通信、リソースの共有といった目的が広域ネットワーク程強いものとはなっていないということが予想される（このような状況を反映して、本システムに於ても、下位２層は標準プロトコルを、上位は独自のアーキテクチャとプロトコル仕様が実現されている）。

２．２　本章の位置づけ

本サブシステムの開発に当っては、遠隔地にあるリソースの共有という観点よりも、近距離におかれた各プロセッサを、より高速のネットワークで結合し、並列処理を行うことにより、効率の良い分散データベースシステムを開発することに重点がおかれた。このため、コンピュータネットワークとしては、当初からローカルエリアネットワークに限定した。

本ローカルエリアネットワークの下位２層は、ＩＥＥＥ８０２委員会においても標準化の対象としてとりあげられているＣＳＭＡ／ＣＤタイプのもの、より具体的には、Ethernetを採用した。その基本的特徴は次の通りであるB13)。

伝送媒体　：　同軸ｹｰﾌﾞﾙ（伝送速度10Mbps,ﾍﾞｰｽﾊﾞﾝﾄﾞ伝送）

ﾈｯﾄﾜｰｸ形状　：　ﾊﾞｽ形

交換方式　：　ﾃﾞｰﾀｸﾞﾗﾑ形　ﾊﾟｹｯﾄ交換

ｱｸｾｽ方式　：　CSMA/CD (Carrier Sense Multiple Access/Collision Detection)

なお、下位２層にこの方式を選定したことの主な理由は以下である：

（１）雑音の影響が少なく、また比較的経済的にネットワークを構成でき、伝送速度も10Mbpsと充分高速なこと，

（２）分散データベースを構築するためには、伝送遅延が少なく、かつ同報通信の実現が容易なバス形ネットワークが適していると判断したこと，

（３）音声の様な実時間性の厳しい情報をネットワークに通すことは当初から考えておらず、又処理の進行に伴って通信相手が頻繁に変わるため、接続設定に要するオーバーヘッドの少ないデータグラム型パケット交換方式が適切であると判断したこと，

（４）複雑な衝突回避の制御を行わないため、比較的経済的に装置を実現でき、しかも、データリンクレベルにおける衝突の検出を行うため、通信の効率が良いとの理由からＣＳＭＡ／ＣＤ方式が適切であると考えられたこと，

等である。なお、この他にEthernetは、1979年のspec公開以来、各方面で広範に採用されており、装置、ＬＳＩ等の入手が最も容易であったということも、選定に際しての大きな理由となっている。

一方、本ローカルエリアネットワークの上位層は、分散データベース構築を意図した極めて特徴ある独自のアーキテクチャを採用している。

パケット交換をベースとした上位層のアーキテクチャとしてはＩＳＯのＯＳＩをはじめ様々なものが提案されているが、これらは、分散データベースや分散ファイル等の利用形

利用形態に必ずしも適合したものとはなっていない。特にこれらのネットワークアーキテクチャに欠如している機能として、ネットワークに接続する特定グループ内での同報通信機能がある。

既存のコンピュータネットワークにおいては、ネットワーク層、トランスポート層、セッション層といった上位層において、バーチャルサーキット型の交換機能や"セッション"あるいは"コネクション"といったエンド　ツウ　エンドの伝送を保証する機構を持っているが、これらはいずれも１対１通信の概念に基づいたものである。

一方分散されたデータベース、ファイルを扱う際、同報通信が有効なことは、図２．４の例からも明らかである。

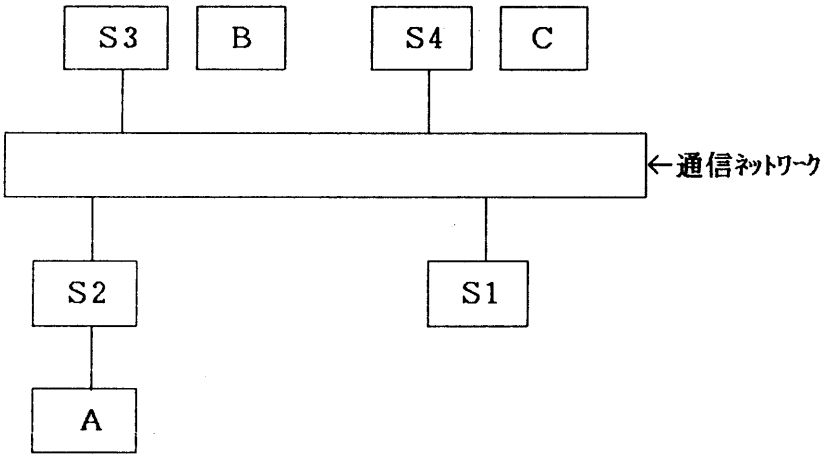


図２．４　同報通信の必要性

今ステーションS1から、S2、S3、S4におかれたデータA、B、Cを更新するトランザクションを送信する場合を考える。このときデータの更新は、一貫性を持って行わなければならない。つまり、更新トランザクションがステーションS2、S3、S4のすべてに到着すること（あるいはどの１つにも到着しないこと）を保証しなくてはならない。通常のバーチャルサーキットタイプのネットワークにおいては、S1から各ステーションに対し計３本のバーチャルサーキットを設定し、更新トランザクションを個々のサーキットを通して転送することになる。したがって、特定ステーションが障害となった場合、各々の

サーキットは独立であるから、それ以外のステーションに対しても、更新トランザクションは送信されてしまうため、更新の一貫性を保証するためには、かなり複雑な処理を必要とし、また非効率なものとなる。これに対し、S2、S3、S4というグループに対するエンド　ツウ　エンドの伝送を保証する機構を通信ネットワーク側が持っていれば、更新の一貫性の保証は極めて効率的であり、かつ単純なものとなる。

これが、バス型のローカルネットワークが採用されたこと、およびグループ内同報通信という従来みられなかった新しい機能がネットワークに組入れられることとなったことの主な理由である。

なお、本システムに於ては、技術的、経済的理由のため実現できなかったが、分散データベースに特に適したＬＡＮという条件を考慮すると、ＬＡＮ側で持つと望ましいと思われる機能は、この他以下のものが考えられる。

（１）持続的通信機能

長期障害に対処するため、隔離された障害サイトへの伝送データの蓄積と回復時の受信を保証する機能で、ネットワーク側にメッセージ蓄積機能を持つ。

（２）障害サイトおよびネットワーク分割の検出と隔離機能

長期障害となったサイトの検出と、サイトもしくは通信リンク障害に基づくネットワーク分割の検出機能。

（３）暗号化／復号化機能

分散されたノード間のデータの送受信に対するセキュリティ維持が特に重要となるため、ネットワーク側に暗号化／復号化機能を備えることは望ましい条件となる。

２．４　ローカルネットワークサブシステムに関する基本的考察

ＳＤ^{3}におけるローカルエリアネットワークの上位層は、ＢＡＮＥＴ（Broodcast Architecture NETwork）と呼ばれる分散データ処理指向の独自のネットワークアーキテクチャとなっている。

本節では、このネットワークアーキテクチャの概要を論ずる。

２．３．１　ＢＡＮＥＴの特長と機能

［１］ＢＡＮＥＴの特長

ＢＡＮＥＴの最大の特長は、グループ通信機能を提供していることと、コミットメント制御機構B11)をネットワークの基本的な機能として位置付け、組み込んでいることにある。

１　グループ通信

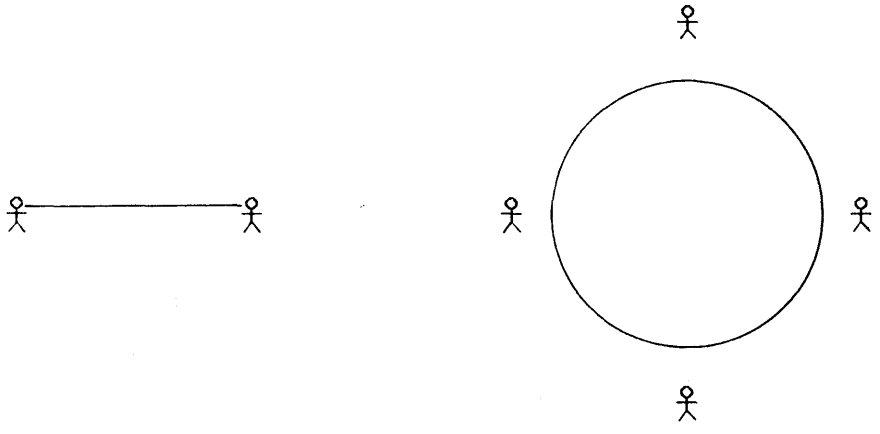
従来提案されてきたほとんどのネットワークアーキテクチャは、２つのエンドプロセス間にセッションあるいはコネクションと呼ばれる論理的なリンクを設定し、その論理リンク上でエンド　ツウ　エンドの伝送を保証するという１対１通信の概念に基づいている。しかし分散されたファイルやデータベースを扱う分散処理システムにおいては、１対１通信にかわって、複数のエンドプロセス間で通信のグループを形成し、グループ内でデータを同報通信する機能が必須になってくる。このようなグループ通信機能は丁度、複数の人間がテーブルを囲んで会話するような新しい通信の概念といえるB15,19,20)(図２．５)。

ＢＡＮＥＴは、このような"通信グループ"をダイナミックに形成し、そのグループ内でデータを同報通信する機能を提供している。たとえば、４つのプロセスが互いに連係をとりながら１つの仕事をする場合、従来の１対１通信ではこれらのすべてのプロセス対間に論理リンクを設定しなければならず、合計６つの論理リンクが必要となる。一方、ＢＡＮＥＴではこれら４つのプロセス間で１回だけ通信グループの形成を行えばよい（図２．６）。さらに、１つのプロセスが他の３つのプロセスに同一データを転送したい場合、従来のネットワークでは個々の論理リンクを通して３回同じデータの転送をしなければならなかった。ＢＡＮＥＴでは１度通信グループを形成すればよく、あとは１回だけそのグループ内にデータを送出すれば、すべてのグループのメンバにデータがブロードカストされることになる。

このように、ＢＡＮＥＴのグループ通信機能は、従来のネットワークに比し大巾にデータ転送量を軽減させる効率のよいデータ転送を可能としている。

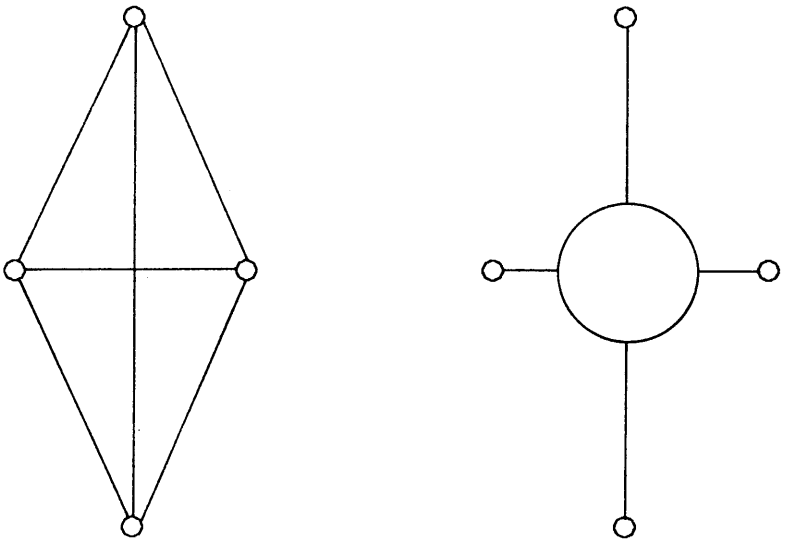
グループ通信は従来の１対１通信の概念を拡張したものであり、通信グループが２つのエンドプロセスから構成される場合には１対１通信そのものになっている。

たとえば従来の１対１通信では片方向、半二重、全二重の３種類の送信権モードが考えられているが、ＢＡＮＥＴではその概念を拡張して、固定送信元転送（通信グループの形成を要求したプロセスのみがデータ転送できる）、可変送信元転送（通信グループのどのメンバもデータ転送可能であるが一時には一つのメンバしかデータ転送できない）、多重送信元転送（通信グループのどのメンバもいつでもデータ転送できる）の３つの送信権モードをもうけている。



（ａ）１対１通信　　（ｂ）グループ通信

図　２．５　１対１通信とグループ通信



(a)　　(b)

図 2．6　4つのプロセス間の通信

② コミットメント制御

分散されたデータを更新しようとする場合、あるデータは正常に更新されたが、あるデータは更新できなかったというような事態が発生すると、データの一貫性が保たれなくなってしまう。そこで、このような場合には、全てのデータが正常に更新されるか、あるいは全てのデータが更新されないかのどちらかであることを保証する機構が必要となる。

さらに、このようなデータの更新処理に限らず、一般的に複数のプロセスでひとまとまりの仕事を分散して処理する場合、あるプロセスは正常に処理を実行したが、あるプロセスは処理を実行できなかったというような事態を避けるため、全てのプロセスが正常に処理を実行するか、あるいは全てのプロセスが全く処理を実行しないかのどちらかであることを保証する機能が必要となる。これはコミットメント制御B11,15) と呼ばれ、通常は2フェーズコミット法と呼ばれる方式により実現される。

従来のコンピュータネットワークにおいては、このような制御はアプリケーションプログラムにまかされていたが、ＢＡＮＥＴにおいては、この制御はリライアブルな同報通信機能を提供するという意味で本質的には通信の機能であると考えた。従って、これをネットワーク側の機能としてグループ通信機能の一つに組み込んでいる。

［２］ＢＡＮＥＴの機能

ＢＡＮＥＴの機能には以下のものがある：

① 通信グループの形成・消滅，

② グループ内データ転送，

・コミットメントモード転送

・モニタリングモード転送

・トランザクションモード転送

③ グループ外データ転送

・モニタリングモード転送

・トランザクションモード転送

ＢＡＮＥＴのグループ内データ転送ではコミットメントモード、モニタリングモード、トランザクションモードの３種類のデータ転送モードが用意されており、アプリケーションプログラムが用途に応じて任意に選択できるようになっている。

コミットメントモードでは、２フェーズコミット法に基づきデータ転送が実行される。

モニタリングモードでは、送達確認機能が提供される。このモードでは、データを受信した各メンバはＡＣＫを返し、ＡＣＫを返さなかったメンバはアプリケーションプロセスに通知される。

トランザクションモードでは各メンバからＡＣＫは返されず、エラー等の検出されたデータは廃棄される。このモードのデータ転送は、それほど信頼性の必要のない場合やアプリケーションレベルで応答を行う場合などに使用される。

２．３．２　システム構成

図２．７にＢＡＮＥＴのシステム構成例を示す。

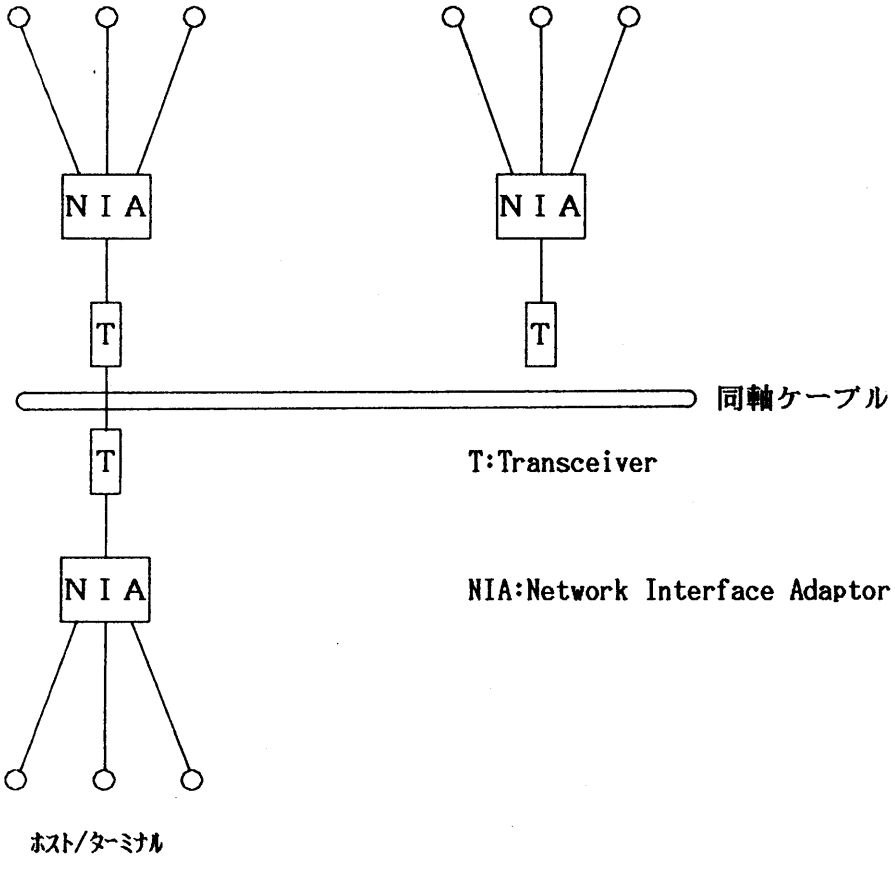


図２．７　ＢＡＮＥＴのシステムの構成例

同軸ケーブルとトランシーバの仕様はEthernetと同様である。ＮＩＡ(Network Interface Adaptor)にはＲＳ２３２Ｃ等の汎用インタフェースを介して複数のホスト計算機や端末を接続することができる。

２．３．３　プロトコル

ＢＡＮＥＴのプロトコル階層を図２．８に示す。

プロトコル階層のうち、物理レベル、データリンクレベルはEthernetと同様ＣＳＭＡ／ＣＤ方式のバス型アーキテクチャになっている。

ブロードカストレベル（ＢＬ）がＢＡＮＥＴの中核をなす階層であり、通信グループの形成・消滅およびグループ内／グループ外の同報通信機能を提供している。

［１］通信グループの形成

通信グループの形成を要求するプロセスは、ＮＩＡにＣＧＦＯＲＭコマンドを発行する。このとき、形成されるグループの候補メンバのリストを付与する。また、候補メンバ全てが通信グループに加わることが必須であるか否かを指定することもできる。

ＮＩＡはcgformパケットをネットワーク内の全ＮＩＡにブロードカストする。cgformパケットを受信した各ＮＩＡは、候補メンバリストに自分の支配下のターミナル内のプロセスが入っているか否かをチェックし、入っている場合にはそのターミナルにＣＧＦＯＲＭコマンドで通信グループ形成要求を通知する。

通信グループ形成要求を通知された候補メンバプロセスは、通信グループ形成条件や内部状態等をチェックし、通信グループに加わるか否かを決定する。通信グループに加わる場合にはＣＧＡＣＫコマンドを、加わらない場合にはその理由を付けてＣＧＮＡＣＫコマンドを自ＮＩＡに発行する。ＮＩＡではそれぞれのコマンドに対応してcgackまたはcgnackを通信グループ形成要求元ＮＩＡに返送する。

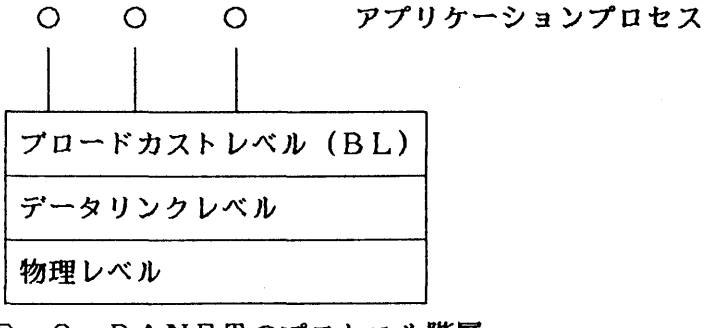


図２．８　ＢＡＮＥＴのプロトコル階層

通信グループ形成要求元ＮＩＡは各候補メンバからの応答を監視する。以後の手順は、応答の状況（すべての候補メンバプロセスがcgackを返してきたか、cgnackを返してきたプロセスがあるか、タイムアウトで無応答となったプロセスがあるか）や全メンバ必須か否か、送信権モードが何であるか（固定送信元、可変送信元、多重送信元）および通信グループのメンバ数が２かそれより大きいかにより異なる。

まず、通信グループのメンバ数が２の場合には、応答がcgackのとき正常終了をＲＣＧＦＯＲＭコマンドで要求元プロセスに通知し、通信グループの形成を終了する。応答がcgnackのとき、あるいは無応答のときは形成失敗を要求元プロセスに通知する。

通信グループのメンバ数が２でなく、送信権モードが可変送信元あるいは多重送信元の場合、全ての候補メンバからcgackが返ってきたときは全ての候補メンバをメンバリストにのせて、また全メンバ必須が指定されていず、かつcgnackを返してきたものがある場合は、cgackを返してきたメンバのみでメンバリストを作成し、cgmemberパケットでそれを通信グループ内の全メンバに通知する。一方各メンバプロセスの存在するＮＩＡは、各メンバプロセスにＣＧＭＥＭコマンドで通信グループのメンバを通知し、送信元ＮＩＡにcgmember-rackパケットを返送する。送信元ＮＩＡは各メンバからの応答を監視し、全てのメンバからcgmember-ackパケットが送られてきた時点で要求元プロセスにＲＣＧＦＯＲＭコマンドで通信グループの形成終了を通知する。

上記の手順において、送信権モードが固定送信元のときは、メンバリストの通知手順は省略される。

全メンバ必須でcgnackを返してきたものがある場合、あるいはタイムアウトになったものがある場合には、通信グループの形成を断念し、通信グループ消滅手順に入り、通信グループ消滅後、その旨をＲＣＧＦＯＲＭコマンドで要求元プロセスに通知する。

図２．９に通信グループの形成手順例を示す。図２．９でＡ（・）はアプリケーションプロセスを、Ｎ（・）はＮＩＡを表す。また、Ｓは通信グループの形成要求もと（イニシエータ）を、Ｒｉは形成要求の受信者（レスポンダ）を表している。

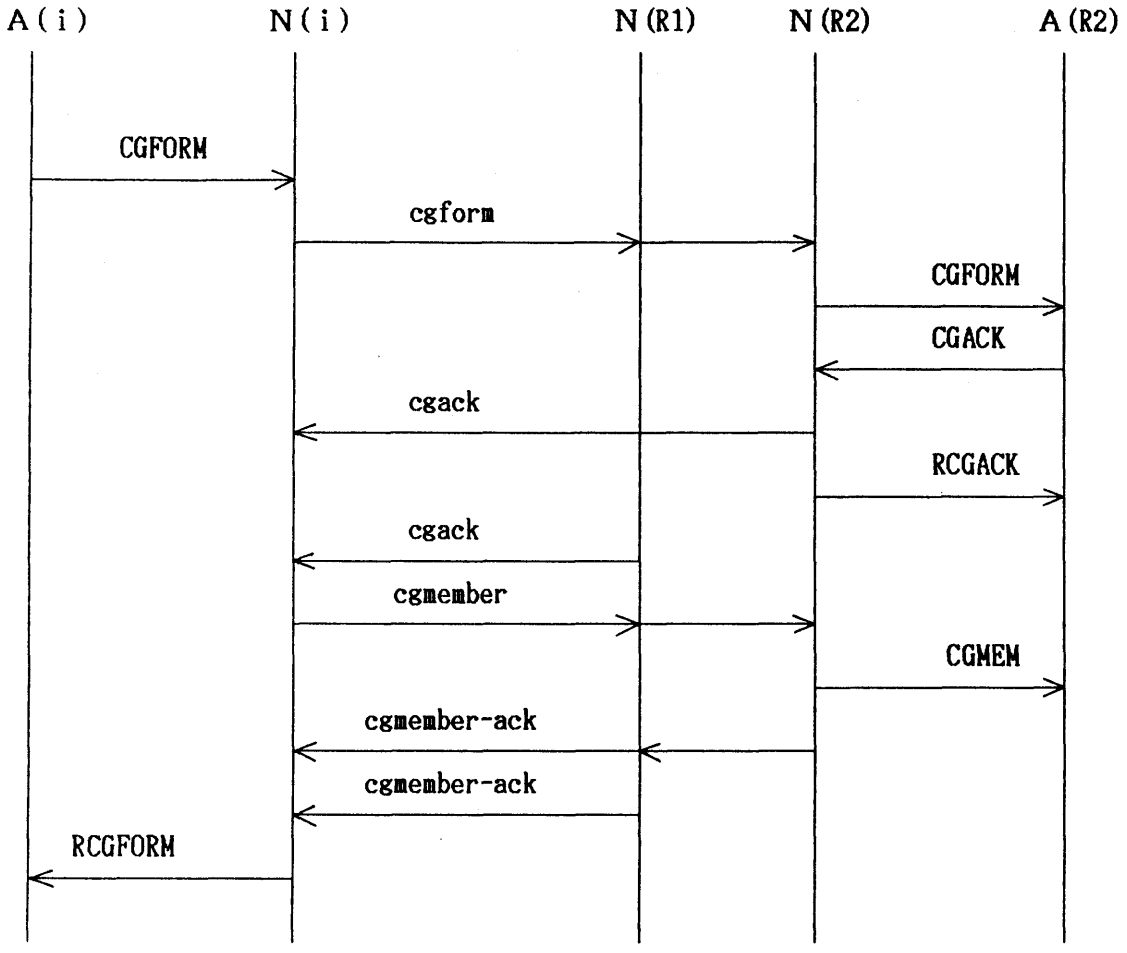


図２．９通信　グループの形成手順

［２］通信グループの消滅

通信グループの消滅を要求するメンバプロセスはＮＩＡにＣＧＤＲＯＰコマンドを発行する。これを受けたＮＩＡは、メンバプロセスの存在する各ＮＩＡにcgdropパケットをブロ

ードカストする。cgdropを受信したＮＩＡはメンバプロセスの存在するターミナルにＣＧＤＲＯＰコマンドで通信グループの消滅を通知するとともに、要求元ＮＩＡにcgdrop-ackパケットを返送する。要求元ＮＩＡはcgdop-ackパケットを監視し、全メンバから応答が返ってきた時点で、ＲＣＧＤＲＯＰコマンドで要求元プロセスに通信グループの消滅完了を通知する。

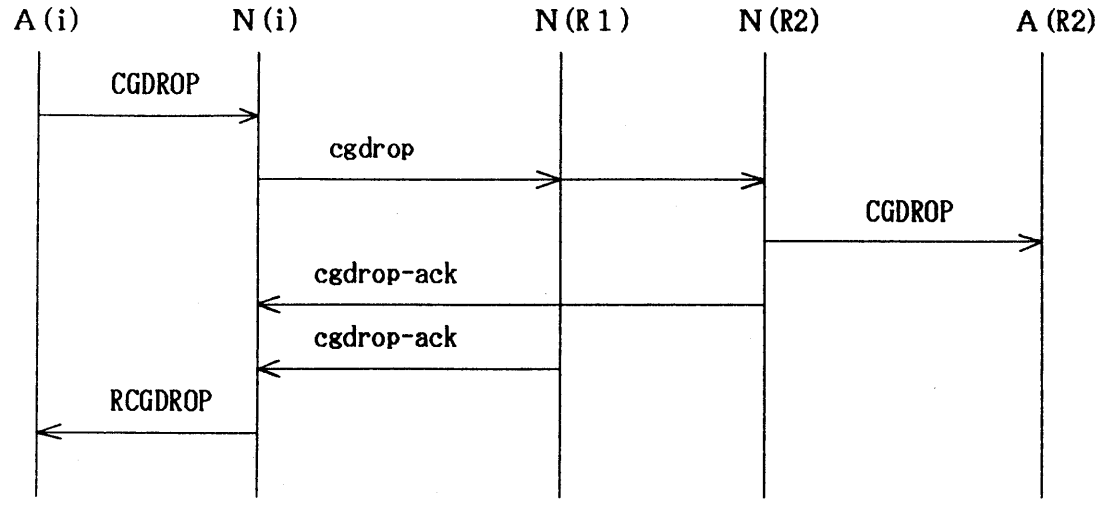


図２．１０　通信グループの消滅手順

［３］グループ内データ転送

① コミットメントモード

コミットメントモードによるデータ転送を要求するプロセスは、ＮＩＡにＧＣＯＭＭＩＴコマンドを発行する。これを受けてＮＩＡは、各メンバプロセスの存在するＮＩＡにsecureパケットおよび送信データをブロードカストする。

secureパケットを受信したＮＩＡは、メンバプロセスにＳＥＣＵＲＥコマンドでデータを引き渡す。データを渡されたプロセスは、もし、受信データの処理を保証できるならばＳＥＣＵＲＥＤコマンドを自ＮＩＡに返す。もし、受信データの処理を何らかの理由で保証することができないならば、ＮＯＴ－ＳＥＣＵＲＥＤコマンドを自ＮＩＡに返送することになる。ＮＩＡはそれぞれのコマンドに応じてsecure-ackパケット(ＳＥＣＵＲＥＤの場合)、またはsecure-nackパケット(ＮＯＴ－ＳＥＣＵＲＥＤの場合)を送信元ＮＩＡに応答する。

送信元ＮＩＡではsecureパケットに対する応答を監視する。全てのメンバからsecure-ackパケットが返ってきたならばcommitパケットを、ひとつでもsecure-nackを返してきたり、あるいはタイムアウトになったメンバがあるならば、recoverパケットを各メンバにブロードカストする。

commitあるいはrecoverパケットを受信したＮＩＡは、受信したパケットのタイプに応じてＣＯＭＭＩＴコマンドあるいはＲＥＣＯＶＥＲコマンドをメンバプロセスに指示する。ＣＯＭＭＩＴを通知されたプロセスは先にＳＥＣＵＲＥコマンドで渡されたデータに対する処理を完了させる。ＲＥＣＯＶＥＲを通知されたプロセスは先に渡されたデータを廃棄する。

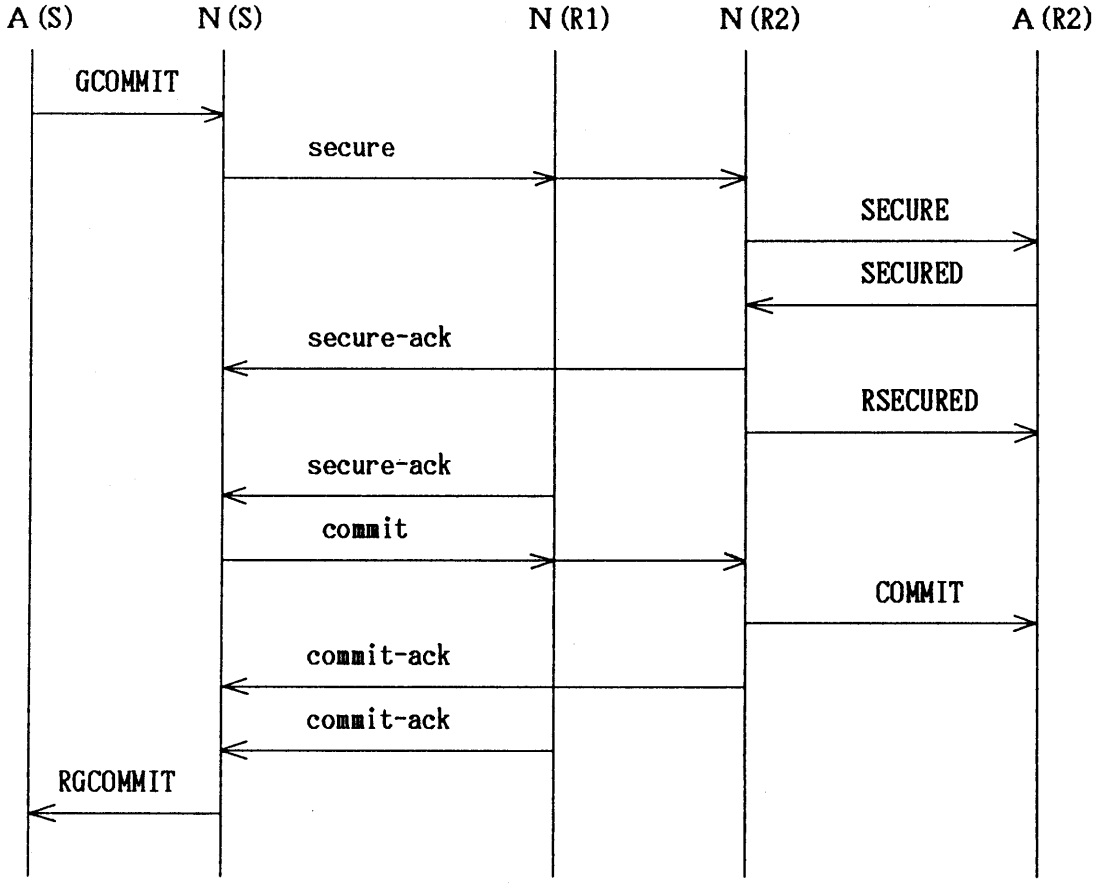


図２．１１　コミットメントモードデータ転送（コミットの場合）

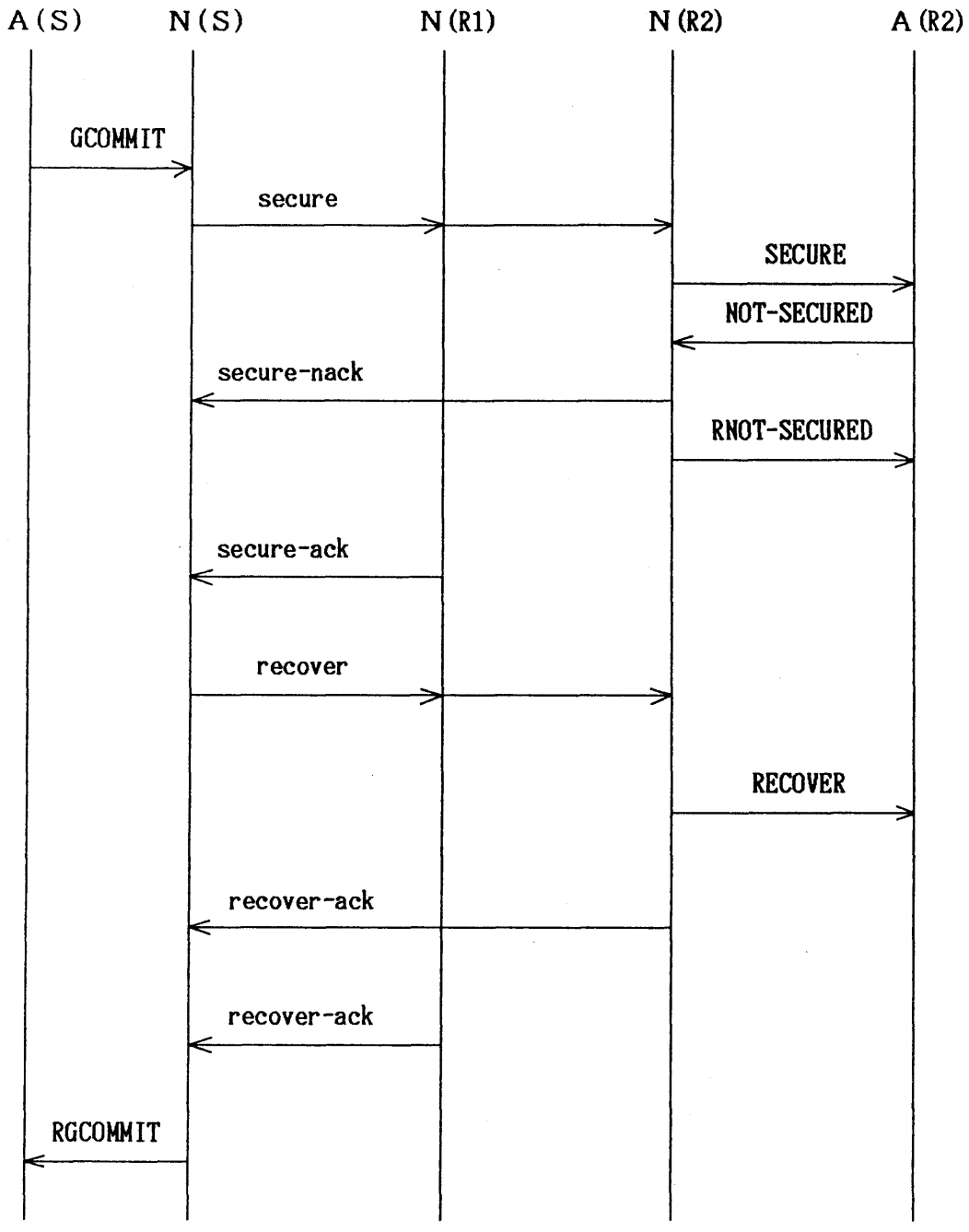


図２．１２　コミットメントモードデータ転送（リカバの場合）

② モニタリングモード

モニタリングモードによるデータ転送を要求するプロセスは、ＮＩＡに対してＧＭＯＮＩＴコマンドを発行する。これを受けたＮＩＡは、各メンバプロセスの存在するＮＩＡに、gmonitパケットをブロードカストする。

gmonitパケットを受信したＮＩＡは、メンバプロセスにＧＭＯＮＩＴコマンドでデータを引き渡すとともに、送信元ＮＩＡにgmonit-ackパケットを返送する。

送信元ＮＩＡはgmonit-ackパケットを監視し、全てのメンバプロセスからgmonit-ackパケットが返ってきた時点で、要求元プロセスにＲＧＭＯＮＩＴコマンドで転送終了を通知す

る。

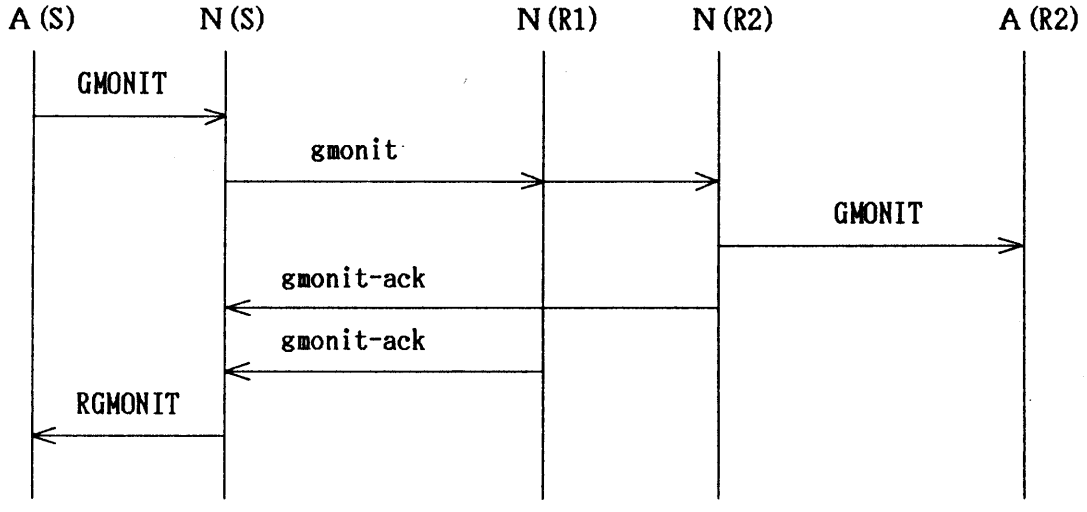


図２．１３モニタリングモード転送

③ トランザクションモード

トランザクションモードによるデータ転送を要求するプロセスは、ＮＩＡにＧＴＲＡＮＳコマンドを発行する。これを受けたＮＩＡは、各メンバプロセスの存在するＮＩＡに、gtransパケットをブロードカストする。トランザクションモード転送では送達確認機能がないので、この時点で要求元プロセスにＲＧＴＲＡＮＳコマンドを返す。

gtransパケットを受信したＮＩＡは、メンバプロセスにＲＧＴＲＡＮＳコマンドでデータを引き渡す。

［４］グループ外データ転送

① モニタリングモード

グループ外のモニタリングモードによるデータ転送を要求するプロセスは、ＮＩＡにＭＯＮＩＴコマンドを発行する。ＭＯＮＩＴコマンドには宛先プロセスのアドレスリストが付与される。これを受けたＮＩＡは、全ＮＩＡにmonitパケットをブロードカストする。

monitパケットを受信したＮＩＡは、宛先プロセスリストに自分の支配下のプロセスが入っているか否かをチェックして、もし入っていれば当該プロセスにＭＯＮＩＴコマンドでデータを引き渡すとともに、送信元ＮＩＡにmonit-ackパケットを返送する。

送信元ＮＩＡはmonit-ackパケットを監視し、全ての宛先プロセスからmonit-ackが返ってきた時点で、要求元プロセスにＲＭＯＮＩＴコマンドで転送終了を通知する。

② トランザクションモード

グループ外のトランザクションモードによるデータ転送を要求するプロセスは、自ＮＩＡにＴＲＡＮＳコマンドを発行する。ＴＲＡＮＳコマンドには宛先プロセスのアドレスリストが付与される。これを受けたＮＩＡは、全ＮＩＡにtransパケットをブロードカストする。

トランザクションモード転送では送達確認機能がないので、この時点で要求元プロセスにＲＴＲＡＮＳコマンドを返す。

transパケットを受信したＮＩＡは、宛先プロセスリストに自分の支配下のプロセスが入っているか否かをチェックして、もし入っていれば、当該プロセスにＴＲＡＮＳコマンドでデータを引き渡す。

第　3　章

分散データベースの技術課題と
制御アルゴリズム

本章では、ＳＤ^{3}システムの中核的な構成要素である分散データベースシステムについてその技術の背景、概念と特徴、主要な動向と技術課題について概観する。

次に、本分散データベースサブシステムの位置付けと、その基本アーキテクチャについて考察する。

３．１　背景

コンピュータネットワークの普及と進展に伴って、ネットワーク上にファイルあるいはデータベースを分散させるという試みが1975年頃よりなされるようになったC17)。このようなシステムつまり分散データベースシステムは、地理的、物理的に分散しているデータベースを論理的には単一のデータベースシステムとしてとらえるというアプローチC1）であり、リソースをなるべく多くのユーザに共有させるというコンピュータネットワークの観点からも自然な要求であった。

共有すべきリソースとしてのデータベースの重要性は、近年、増々顕著となってきている。その主な理由の１つは、ＶＬＳＩ技術を初めとするデバイス技術の進歩により、プロセッサ、メモリ素子等のハードウェアが増々低コスト化され、その結果、高処理能力を持つパーソナルコンピュータさえ出現するようになってきていること、つまりコンピュータの処理能力は、かつての共有リソースとしての意義を次第に失いつつあり、かわりにデータやプログラムといった情報そのものの価値が高まりつつあること、また他の１つは、通信技術の進歩とコンピュータネットワークの普及により、リソース共有の概念に基づく分散システムの実現というアプローチが増々魅力あるものとなってきていることの２点である。

このため、最近のオフィス情報システム、設計支援システム、および意志決定支援システム等においては、データベースが中核的な位置を占めるようになってきている。

処で、分散データベースの研究開発は、1975~1980年頃までに行なわれた第１期のシステムと、1980年以降現在まで研究が継続している第２期のシステムとに分けて考えることが可能である。

第１期システムの代表的例は、ＣＣＡ(Computer Corporation of America)社のＳＤＤ－１C2,3,19,20)，カリフォルニア大バークレー校のDistributed INGRES C4,22),仏グルノーブル大のPOLYPHEME等である。これら第１期システムは、データベース管理システム（ＤＢＭＳ）のモデル、言語が同一である（Homogeneous）、ネットワークとしてはARPANET,CYCLADE B3）等の広域パケット網を使用する等の特徴を持っている。

一方第２期のシステムの代表例は、ＣＣＡのMulti-base,仏・ＩＮＲＩＡのSIRIUS-DELTA,ＩＢＭのSystem R*等である。第２期システムは、異なるＤＢＭＳの統合(Heterogeneous)も考える、広域網のみならずローカルエリアネットワークも使用する、パーソナルコンピュータの使用とオフィスシステムへの応用を指向するといった特徴を持っている。

第２期システムでは、その主適用領域としてオフィスシステムを想定しているため、文書、図形、音声等様々な形式のデータをどのように扱うか、また設計支援や意志決定支援等への適用が可能なように、データベースシステムをどのように高度化していくかも重要な技術課題となっている。

以降ではまず分散データベースの概念とその特徴を概説し、次に、分散データベース構築として現在までに提案されたアプローチについて、論ずることとする。

３．２　分散データベースの概念と特徴

分散データベースの概念は、図３．１に示すようになる。すなわち、各データベース管理システム（ＤＢＭＳ）は、各々のローカルデータベースを管理し、またＤＢＭＳどおしは、通信回線で結合される。ユーザもしくはアプリケーションプログラムは、通信回線を経由して全ローカルデータベースをアクセスすることが可能である。このときに、各ローカルＤＢＭＳは同一データのコピーを重複して保持することもある。

ローカルＤＢＭＳの集合を論理的に単一のデータベースとして見せるためには、ロケーショントランスペアレンシーと呼ぶ概念C1）が必要とされる。これはユーザもしくはアプリケーションプログラムが、自分の必要とするデータの所在サイトを知らなくとも、そのデータへのアクセスを可能とするもので、コンピュータ間通信という観点からこれを眺めると、従来のような通信相手が一意に定まっている通信形態とは根本的に異なるものとなる。

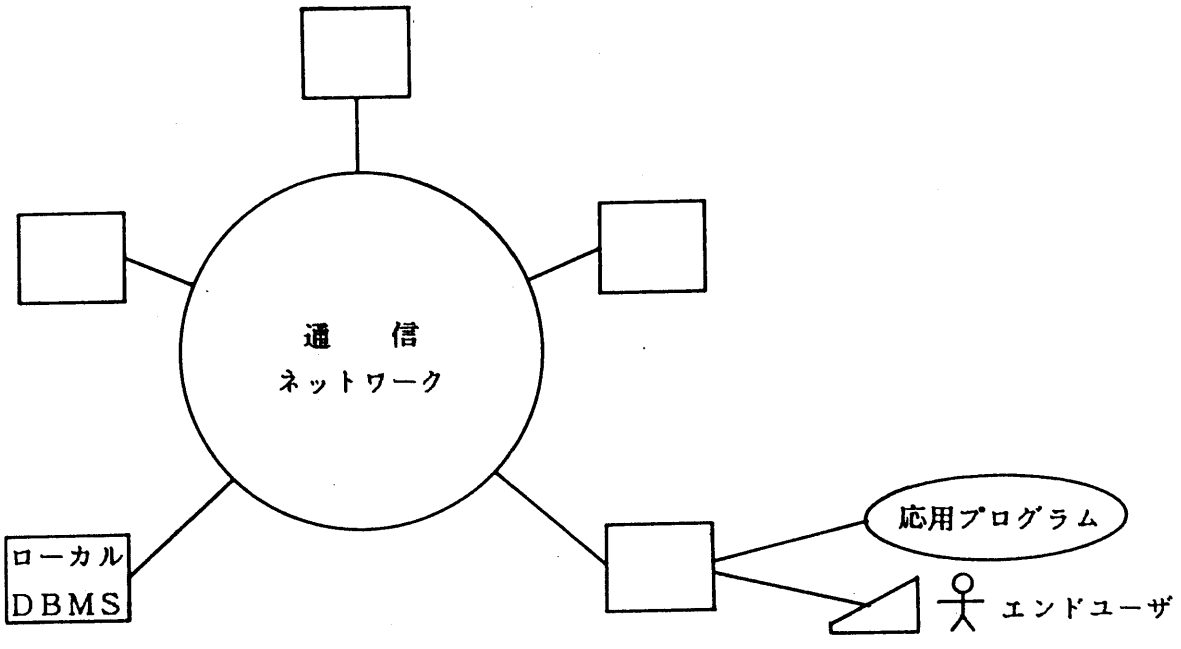


図３．１　分散データベースシステムの概念

分散データベースシステムは、単一の大規模データベースシステムと比べて、いくつかの利点を持っている。ＣＣＡのＪ．Ｒｏｔｈｎｉｅ他は、主な利点として次の３点をあげている[C1)]：

（１）高信頼性：　分散システムの特徴である一部の障害がシステム全体のダウンにつながる可能性が極めて少ないという利点を、分散データベースも又持っている。

（２）通信コストの低減：　各ローカルＤＭＢＳはアクセス要求の発生地点で、もしそれがローカルに処理可能であれば、処理を実行する。従って、各ローカルＤＢＭＳに夫々の地で頻繁に利用されるデータを収容することにすれば、各地域とセンタとを通信回線で結合する集中型データベースシステムに比べて、大巾な通信コストの低減が可能である。

（３）拡張の容易性：　分散データベースでは、集中型データベースと違って、サービスを中断することなく、容量の拡張、機能の追加等が可能となる。

３．３　分散データベースの技術課題

前節で述べたように、分散データベースは、従来の集中型データベースに比し、極めて有望な利点を持ってはいるが、また解決すべき技術課題も数多くある。以下では、これらの課題について、その主なものと、解決策について概説する。

３．３．１　データの一貫性維持と同時実行制御方式

一般に、分散されたデータへの更新操作に関しては次の２種の問題を考察する必要がある[C33)]。１つは、操作の無矛盾性あるいは原子性（atomicness)の保証とでも呼ぶべきものであり、図３．２によって例示されるようなデータの一貫性を保証することである。

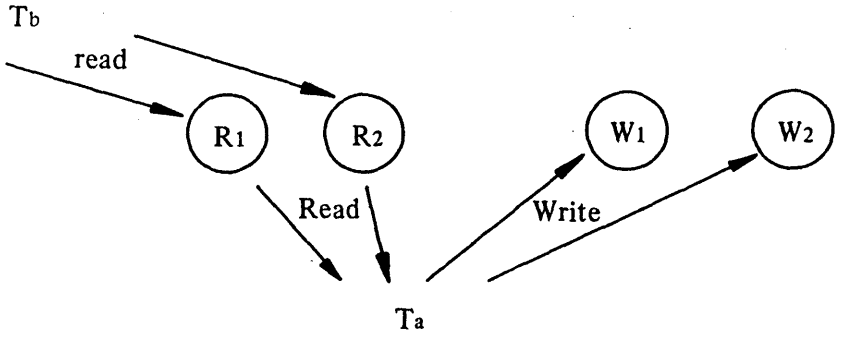


図３．２　データ操作の原子性

たとえば、トランザクションＴａはファイル R1，R2を読み込んで更新データを作成し、それをファイル W1，W2 に書き込むことを実行するものであるとする。このときＴａの読み込み操作は、原子的に行わなければならない。たとえば、Ｔｂを R1，R2 に書き込みを行うトランザクションであるとしたとき、Ｔｂが R1 に書き込みを行い、次に R2 に書き込

みを行う直前で、Ｔａによる R1，R2 の書き込みが行なわれたとすれば、Ｔａの書き込み操作は一貫性を失ってしまう。なぜならば、 R1 にはＴｂの実行結果が反映されているのに、 R2 には反映されておらず、従って、Ｔａが R1，R2 を読み込んで作成した更新データは一貫性を欠くものだからである。

明らかに、このようなデータの一貫性は、更新操作時に、対象データを全く同時に操作するか、あるいはロックをかけるかして、操作を原子的に行うことができれば、保証することができる。尚この問題は、たとえばメディア障害等で W1 が書き込み不能となった場合にも発生する。

他の１つは、複数の操作が同時に与えられたときの操作の順序の保証とでも呼ぶべきもので、図３．３によって例示されるようなデータの一貫性の保証である。

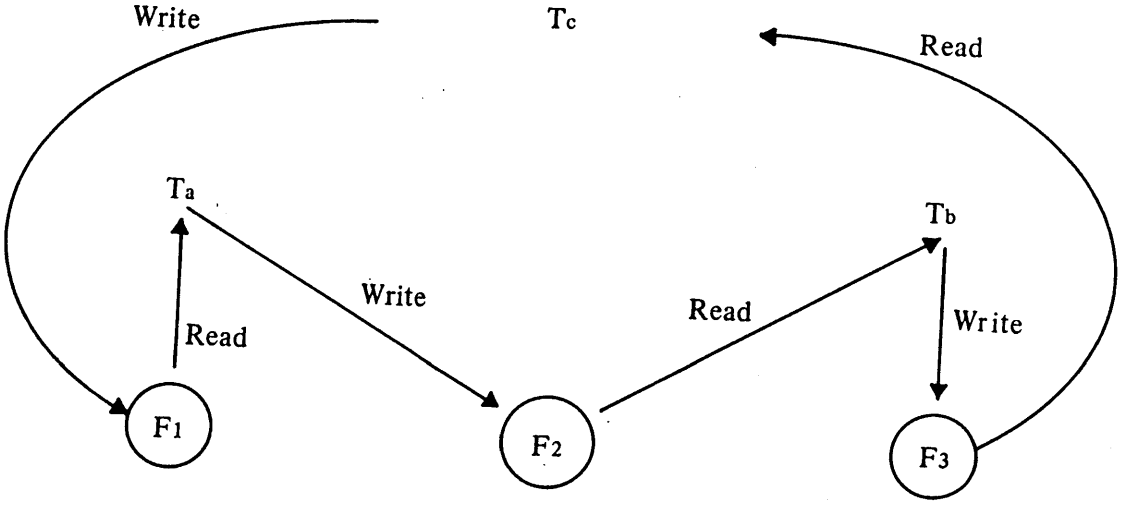


図３．３　トランザクション群の順序づけ

たとえば、Ｔａ，Ｔｂ，Ｔｃを３つのトランザクションとし、ＴａはF1 を読み込んでF2 へ書き込み、ＴｂはF2 を読み込んでF3 へ書き込み、ＴｃはF3 を読み込んでF1 への書き込みを行うものとする。各トランザクションの読み／書きの対象は１つであるから、どのようなトランザクション群の同時実行形態をとろうとも、操作の原子性はこの場合保証される。

しかしながら、たとえば、３つのトランザクションＴａ，Ｔｂ，Ｔｃの読み込みを一斉に行い、その後Ｔａ，Ｔｂ，Ｔｃの書込み操作を一斉に行った場合、ＴａはＴｃによって書込みの行なわれる前のF1 を読みこんでいるため、ＴａはＴｃに先行したと考えられ、同様の理由でＴｂはＴａに先行し、ＴｃはＴｂに先行したと考えられる。従って、Ｔａ，Ｔｂ，Ｔｃの間にどちらが先に実行されたかを決定する順序が定義できなくなってしまう。従ってこのような、トランザクション群の実行形態では、順序づけという意味でのデータの一貫性は失われることになる。

分散データベースの一貫性維持に関する同時実行制御方式は、従来の集中型データベースにおけるそれの発展形として考えることができるということもあって、比較的よく研究された技術課題となっている。現在までに、いくつかの制御方式が提唱されているがC2~4, 16,18,21,23~25,30,33)、これらは大別して、従来型データベースにおけるロックの手法C6~12) を何らかの形で分散データーベースに拡張したロック制御方式と、トランザクションの競合判定に時刻印 (time stamp) を用いるタイムスタンプ方式C2,3,19,20) とがある。一般に前者の方式は、データは分散しているものの制御の方式は集中型となるのに対し、後者の方式は、制御までも分散されるものが多い。前者の代表的なものは、カリフォルニア大学の分散版ＩＮＧＲＥＳ，後者の代表的なものは、ＣＣＡ社のＳＤＤ－１であり、いづれも、前述の２種のデータの一貫性を１つの制御機構によって保証したものとなっている。

なお、ＳＤＤ－１では、この制御機構とは別に、トランザクションのパターンをあらかじめ解析しておき、そのパターンに応じて同時実行のプロトコルを強いものから弱いものまで定めておくことにより、プロトコルのオーバーヘッドを低く押えるための提案もなされているC2,3,19,20)。

３．３．２　ロケーショントランスペアレンシイの実現

前述のように、分散データベースは、物理的に分散したローカルデータベースを論理的には１つのデータベースに統合したものであった。このため、システムには、ユーザが要求するデータの所在サイトを知らなくとも、そのデータへのアクセスを可能とする機構、すなわちロケーショントランスペアレンシイの実現機構を備えなければならない。

この実現方式には大別して２つの方法がある。つまりデータとその所在サイトとの関係を示す情報（ディレクトリと呼ぶ）を持つ方式と、持たない方式である。この方式をさらに細分すると図３．４のようになる。

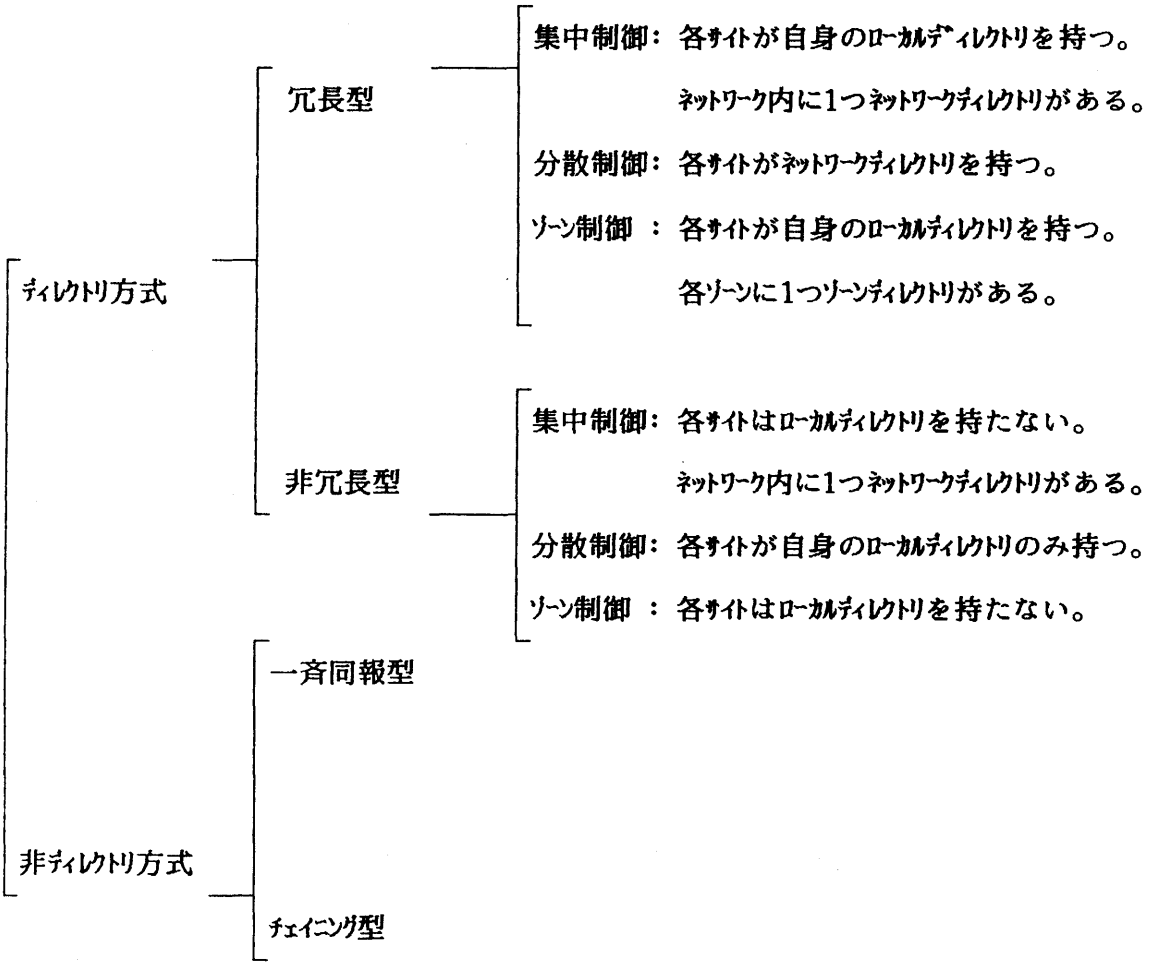


図３．４　ロケーショントランスペアレンシイの実現方式

なおここで、ネットワークディレクトリとは、分散データベースシステム全体に対する全ネットワークワイドなディレクトリであり、ゾーンとは、ネットワークをいくつかの部分ネットワークに分割したものを指す。

分散版ＩＮＧＲＥＳは、基本的には非冗長の分散制御方式をとってあり、オプジョンとして、冗長型の集中制御方式、分散制御方式とを併用している。またＳＤＤ－１は、冗長型の集中および分散制御と非冗長型の分散制御方式とを併用したものとなっている。

３．３．３　障害検出と回復処理C13~15,27)

分散データベースシステムの利点のひとつは、高い信頼性にあった。すなわち小数のコンポネントが障害となっても、全体としてのサービスは継続できるという点である。しかしながらこのような利点を実現させるためには、いかにそれらを全体から分離するか、あるいは、サイトの修復時に、それをどのようにして矛盾なくシステム全体に組み入れていくかといった事項がシステム実現時の技術課題となる。

一般に分散データベースにおける障害には、伝送路の一時的障害やネットワーク上でのプロトコル実行時におけるバッファビジーのような短期障害と、サイトのシステムダウン伝送路の恒久的障害、メディア障害（ディスク等）のような長期障害とが考えられる。以下に示す技術課題のうち、リライアブルブロードカストは短期障害に、その他は長期障害に対する対策となっている。

［１］リライアブルブロードカストの実現C1,3)

図３．５に示すように、更新トランザクションを３つのサイトに同報しようとするとき、あるサイトには要求が届き、あるサイトには届かなかったという状況が発生すると、データの一貫性は失われてしまう。このような状況を回避するための制御方式とプロトコルの実現が課題となり、これはまた前述の同時実行制御方式とも関連する問題となる。

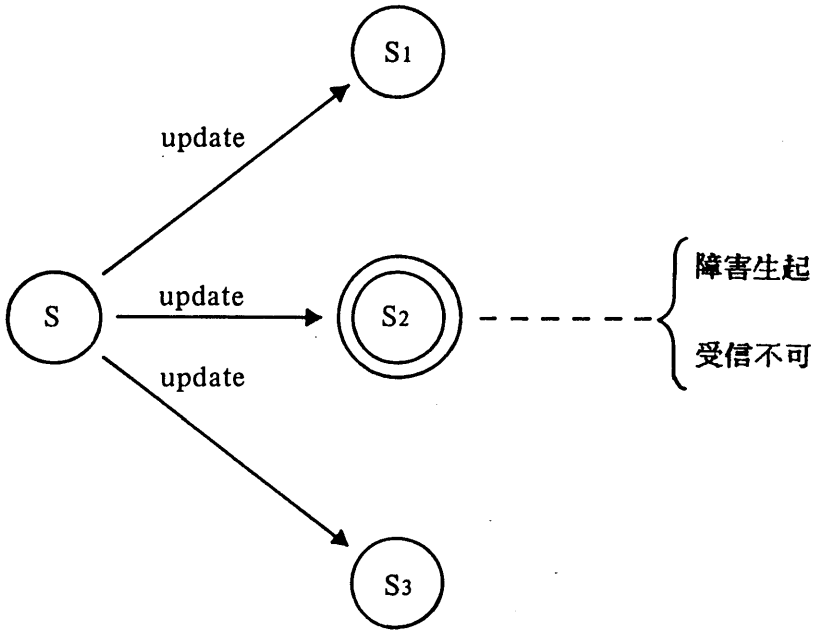


図４．５　更新トランザクションのブロートカスト例

［２］障害サイトの検出方式

障害が発生しているサイトに対して、更新操作の必要が生じた場合、その要求は、障害生起中には処理出来ない。これがデータの一貫性を失う原因となる。このため、障害サイトの早急な検出と、その隔離の実現方式が問題となる。

［３］障害サイトの復旧方式

障害から復旧したサイトは、その障害生起中に要求のあった更新操作は全て未実施となっている。このため、障害復旧サイトを再度システムに組み入れるためには、失われた更新要求を再実行する必要がある。このための再実行の方式がひとつの技術課題となっている。

［４］ネットワーク分割の検出と復旧方式

図３．６に示すように、ネットワークが障害により、N1，N2 の２つの部分ネットワークに分割されてしまうと、N1 は N2 を障害部分とみなして隔離し、また N2 は N1 を同様に障害部分とみなして、各々独立に更新要求を受付けてしまう。このため部分ネットワーク間ではデータの一貫性は失われ、従って分割ネットワークの復旧時に問題を生じることになる。

このような状況を少ない通信量で検出するための機構が技術課題となる。

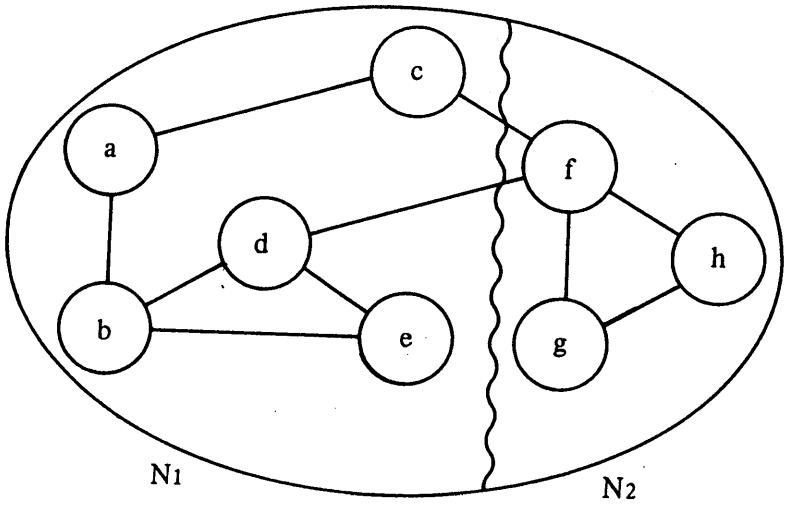


図3．6　ネットワークパーティション

なお、分散版ＩＮＧＲＥＳでは、障害の検出がなされると全サイトへの診断メッセージの同報を行い、障害部分の隔離、パーティションの検出を含むシステムの再構成を行っている。復旧時の処理も同様であり、障害中に送られてきた更新要求は正常サイトにキューづけしておき、回復時にそれを処理した後、診断メッセージの同報により再度システムの再構成を行うC3)。

ＳＤＤ－１における障害処理もほぼ同様であり、spooler というプロセスにより障害中に発生した要求のキューづけを、またリライアブルネットワークサブシステムによってリライアブルブロードカスト機構を実現したものとなっている。なお、パーティションの検出は、ＳＤＤ－１においては、考慮されていないC2)。

３．３．４　分散型問合せ処理の最適化

システムに出されるトランザクションは、いくつかの基本的要求に分解できる。このため、分散処理の特性を生かし、これらの要求を並列に処理することも可能となる。

ユーザから出されたトランザクションをどのような基本的要求に分解し、かつどのサイトで実行するかを遅延時間、通信コスト、等の観点から考察し、問合せ（キュアリ）実行の戦略を決定するアルゴリズムを実現することが問題となる。

なお、この技術課題は、最初、ＳＹＳＴＥＭＲ等の集中型リレーショナルデータベースにおけるキュアリの最適実行方式の研究から始まり、その後Ｗｏｎｇ等により分散キュアリへと拡張され研究が行なわれたC22)。

分散問合せ処理におけるＷｏｎｇの方法は、ＳＤＤ－１，分散版ＩＮＧＲＥＳの双方に影響を与えたものとなっている。

３．３．５　異種データベースの統合化

既存データベースや公共大規模データベースをネットワークにより共有させる際に問題となるのは、各々のデータベースのモデル、操作演算、それらが基づくＯＳ、ハードウェア、セキュリティ機構の違いである。これらのデータベースの真の共有化をはかるためには、エンドユーザに対しては、一様なインタフェースとアクセス法を提供しておく必要がある。

このためのアクセス言語間の変換、共通モデルの確立等が技術課題となる。

３．３．６　ファイル割当て

どのようなデータをどのサイトに配置するかということは、通信コストを減ずる上で極めて重要な意味を持つ。応用に依存してデータにどのような性質があるか、どのデータを重複して配置すべきかといったことが考察の対象となる。

解決法は、主に古典的な最適値問題の適用によっている。つまり目的は、あるファイル割当てにより通信コストを最小とすることであり、与えられるパラメータは：

① 各サイトから発生するキュアリの更新要求と検索要求の比率

② ネットワークのトポロジー、リンク容量、サイト容量

等である。

３．４　本章の位置付け

本サブシステムは、前節で述べたようなローカルネットワークをベースとしているため、分散データベースシステムとしては、極めて際立った種々の特徴を持っている。

以下では、各技術課題に対する本システムでのアプローチを示すことにより、分散データベースとしての本システムの位置付けを行う。

［１］データの一貫性維持と同時実行制御方式

まず操作の原子性は、前章で述べたローカルエリアネットワークＢＡＮＥＴによって保証されている。

従って分散データベースシステムとして考慮すべきは、順序づけという意味でのデータの一貫性の保証である。この技術課題に対しては、本システムでは、ＳＤＤ－１に於て提案されていた、入力されるトランザクションの性質を前もって解析しておくという手法と類似の手法を採用しているC39,43)。但し、ＳＤＤ－１における方法との差異は、本システムに於ては操作の原子性の保証と順序づけの保証という２つの事項を明確に区別している点にあり、その結果、解析の手法が簡単であり、タイムスタンプやリソースに対するロックといった方式を併用しないでも済むという利点を得ている。

［２］ロケーショントランスペアレンシイの実現方式

広域ネットワークをベースとしたシステムに於けるロケーショントランスペアレンシイの実現方式については、様々な評価を行い、与えられたトラフィック量、update 比率等に依存して適切な方式が定まるとの結論を得たC31)。

詳細については文献 C26),C31)を参照されたい。なお本システムにおいては、バスタイプのローカルネットワークを採用したため、基本的には非ディレクトリ方式一斉同法型となったが、障害等に対処するため、各サイトが全ディレクトリを持つ冗長分散型となっている。

［３］障害検出と回復処理

リライアブルブロードカスト機能はローカルネットワークＢＡＮＥＴによって実現されている。

なお、パーティションの検出方法および障害サイトの検出と回復処理については、より少ない通信量でこれを実現するためのアルゴリズムを開発したC28)。但しこのアルゴリズムは、広域ネットワーク上の分散データベースを対象としたものであり、バスタイプのローカルネットワーク上に構築された本システムにおいては採用されていない。

［４］分散型問合せ処理の最適化

ローカルネットワークをベースとする分散型データベースにおいては、通信コストは広域ネットワークにおけるそれ程、クリティカルな要因とはならない。このため、従来から研究されてきた通信量に焦点をあてた最適化をそのまま適用することは難しく、新たに、最も並列実行性を発揮する様な最適化を考慮することが必要となった。

この問題は、またデータベースの設計、データベース上の応用等と関連が深いため、ＳＤ^{3}システムにおいては、分散型推論における各ノードへの割当て知識量という形で対処することとし、分散データベースシステムに閉じた技術課題としては取り上げなかった。

［５］異種データベースの統合化

本システムは全て同種のデータベースモデルで統一することとしたため、この問題は生じなかった。

［６］ファイル割当て

データベースの設計と応用にかかわる問題であり、4.4.4 と同様知識ベース設計時の問題として対処することとした。

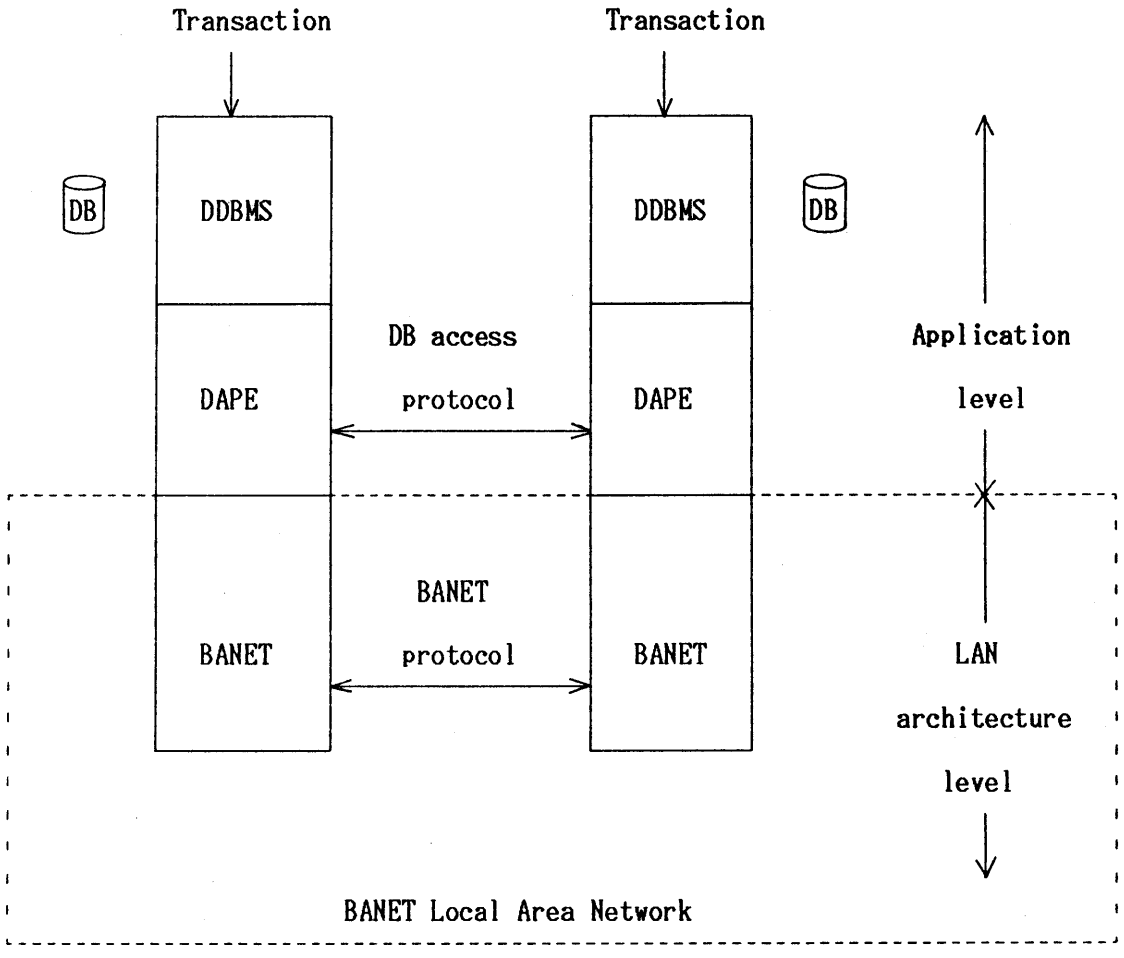
３．５　分散データベースサブシステムに関する基本的考察

３．５．１　データの一貫性維持

第 3.3.1 節で既に述べたように、データベースにおけるデータの一貫性維持のためには、リード／ライト操作のアトミック性とトランザクション実行の順序づけ、という２種の異なる問題を考慮しなくてはならない。本システムにおいては、前者はローカルネットワークＢＡＮＥＴのコミットメント制御機構により実現されている。一方後者は、後述するように、分散データベースサブシステム内のデータベースアクセスプロトコルによって実現されることになる。

３．５．２　分散データベースサブシステムの概要

分散データベースサブシステムの論理構成を図３．７に示す。ここで、ＢＡＮＥＴはブロードカストに基づくＬＡＮプロトコルをサポートし、コミットメント制御を含む分散データベースのアクセス・プロトコルを実行する。



*DDBMS=Distributed Data Base Management System

*DAPE =Database Access Protocol Executer

図３．７　分散データベースサブシステムの論理構成

３．５．３　システムモデルとトランザクションの前解析（プレアナリシス）

［１］システムモデル

トランザクションの実行は、必要なデータを読み、それによって更新データを一時記憶領域に作成するリードフェイズと、その更新データを二次メモリ上に実際に書き込むライトフェイズとからなる。このとき読み込まれるメモリ領域の集合をリードセット、書き込みの行われる領域の集合をライトセットといい、同一のリードセットとライトセットを持つトランザクションの集合を、トランザクションクラス（または単にクラス）と呼ぶ。

クラスａのリードセットとクラスｂのライトセットとが空でない共通部分を持つとき、図３．８(ａ）のような方向付きエッジで表し、これをａのリードエッジと呼ぶ。クラスａとｂのライトセット同士が交差するときは、（ｂ）のような方向を持たないエッジで表し、これをａまたはｂのライトエッジと呼ぶ。また、ａのリードセットとライトセットが交差するときは、（ｃ）のように表し、これをａのセルフループと呼ぶ。

システムが扱う全てのクラスをノードとし、それらの間の交差関係をエッジにより表したものをクラス競合グラフと呼ぶ。

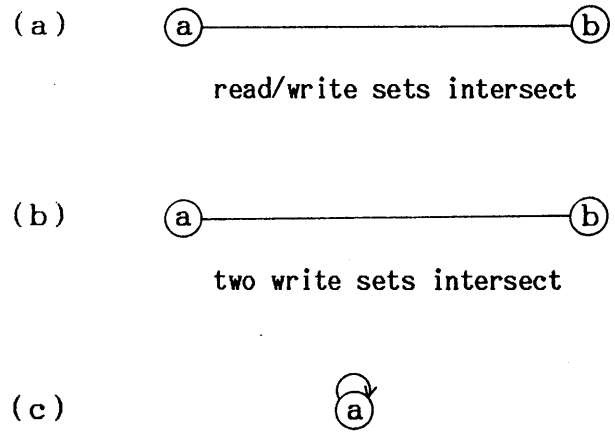


図３．８　クラス間の競合関係

［２］クラスの解析

与えられたクラスａに対し、ａのライトフェーズを　　と記す。以降では、これをひとつのクラスのように扱う。

次に、同時実行中のトランザクションの集合Ｔ＝｛Ta,Tb･･･｝が与えられたとする。任意のクラスａに対し、述語ａ（・）は次のように定義される：

ａ（Ｔ）＝１　ｉｆ　Ｔａ∈　Ｔ　ただしＴａはクラスａに属するトランザクション。

同様に、述語a_w（・）も次で定義される：

a_w（T）＝1　if　Ta ∈ T，かつTaのライトフェーズが実行中。

明らかに、a_w（T）＝1ならばa（T）＝1である（これを　w（・）がa（・）をインプライすると呼ぶ）。もちろん、逆は成り立たない。

次に、複合クラスs＝a，b，c，・・・（つまり、クラスおよびそのライトフェーズの重複を許さない組みあわせ）に対し、その述語S（・）は、クラス述語を用いて次のように定義される。

S（T）＝abc・・・・（T）＝a（T）∧b（T）∧c（T）∧・・・

明らかに、a_w（・）はa（・）をインプライし、abw（・）はab（・）をインプライすることになる。

二つの複合クラスSおよびS'において、述語S（・）がS'（・）をインプライするとき、SがS'をインプライすると呼ぶ。

与えられた複合クラスSに対し、トランザクション集合Tが次の性質を満たすとき、TをSの生起と呼ぶ：

S（T）＝1かつ、Sをインプライする任意の複合クラスS'についてS'（T）＝0が成り立つ。

複合クラスS＝a_1 a_2・・・a_nが与えられたとき、Sのラベル付きグラフは、次の手続きをクラス競合グラフの各a_iにほどこすことによって得られる：

① a_iがクラスであれば、a_iのリードエッジにラベルを付ける。

② a_iがライトフェーズであれば、a_iのすべてのライトエッジに対し、a_iから他ノードへ向かう方向づけをし、ラベルを付ける。

③ クラス競合グラフから、すべてのラベルなしエッジを取り除く。

このようにして、与えられたＳに対するラベル付きグラフを得ることができる。もし、ラベル付きグラフがループを形成した場合、複合クラスＳはループであると呼ぶ。

また、複合クラスＳの生起であるトランザクション集合が、順序づけの一貫性を失うことなしに同時実行可能であるとき、Ｓは同時実行可能であると呼ぶ。

命題１．

複合クラスＳがどのようなループもインプライしなければＳは同時実行可能である。

証明

Ｓは、どのようなループもインプライしないから、Ｓの各要素記号間にラベル付きエッジに従った半順序関係を定義する事ができる。クラス競合グラフの定義から明らかなように、この半順序に従ってトランザクションを逐次実行させたときの結果とＳの同時実行結果とは一致するから、Ｓは順序付き一貫性を失わない。よって同時実行可能である。

ループであってかつ他のループをインプライしない複合クラスのことをベーシックループと呼ぶ。

命題２．

もし、ＳがどのようなベーシックループもインプライしなければＳは同時実行可能である。

証明

ベーシックループの定義から、任意のループはあるベーシックループをインプライする。したがって、Ｓはどのようなベーシックループもインプライしないからどのようなループもインプライしない。ゆえに命題１より、Ｓは同時実行可能である。

こうして、クラスコンフリクトグラフ上のすべてのベーシックループを導くことで、トランザクションクラスの解析は終了する。

なお、後述するように、ベーシックループの生起であるようなトランザクション集合Tの同時実行は、分散データベースアクセスプロトコルの実行により回避させられることになる。

与えられたベーシックループℓに対し、ℓに関係するリード／ライトセットを格納しているようなすべてのＤＢサイトの集合を、ℓのコンフリクトグループと呼ぶ。

なお、ここで、クラスがセルフループを含むとき、そのトランザクションンクラスに属する複数のトランザクションを同時実行してはならないことに注意されたい。

ベーシックループＳ＝$a_1, a_2 \cdots a_n$と、Ｓの生起とが与えられ、クラス ａｉに属するトランザクションがｍｉ個同時実行されているものとする。このとき、ベクトル（m_1, m_2, ・・・m_n）をロック情報（Ｌｓｔ）と呼ぶ。同様に、実行が終了したトランザクション数　のベクトル（m'_1, m'_2, ・・・m'_n）をリリース情報（Ｒｓｔ）と呼ぶ。ベクトル対（Ｌｓｔ，Ｒｓｔ）を実行情報（Ｅｓｔ）と呼ぶ。

Ｌｓｔは、トランザクションの実行開始によって、該当桁に１が加算され、Ｒｓｔはその実行終了に伴なって１加算されるものとする。もし、Ｌｓｔのすべての桁が非ゼロとなったとき、その実行情報はコンフリクティングであると呼ぶ。

以降では簡単のため、コンクリフティングではない実行情報は、最下桁（Right most digit）がＯとなるように巡回シフトしておくものとする（実行情報は巡回シフトしてもその意味は変わらない）。

次に、コンクリフティングでない実行情報　Ｅｓｔ＝（Ｌｓｔ，Ｒｓｔ），が与えられたとき、変換ϕを次のように定義する

但し、Ｌｓｔ＝（b_1, b_2, $\cdots b_n$）。

Ｒｓｔ＝（a_1, a_2, $\cdots a_n$）

$$\phi((Lst, Rst)) = (L'st, R'st)$$

$$a'_1 = a_1 - b_1$$

$$a'_i = \begin{cases} a_i - b_i & if\ a'_i - 1 = 0 \\ a_i & if\ a'_i - 1 \neq 0 \end{cases}$$

$$b'_i = 0$$

$$b_j = \begin{cases} 0 & if\ a_j \neq a'_j \\ b_j & if\ a_j = a'_j \end{cases}$$

この変換ϕは、トランザクションの終了に伴って、実行情報を再設定するのに用いられる。実行終了に伴ってＬｓｔの対応桁を１減ずるだけでは、順序付けの一貫性を維持することができないのは、明らかである。

このようにして、任意のベーシックループℓに対し、ℓのコンフリクトグループのメンバーサイトは、ℓに対する実行情報を持つことになる。

３．５．４　分散データベースアクセスプロトコルと同時実行制御

本節では分散データベースアクプロトコルの概要を述べ、その実行をＢＡＮＥＴがどのようにサポートしているかについて論ずる。

データベースアクセスプロトコルの正常シーケンスおよびそのときのＢＡＮＥＴの動作を図３．９に示す。

（１）　　トランザクションの実行要求

ユーザからクラスａに属するトランザクションＴａの実行要求を受けると、ＤＤＢＭＳのプロセスＰ（ｉ）はクラスａを含むベーシックループαをサーチし、コンフリクトグループのメンバーサイトにリード要求（ＲＥＱ　ＲＤ）を送信するようにＤＡＰＥに依頼する。

（2）　リード要求の受信

ＲＥＱ　ＲＤを受信したＤＤＢＭＳは、Ｌｓｔのａに対応する桁に１を加算し、その結果コンフリクティングとならなければＰＥＲＭＩＴを、そうでなければ　ＮＯＴ　ＰＥＲＭＩＴをＤＡＰＥを通して返送する。ＮＯＴ　ＰＥＲＭＩＴの場合にはＤＡＰＥはＢＡＮＥＴにＣＧＮＡＣＫを送信し、通信グループは形成されない。

（3）　リードフェーズの実行

ＤＡＰＥから　ＲＥＱ　ＰＥＲＭＩＴＴＥＤを受信すると、Ｐ（ｉ）はリードフェーズを実行し、ライトデータを生成する。

（4）　ライトフェーズの要求

更新データを作成したＰ（ｉ）は、ライトフェーズ　ｗを含むベーシックループβをサーチする。さらにＰ（ｉ）は、"ａ"のライトセットを格納しているすべてのサイト（実際にライト操作が実行されなければならないサイト）をサーチする。さらに、Ｐ（ｉ）はコンフリクトグループβ（図３．９のβ）のすべてのメンバーリスト、およびライト操作が実際に行なわれるべきサイトのリスト(図３．９のγ)を指示して、ＤＡＰＥに対してライト要求（ＲＥＱ　ＷＴ）の転送を依頼する。

（5）　ライトフェーズの実行

ＢＡＮＥＴによって通信グループ形成の正常終了（ＣＧＦＯＲＭ success)が通知されると、ＤＡＰＥはコミットメントモードのデータ送信によりライト操作を実行する。

（6）　トランザクションの終了

コミットメントモードのデータ送信が完了すると、ＤＡＰＥはＰ（ｉ）にライト操作が正常終了したことを通知する（ＷＲＩＴＥ　ＣＯＭＰＬＥＴＥ）。Ｐ（ｉ）は全ての関連サイトに対するＣＯＭＰＬＥＴＥメッセージの転送をＤＡＰＥに依頼する。

ＤＡＰＥはＢＡＮＥＴに対してＣＧＤＲＯＰコマンドを発行し、関連するすべての通信グループ（図３．９における３つのグループ、＃α,＃β,＃γ）を消滅させる。

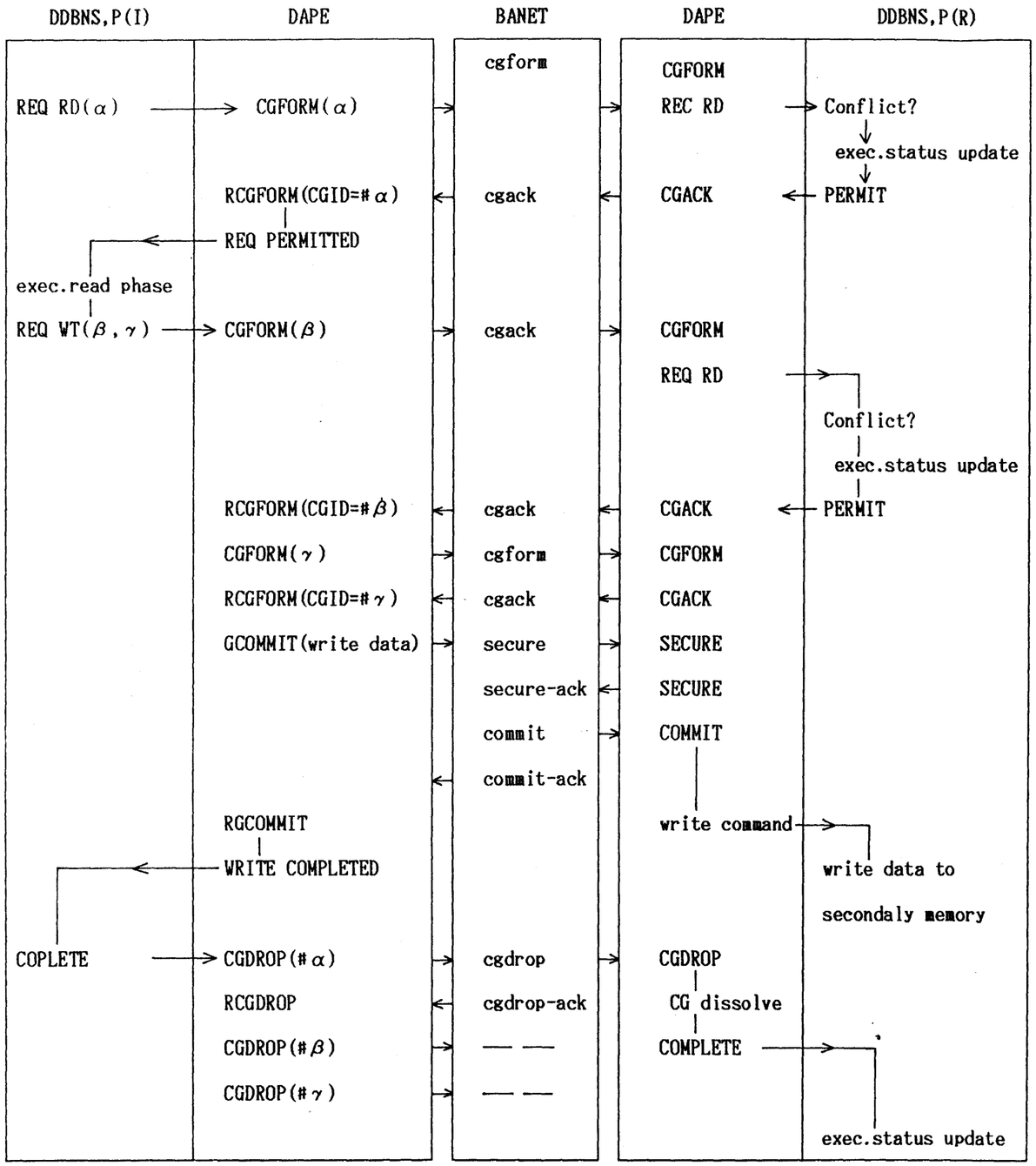


図３．９　分散データベースアクセスプロトコル　概要

なお、ステップ（4）においてコンフリクトが発生した場合、そのライト要求はペンディングとなり、一定のdelayをおいた後再度要求が発せられることとなる。このとき、ともにペンディングとなった2つのライト要求　wと　wとがデッドロックを引き起こすことはない。なぜならば、もし2つのライトアクション　wと　wとがデッドロックを引き起こしているとすれば、これは次の2条件が成り立っていることと等価になる；

①　2つのベーシックループ；

$\iota 1 = x_1\ x_2 \cdots x_K\ a_w$　$(x_s = b \quad 1 \leqq s \leqq k)$

$\iota 2 = y_1\ y_2 \cdots y_K\ b_w$　$(y_\iota = a \quad 1 \leqq t \leqq j)$

が存在する。

②　$\iota 1$，$\iota 2$に対応するロック情報Ｌｓｔ１，Ｌｓｔ２が；

$$x_1 \quad x_2 \cdots x_K \quad a_w \qquad\qquad y_1 \quad y_2 \cdots y_j \quad b_w$$

$Lst = (m_1, m_2 \cdots m_K, 0)$，$Lst = (n_1, n_2 \cdots n_j, 0)$

となっている。ただし、$x_1 \cdots x_K$　，　$y_1 \cdots y_j$　はクラスまたはライトアクションで、$m_1, \cdots m_K$，$n_1 \cdots n_j$ は正の整数である。（図３．１０）。

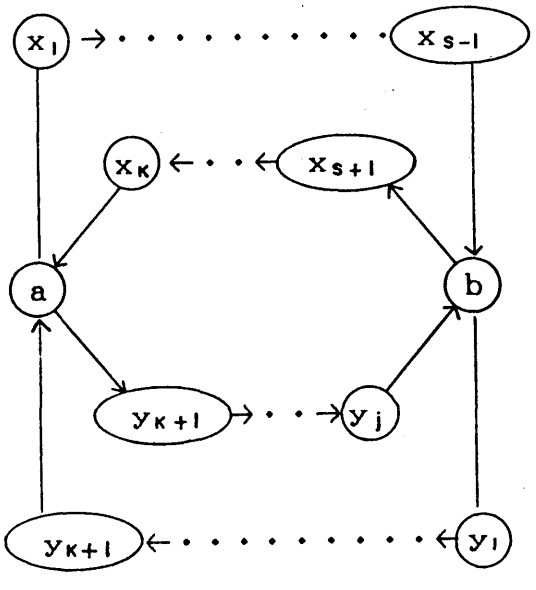


図３．１０　ライト要求コンフリクト時のクラス競合グラフ

この２条件が成り立っているときには必ずあるベーシックループが存在して、その実行ステータスはコンフリティングとなっている。というのは、

$\iota 1, \iota 2$はともにベーシックループだから、記述a $y_{\iota+1} \cdots y_j \quad b_{xs+1} \cdots x_k$ はループである。したがって、これによりインプライさせられるベーシックループ　が存在する。一方条件２より、ιの実行情報はコンフリクティングとなっているからである。

３．５．５　トランザクションの前解析および実行の例

［１］クラス解析の例

次の例は、ある販売会社の注文処理の流れを簡略化したものである。この会社は図３．１１のように、営業、購入、管理／検査の３部門から成っている。

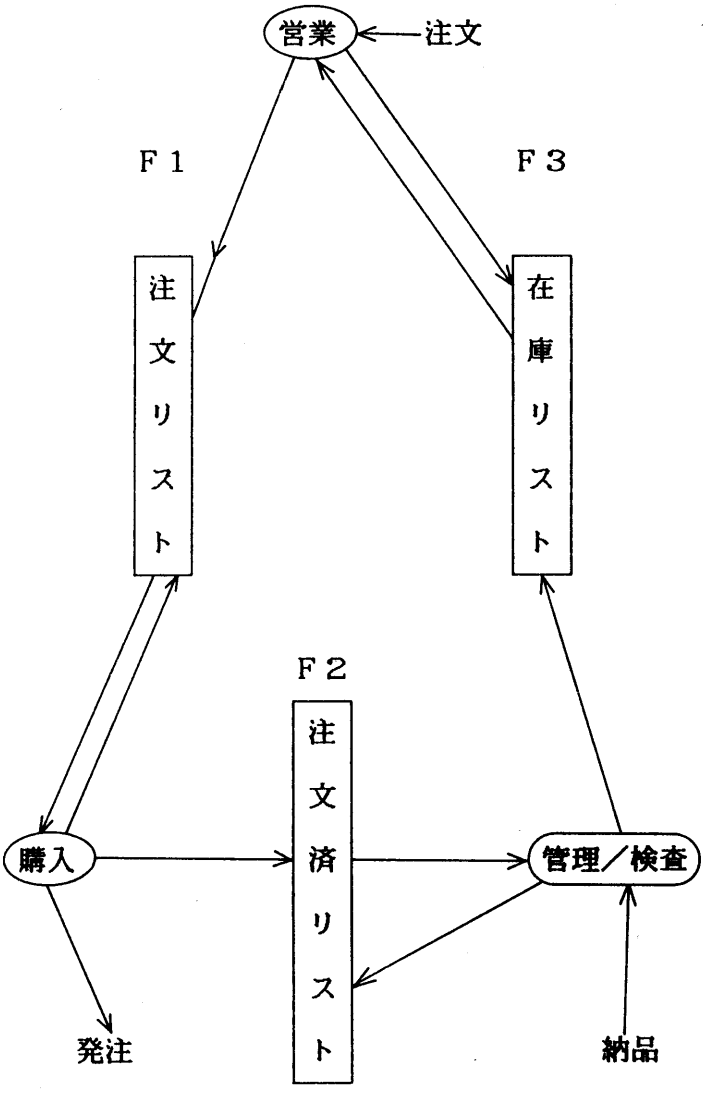


図 ３．１１　注文処理システム

① 販売業務

まず営業部門はユーザからの注文を受付け、注文品目が在庫リストＦ３中にあれば、在庫から取り出し、ユーザに販売する。このとき、Ｆ３の内容から販売した品目は取り除かれる。もし注文品目が在庫リスト中になければ、それをリストＦ１に付け加える。この品目は、後日在庫リストＦ３に追加された後、ユーザに販売されることになる。

② 発注業務

購入部門は、注文リストＦ１に記載されている品目をメーカー側に発注する。発注の済んだ品目については、それを注文済みリストＦ２に付け加え、Ｆ１より除く。

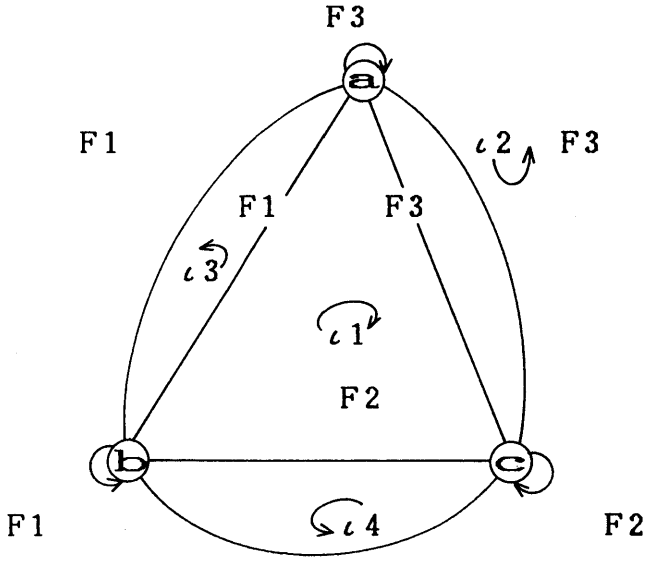
③ 納品、検査業務

管理／検査部門は、メーカー側より納品された品目の検査を行なう。検査合格の品目については、それぞれが注文済みリストＦ２に記載されていることを確認の上、注文済みリストＦ２からその品目を除く。さらにその品目を在庫リストＦ３に追加する。

この３つの業務をそれぞれクラスａ，ｂ，ｃとしたとき、そのリードセット、ライトセットは、表３．１のようになる（このとき品目の追加はライトのみの操作となることに注意）。

表３．１　各業務のリード／ライトセット

トランザクションクラス名	リードセット	ライトセット	動作
a（販売業務）	F3	F3	F3から在庫品目を除き新しいF3とする（リードおよびライト）。
		F1	注文品目をF1に追加（ライトのみ）。
b（発注業務）	F1	F1	F1から発注品目を除き新しいF1とする（リードおよびライト）。
		F2	発注品目をF2に追加（ライトのみ）。
c（納品検査業務）	F2	F2	F2から検査合格品目を除き新しいF2とする（リードおよびライト）。
		F3	合格品目をF3に追加（ライトのみ）。



F2

図３．１２　クラス競合グラフ

表３．２　ベーシックループとそのコンフリクトグループ

ベーシックループ名	複合クラス	コンフリクトグループ
ι 1	a c b	S1, S2, S3
ι 2	$a\ c_w$	S3
ι 3	$b\ a_w$	S1
ι 4	$c\ b_w$	S2

したがって、この例におけるクラス競合は図３．１２のようになり、また各ベーシックループとそのコンフリクトグループとの対応は表３．２で与えられる（ただし各Ｆｉはサイトｓｉが保持するものとする）。

この例では、たとえばループｂ　ｗが実行されると次のような矛盾がおこることがわかる：Ｆ１の内容がＡ，Ｂ，Ｃ，Ｆ２の内容が空、Ｆ３の内容がＸ，Ｙであったとする。さらに営業部門がユーザからＸ，Ｄなる品目の注文を受けたと仮定する。もし発注業務Ｔｂが Ｆ１を読むことと、販売業務Ｔａによる品目ＤのＦ１への追加が同時に行なわれると、次にＴｂのライトアクション実行後ではＦ１は空になってしまい、品目Ｄの発注という業務が消失してしまう。

［２］トランザクション実行の例

この例に基づき、トランザクションの実行例を以下に示す。

例における営業、購入、検査の各部門はそれぞれリストＦ１，Ｆ２，Ｆ３を保持する分散データベースのサイトを構成していると仮定し、各々をＳ１，Ｓ２，Ｓ３で表わす。次に、営業部門Ｓ１がユーザより注文を受け、販売業務を行おうとしているとする。

① S1は自サイト内に販売業務処理のためのプロセスPを発生させる。このときの各実行情報を表中の t_0 で表す。

② PはREQ RDa（ι1，ι2)をS1，S2，S3に送る（このときPとS1とは自サイト内通信、他は回線を介した通信となる）。

③ REQ RDa（ι1，ι2)を受信したサイトはPERMITを応答する（実行情報はt1となる）。

④ S1，S2，S3からPERMITを受信したPはリードフェーズ（F3の読みこみ）を実行し、更新データ(F1への追加内容)を作り出す。

⑤ 次に、Pがライトフェーズを要求する直前に、購入部門がS2内に発注業務処理のためのプロセスP'を発生させ、P'はS1，S2，S3にREQ RDb（ι1，ι3)を送信したとする。

⑥ REQ RDb（ι1，ι3)を受信した各サイトはPERMITをP'に返す。（t2に示す)。

⑦ 更新データを作成したPは、Taのライトフェーズ実行要求REQ WTa（ι3，F1）をS1に送信する。

⑧ REQ WTa（ι3，F1）を受信したS1はベーシックループ ba_W をコンフリクティングとしてしまうためNOT PERITをPに返送する（t_2'に示す。）

⑨ NOT PERITを受け取ったPはABORT WTa(ι3，F1）をS1に送り、実行ステータスはt2に戻る。

⑩ 一方、S1，S2，S3よりACKを受信したP'は、リードアクションを実行し、更新データ作成後REQ WTb（ι4，F2）をS2に送る。

⑪ S2はPERITを送る。(t3に示す)。

⑫ P'はF1，F2に対するライトアクションを実行する。

⑬ P'は更新終了後COMPLETE bをS1, S2, S3, に送る。

COMPLETE b受信後の実行ステータスはt4となる。この後変換ϕにより実行ステータスはt5のようになる。

⑭ ⑨でペンディングとなっていたa は受付可能となり、Pによる

REQ WTa (ι3, F1)再送信へと続く。

表3．3 実行情報の遷移

コンフリクティングループ	ベーシックループ		時点 t	t	t	t'	t	t	t
S1			a c d	a c b	b a c	b a c	b a c	b a c	b a c
S2	acb	Lst	0 0 0	1 0 0	1 1 0	1 1 0	1 1 0	1 1 0	0 1 0
S3	(ι1)	Rst	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
			a c_w	a c_w	a c_w	a c_w	a c_w	a c_w	a c_w
S3	ac_w	Lst	0 0	1 0	1 0	1 0	1 0	1 0	1 0
	(ι2)	Rst	0 0	0 0	0 0	0 0	0 0	0 0	0 0
			b a_w	b a_w	b a_w	b a_w	b a_w	b a_w	b a_w
S1	ba_w	Lst	0 0	0 0	1 0	1 1	1 0	1 0	0 0
	(ι3)	Rst	0 0	0 0	0 0	0 0	0 0	1 0	0 0
			c b_w	c b_w	c b_w	c b_w	b_w c	b_w c	b_w c
S2	cb_w	Lst	0 0	0 0	0 0	0 0	1 0	1 0	0 0
	(ι4)	Rst	0 0	0 0	0 0	0 0	0 0	1 0	0 0

３．５．６評価

プレアリナリシスによるトランザクション実行制御が、従来のリソースをロックする制御方式と比較してどのくらい効率がよいかを簡単な例について定量的に比較する。なお、ここでは図３．１３で示されたようなクラス競合グラフを持つ場合について、任意の時点でトランザクション実行要求が発生したとき、コンフリクトのためにその実行が阻止される確率、すなわち呼損率を評価指標として選択する。

［１］評価モデル

評価の対象とするのは、図４．１３で示されるクラス競合グラフを持つシステムとする。すなわち、

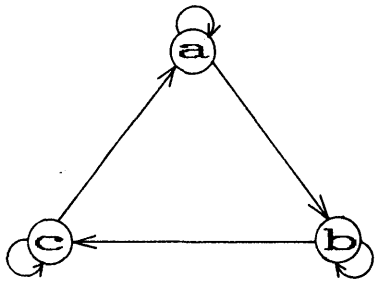


図３．１３　評価対象とするクラス競合グラフ

すなわち、

① ３つのトランザクションクラスａ，ｂ，ｃがある。

② それぞれのクラスはセルフループを持つ。従って同一クラスに属するトランザクションはひとつずつしか実行できない。

③ ａのライトセットがｃのリードセットと、ｂのライトセットがａのリードセットと、ｃのライトセットがｂのリードセットと交差する。

さらに、４ 、５ を仮定する。

④ 各トランザクションクラスにおけるトランザクション実行要求の発生分布は同一であり、平均λのポアソン分布に従う。

⑤ 各トランザクションクラスにおけるひとつのトランザクションの処理に要する時間、すなわちサービス時間の分布は同一であり、すべて平均μの指数分布に従う。

［２］リソースロック方式の呼損率

リソースをロックする方式では、このようなクラス競合グラフが抽かれる場合ひとつでもトランザクションが発生して実行を開始すると、他のトランザクションは一切実行できなくなってしまう。従ってこの場合には、３つのトランザクションクラスのトランザクションの発生はすべて同一窓口に対するものとみなし、入力が平均３λのポアソン分布、サービス時間が平均μの指数分布の場合のM／M／１モデルE3) と考えることができる。

従って、$\lambda / \mu = \rho$ とおき、リソースロック方式の呼損率を$F_{\iota}(\rho)$とすると、

$$F_{\iota}(\rho) = 3\rho / 1 + 3\rho \qquad \text{eq(1)}$$

となる。

［３］プレアナシス方式の呼損率

プレアナシスによるトランザクション実行制御の方式は図３．１４で示すS0 ～S9 の10ケ状態をもつ Continuous-Time Markov Chain としてモデル化できるE5)。図３．１４で示された各状態の上段はロックステータス（Ｌｓｔ）を、下段はリソースステータス（Ｒｓｔ）をあらわすが、便宜的に左から順にａ，ｂ，ｃの状態を固定的に表現するものとし、最下桁が０になるようなシフトはここでは示していない。

Continuous-Time Markov Chain の理論に従い、以下の記号を定義する。

：定常状態における任意の時点でシステムの状態が　　である確率

：今システムが　　にあるとして、ｔ時間後に　　となる確率

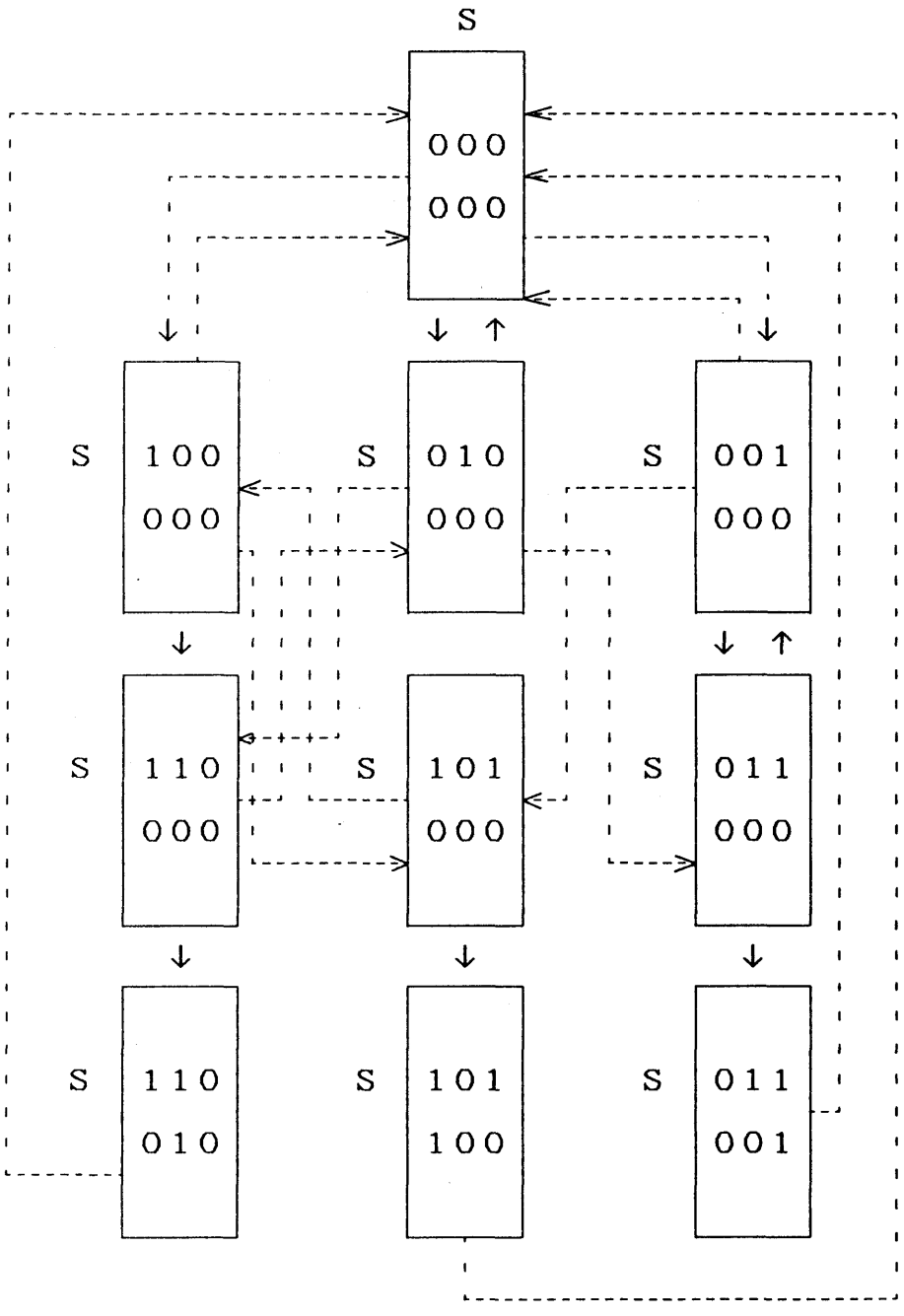


図３．１４　システムの状態とその遷移

q_{ij}：transition rate，すなわち、

$$q_{ij} = \lim_{\Delta t \to 0} P_{ij}(\Delta t) - 1 \Big/ \Delta t \qquad \text{eq(2)}$$

$$q_{ij} = \lim_{\Delta t \to 0} P_{ij}(\Delta t) \Big/ \Delta t \qquad i \neq j \qquad \text{eq(3)}$$

$\pi i, q_{ij}$を要素とする行列を

$$\pi = [\pi i]$$

$$Q = [q_{ij}]$$

とすると、

$$\pi Q = 0 \qquad \text{eq(4)}$$

が成り立つ。

従って、各 q_{ij} を求めればよいことになる。各 q_{ij} は例えば以下のようにして求められる。

$$q_{oo} = \lim_{\Delta t \to 0} P_{oo}(\Delta t) - 1 \Big/ \Delta t$$

$P_{oo}(\Delta t)$は今システムが So（アイドル状態）にあるとしたとき、Δt時間後もSoのままである確率であるから、いいかえれば、Δtの間にａもｂもｃも発生しない確率である。従って、

$$P_{oo}(\Delta t) = EXP(-\lambda \Delta t) \cdot EXP(-\lambda \Delta t) \cdot EXP(-\lambda \Delta t)$$

であり、

$$q_{oo} = -3\lambda \qquad \text{eq(5)}$$

となる。

同様に、例えば$P_{22}(\Delta t)$ は Δtの間にｂの処理が終了せず、かつａもｃも到着しない確率であるから、

$$P_{22}(\Delta t) = EXP(-\mu \Delta t) \cdot EXP(-\lambda \Delta t) \cdot EXP(-\lambda \Delta t)$$

であり、

$$q_{22} = -(2\lambda+\mu) \qquad \text{eq(6)}$$

となる。

同様にして、すべての q_{ij} が求められ、Qは以下のようになる。

$$Q = \begin{pmatrix}
-3\lambda & \lambda & \lambda & \lambda & 0 & 0 & 0 & 0 & 0 & 0 \\
\mu & -(2\lambda+\mu) & 0 & 0 & \lambda & \lambda & 0 & 0 & 0 & 0 \\
\mu & 0 & -(2\lambda+\mu) & 0 & \lambda & 0 & \lambda & 0 & 0 & 0 \\
\mu & 0 & 0 & -(2\lambda+\mu) & 0 & \lambda & \lambda & 0 & 0 & 0 \\
0 & 0 & \mu & 0 & -2\mu & 0 & 0 & \mu & 0 & 0 \\
0 & \mu & 0 & 0 & 0 & -2\mu & 0 & 0 & \mu & 0 \\
0 & 0 & 0 & \mu & 0 & 0 & -2\mu & 0 & 0 & \mu \\
\mu & 0 & 0 & 0 & 0 & 0 & 0 & -\mu & 0 & 0 \\
\mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu & 0 \\
\mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu
\end{pmatrix}$$

eq(4) より、以下の式を得る。

$$-3\lambda\pi0 + \mu(\pi1+\pi2+\pi3+\pi7+\pi8+\pi9) = 0$$

$$\lambda\pi0 - (2\lambda+\mu)\pi1 + \mu\pi5 = 0$$

$$\lambda\pi0 - (2\lambda+\mu)\pi2 + \mu\pi4 = 0$$

$$\lambda\pi0 - (2\lambda+\mu)\pi3 + \mu\pi6 = 0$$

$$\lambda(\pi1+\pi2) - 2\mu\pi4 = 0$$

$$\lambda(\pi1+\pi3) - 2\mu\pi5 = 0$$

$$\lambda(\pi1+\pi3) - 2\mu\pi6 = 0$$

$$\mu\pi4 - \mu\pi7 = 0$$

$$\mu\pi5 - \mu\pi8 = 0$$

$$\mu\pi6 - \mu\pi9 = 0$$

これらの式と

$$\sum_{i=0}^{9} \pi_i = 1 \qquad \text{eq(7)}$$

とから、

$$\pi_0 = 1+\rho \,/\, 1+4\rho+6\rho^2$$

$$\pi_1 = \pi_2 = \pi_3 = 1+\rho \,/\, 1+4\rho+6\rho^2$$

$$\pi_i = \rho \,/\, 1+4\rho+6\rho^2 \qquad i=4\sim9$$

を得る。

クラスａのトランザクションが呼損となるのは、状態がＳ1 およびＳ4~Ｓ9のときであるから

$$\pi_1 + \sum_{i=4}^{9} \pi_i = \rho+6\rho^2 \,/\, 1+4\rho+6\rho^2$$

となる。

クラスｂ，ｃについても同様であるから、結局システム全体の呼損率を Fp(ρ)とすると、

$$Fp(\rho) = \rho+6\rho^2 \,/\, 1+4\rho+6\rho^2 \qquad \text{eq(8)}$$

となる。

リソースロック方式の場合の呼損率 $F_L(\rho)$と、プレアナリシス方式の場合の呼損率 $F_L(\rho)$とを比較したグラフを図３．１５に示す。

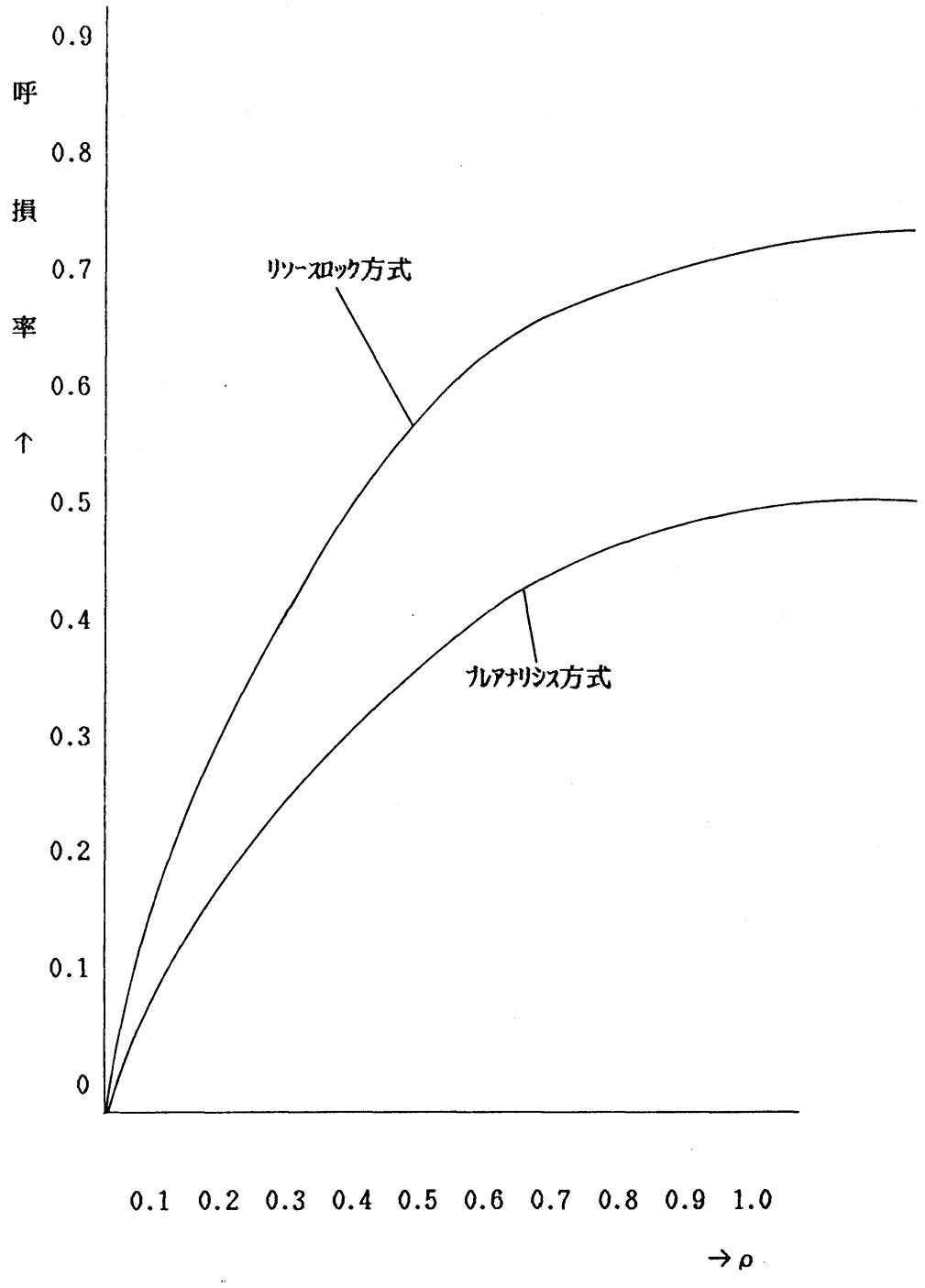


図３．１５　呼損率の比較

第　4　章

演繹データベースシステムとその分散化

本章では、SD^3システムにおける分散型推論サブシステムの技術的背景としての知識情報処理を、その歴史、近年の動向等について論ずる。次に最近注目されている知識ベースシステムや演繹データベースシステムについて概説し、これらの技術と分散処理技術との融合の上で成立する分散型問題解決システムについて、米国での研究事例との対比の上で本研究の位置づけを試みる。

最後に本分散型推論サブシステムの基本アーキテクチャについて考察を行う。

４．１　背景

人間の知的な精神活動である認識と理解、思考、学習および問題解決といったプロセスを数学的に解明し、工学的に実現しようとする研究は、早くからコンピュータ科学に携わる研究者の興味の的となり、また過去30年間、実現に向けての努力が為されてきた。この分野の研究は、今日、「人工知能研究（Artificial Intelligence research)」と呼ばれている。

1950年代までの人工知能研究に於る主テーマは、ゲームや定理証明、翻訳等であった D6,9)。これらは当時のハードウェアの実現レベルから考えると、システムの実現可能性は極めて薄いものであり、実用的観点からも余り興味のあるものではなかった。一時は挫折しかけた人工知能研究ではあったが、その後1960年になって、数値だけでなく記号をも容易に取り扱うことのできる言語ＬＩＳＰの提案がＭｃＣａｒｔｈｙによってなされ、その後の人工知能研究に大きな貢献と発展とを与えることとなった。

この後、1970年代には、人工知能の研究も活発となり、数学を理解して解くプログラムや、積木の世界を対象として会話を理解するシステム、自然言語理解システム、物体認識や部品の組立を行う知能ロボット等に対する研究が盛んになったD6)。

このような歴史を背景として、近年になり、人工知能研究の実用面での展開を重視した知識工学という研究分野が誕生し、各方面の大きな期待と注目を集めるようになった。知識工学の応用として代表的なものが、学習と経験を通して得られた専門家の知識をコンピュータにインストールし、その知識を用いて専門家と同程度の知的問題解決を目指したいわゆるエキスパートシステムであるD6,8,10)。このようなシステムの原点は、スタンフォード大学で開発された有機化合物の分子構造を推定するプログラムＤＥＮＤＲＡＬにみることができる。歴史的には、知識工学（Knowledge Engineering）という言葉は、このＤＥＮＤＲＡＬの開発者であったＤｒ．Ｆｅｉｇｅｎｂａｕｍによって、提唱されたものであるD7)。

その後、現在に到るまで、医療診断（eg, MYCIN, PUFF, etc.）機器システムの故障診断、政策決定支援、地下資源探索、設計支援等の特定問題向きの各種エキスパートシステムが開発され、そのうちいくつかは実用に供されるまでに到っている。

しかしながら、エキスパートシステムの多くは、その問題領域特有の諸要素に、推論の形式や知識の表現形式が依存した形態となってしまっているため、最近では、問題領域特有の要素をシステムから除いた、いわば汎用の知識ベースシステムを考え、この知識ベースシステムを特定問題領域に適用することによってエキスパートシステムを実現させるというアプローチが有力視されている。

知識ベースシステムは、こうした一般的な人工知能研究、および知識工学の適用、さらにはエキスパートシステムの実現手段という背景のもとに提示された新しい形の情報処理システムで、知識ベースと呼ばれる知識を貯えた一種のデータベースと、その知識を利用して問題解決をはかる推論機構と呼ばれる情報処理メカニズムとからなりたっている。

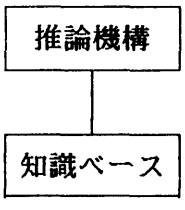


図４．１　知識ベースシステム

４．２　知識ベースシステム

知識ベースシステムは、大須賀によればD19)「一定の形式のもとで構造化された知識と呼ばれる情報を貯える知識ベースと、それを問題解決に利用するための処理装置　―― 推論機構 ――　を基本要素として含むという構成上の特徴を持つシステムである」と定義される。

ここで重要なことは、システムの汎用化のため、知識ベースと推論機構とを論理的に分離し、様々な問題領域に対しても知識を入れ換えるだけで対処できるよう試みていることである。これは、従来のエキスパートシステムにおいては、その問題解決推論の機構までもが特定の問題領域に依存していたため、真の汎用性を欠くという指摘がなされたためである。推論機構と知識ベースの分離を行ったエキスパートシステムの代表的な例は、医療診断システムＭＹＣＩＮから特定問題領域依存の医学的知識と推論のメカニズムを切離したＥＭＹＣＩＮである。

知識ベースシステムにおける知識とは、一般に推論機構の処理対象となる情報を意味する。この知識は、一般規則（ルールと呼ばれる）と個別知識（ファクトと呼ばれる）という２種の異なった知識に分類することができる。ファクト集合とは特定対象とそれに対する属性、性質、対象間の関係等を表したもので、たとえば、"金、銀、銅、鉄、・・・"といった金属とその比重との関係を記述したものはファクト集合であるが「比重が１より大なる金属は、水に沈む」という法則は一般規則である。

通常、コンピュータで行われる推論は一般規則からファクトを導くもので演繹的推論(deduction)と呼ばれている。より正確には、「Ａ(x) ならば Ｂ(x)」という大前提とＡ(a) という小前提（ファクト）とが与えられているとき、Ｂ(a) という結論（ファクト）を導くという形の推論である。

たとえば、前述の例においては、「xが比重１より大なる金属であればｘは水に沈む」というのが大前提であり、「鉄の比重は１より大」という小前提から、「鉄は水に沈む」という結論を導く形態である。

これに対し、帰納的推論（Induction)と呼ばれる推論の形式もある。これはファクト集合から一般規則（つまり大前提）を導くものである。

さらに abduction と呼ばれる推論は、大前提と結論とから小前提を導くもので、仮説生成とも呼ばれる。

後二者の形式の推論を機械的に実行させるための研究も活発になされているがD12,16,17)、現時点では技術上の難点も多い。したがって、本論文では、一般規則とファクト集合とを併せた情報の上で演繹推論および検索の機構が働くような知識ベースシステムに焦点を絞り、以降の考察を行うこととする。

このような知識ベースシステムが現在直面している１つの技術課題は、知識の大規模化にどのように対処するかということである。一般に、実用上効果のあるエキスパートシステムの実現を目差せば、そのルールやファクトが膨大な量となり、これらの物理的収容媒体である、ＲＡＭや磁気ディスク装置への効率の良いアクセス方式の確立なくしては、そのようなシステムの実現は有り得ないことになる。

効率の良いメモリアクセスを実現するため、現在２つのアプローチが考えられている。ひとつは、高速検索アルゴリズムの開発、ハードウェアサポートによる高速ハッシング機構やパターンマッチング機構、連想メモリ、並列サーチ機構の開発といった、アルゴリズムやアーキテクチャの改良によるアプローチである。これに対し、第２のアプローチは、対象となるファクト集合や一般規則をデータベースの手法を用いて管理させ、その上部構造として知識ベースシステムを構築するという方法である。

この第２のアプローチは、既存知識の流用という観点からも重要であり、もしこの方式でシステムが実現されれば、既存のデータベースに蓄積された情報を知識として活用し、問題解決等に役立てることも可能となる。

４．３　演繹データベースシステム

前述のように、大規模知識への対処、および既存知識の流用という目的のため、知識ベースとして汎用のデータベースを用い、演繹推論の機構を組合せることにより、一種の知識ベースシステムあるいは高度データベースシステムを実現しようとする試みが最近注目されている。このようなシステムは、演繹データベース又は論理データベースと呼ばれているD1,2,22,23)。演繹データベースに於るデータベースモデルとしては、その数学的基礎が明確であるという理由により、関係モデルを、推論機構としては述語論理に基づくものを用いるのが、現在の主要な動向D3,13) となっている。

この分野の研究が活発となったのは、1978年のＣＥＲＴワークショップにおいてＧａｌｌａｉｒｅ，Ｍｉｎｋｅｒ等が形式論理E4,6）とデータベース特に関係データベースとの密接な関連を、理論、応用、実動化といった観点から論じて以来のことであるD1,2,13)。

そのなかでＮｉｃｏｌａｓおよびＧａｌｌａｉｒｅは、一階述語によってデータベースを形式化する場合のアプローチを、そのルールとファクトの扱いの差異によって２種に類別し、さらに第３のアプローチというべき方式の提唱を行っている。

４．３．１　演繹データベースへの３つのアプローチ[D2)]

本節では、演繹データベースシステムの理論的基礎を支える上で、又システムの実現方式を検討する上で極めて重要と思われるNicolasおよびGallaireによる３つのアプローチに関する議論の簡単な紹介を行っておく（本節の議論は文献D2による）。

［１］証明理論的アプローチ

このアプローチは、主として質問応答システムで用いられてきたものである。このアプローチでは、ファクトおよびルールはともに一階述語の公理とみなし、それらによって組み立てられる公理体系をＴと定義する。こうすれば、導出によって得られる情報はすべてＴの定理となる。また現実の世界はその公理体系Ｔの解釈（インタプリテーション：Ｔ内のすべての変数および述語に対し具体的対応づけを行ったもので、真偽値を論ずることができる）となるが、それがモデル（公理体系を真とする解釈）となっているか否かは一般に検証できないものとなる。但し、公理体系Ｔが無矛盾か否かのチェックは理論的に可能であり、もし矛盾していなければＴはモデルをもたないことがわかる。

このアプローチによりデータベースを形式化する際特に重要なのは、演繹推論のプロセスと検索のプロセスとをインプリメント上は分離する必要があるということである。なぜなら、通常、データベースは膨大な量のファクト集合を収容しているため、これらを含めて標準的な一階述語における証明法を適用したのでは極めて効率が悪くなってしまうからである。

またこのアプローチにおいては、否定情報を、インプリシットにせよエクスプリシットにせよ何らかの形で表現しておかなければならない。しかしながら多くのデータベースにおいては、明確に否定情報として定義されない（つまり unknown 値を持つ）ものがある、否定情報として定義されたとしても、データ量が余りに多く、それをエクスプリシットに表現するのは困難、等の理由により、通常はこのような表現がなされていない。このため、このアプローチにおいて、データベースを形式化する際には、unknown 値をfalse 値とみなすということが行なわれることになる。

一方、データベースには、通常、インテグリティ制約と呼ばれる一種のルールがある。これはルールではあるが、データベースの無矛盾性のチェックに使われ、情報の導出に使われるものではないため、このアプローチにおいては、このようなルールをうまく扱うことができない。

また、このアプローチによって問合せ(キュアリ)を評価する方法は以下のようになる:

① yes/no の答えを求めるキュアリ（閉キュアリ）Wが与えられたとする。システムはWの導出を試みることになる。もしWが公理系Tの定理であれば答えは"yes"となる。そうでないとすれば、システムは Wの導出を試みることになる。もし Wが定理であれば、答えは"no"である。そうでなければ答えは"don't know"である。

② 自由変数Xを含むようなキュアリ（閉キュアリ），F(x)が与えられたとする。これは、答えとして値を要求するもので例えば、"部品Pの子部品は何か？"とか"部品Pの特徴と色は何か？"といった問合せである。これはF(x)を真とするようなすべての証明を見つけることと等価となる。

［2］モデル論的アプローチ

このアプローチでは、システムは、ルールの集合と、ファクト集合のみを収納したデータベースとから成り、ルール集合によって公理体系Tが、データベースによりTの解釈が与えられる。

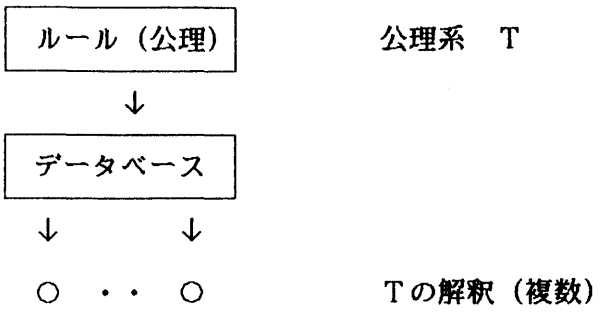


図４．２　モデル論的アプローチ

このときデータベースは明確な１つの解釈に対応する訳ではなく、unknown 値を特定の値でおきかえることによって得られる複数の解釈に対応するものとなる。またデータベース上での追加、更新等の操作は、解釈の変更を意味し、新しくなった解釈も公理系Tのモデルとなるためには、このような操作がルールの真偽値をかえないものであることが必要となる。

一方、ルールの集合は、データベースが実際に公理系Tのモデルを決定できるか否かを検討するインテグリティルールとして用いられる。したがって、もしすべてのルールがデータベース上で真値を持つならば、そのデータベースは公理系Tのモデルを決めていることになる。

このアプローチの１つの欠点は、ルールを情報の導出のために用いていないから、冗長な情報が存在してしまうことである。

またキュアリは、証明すべき定理としてではなく、真偽値表（つまりデータベース）を検索することによって評価されることになる。

しかしながらここで、インテグリティルールではあっても、それをキュアリの評価に用いても有効なルールになるものも存在する。

たとえば次の２つのリレーションを考える。

MN (Company, product)		RP (Agent, Product)	
C11	P3	A6	P3
C11	P5	A6	P2
C13	P2	A6	P5
C11	P6	A6	P6
		A7	P2
		A8	P6

このとき、キュアリ"Agent A6 は C11が出しているすべての製品を扱っているか"（x MN (C11X) → RP (A6,X)）- に対する答えは"yes" となる。ところが、このアプローチにおいては、データベース上で真となるキュアリは、必ずしも定理ではない。つまりその解釈においてのみ正しい。このため、たとえ"yes"という答えが得られたとしても、本当はそれがファクト集合から帰納的に得られたものか、A6 と C11 との間の協定（ルール）に基づいてそうなっているのか不明である。このとき、このインテグリティ規則（つまり x, MN (C11X) → RP (A6,X)）を用いてキュアリを評価すればこの区別は可能となる。

［3］ＮｉｃｏｌａｓおよびＧａｌｌａｉｒｅにより提唱されたアプローチ

（拡張モデル論的アプローチ）

前述の２つのアプローチは、いずれもルールを通り一辺倒に扱っていた：つまり第一のアプローチではすべてが導出ルールとして用いられ、第二のアプローチではすべてがインテグリティルールであった。このアプローチは、第二のモデル論的アプローチに基礎をおくが、ルールは導出ルール、あるいはインテグリティルールのいづれとしても用いられる。そうすると、与えられたルールをどちらに用いるのがより適切であるかの基準が必要となってくる。

たとえば次の例を考える：

・ファクト集合；

CIVIL STATUS (SS number, Age, sex)

11	20	Male
13	45	Fem
12	50	Fem
19	15	Male
15	70	Male
16	68	Fem
17	25	Male
18	80	Male
14	10	Male

LIVE (S.S. NB, Town)

11	New York
12	Paris
13	Paris
16	Syracuse
14	Paris
18	Los Angeles
15	Syracuse
17	Geneva
19	Washington

FATHER (father, son)

12	11
12	14
15	12

GF (grandfather, grandson)

18	19*
15	11*
15	14*

BROTHER (Brother, Brother)

14	17
11	14*
11	17*

MARRIED (husband, wife)

12	13
15	16

・ルール集合

D G1: $\forall x \forall y\ (\exists z(FATHER(x,z) \wedge FATHER(z,y) \rightarrow GF(x,y))$

The father of a father is a grandfather.

D G2: $\forall x \forall y\ (\exists z(BROTHER(x,z) \wedge BROTHER(z,y)) \rightarrow BROTHER(x,y))$

The brother of a brother is a brother.

D G3: $\forall x \forall y \forall z\ (FATHER(z,x) \wedge FATHER(z,y) \wedge (X \neq Y) \rightarrow BROTHER(x,y))$

Two children of the same father are brothers.

I G4: $\forall x \forall y (MARRIED(x,y) \rightarrow \exists z\ CIVIL\ STATUS(x,z,Male) \wedge \exists t\ CIVIL\ STATUS(y,t,Fem))$

In any married couple the husband is a male and the wife is a female.

I G5: $\forall y\ (\exists x \exists z\ CIVIL\ STATUS(x,y,z) \rightarrow (y < 150))$

An age is less than 150.

D G6: $\forall x \forall y \forall z\ (MARRIED(x,y) \wedge LIVE(x,z) \rightarrow LIVE(y,z))$

A wife lives in the same town as her husband.

I G7: $\forall x \forall y \forall z\ (LIVE(x,y) \wedge LIVE(x,z) \rightarrow (y=z))$

Anybody lives in only one town.

I G8: $\forall x\ (\exists y \exists z\ CIVIL\ STATUS\ (y,x,z) \rightarrow AGE(x))$

The second argument of the relation "CIVIL STATUS" is an age.

図4・3 導出ルール/インテグリティルール

[出典は文献D7) Nicolas et al "Database:Theory vs Interpretation" Logic and Database plenum Press NY. '78 より]

モデル論的アプローチにおいては、この例におけるルールはすべてインテグリティルールである。たとえばルールＧ１を考えると、このルールは、新しいタプルがリレーション

ＦＡＴＨＥＲ（ｘ，ｙ）に付加される毎に、この追加操作とリレーションＧＦとが矛盾しないか否かをチェックするのに用いられる。

しかしこれは、素直にＦＡＴＨＥＲ（ｘ，ｙ）への追加操作がＧＦの定義をも変更したとみる方が自然である。つまり、リレーションＧＦはエクスプリシットに定義されたファクトの集合の部分と、インプリシットに導出される情報の部分との２つから成り立っているとみなすべきである。こうすれば※のついたタプルを格納しなくても済み、ストレージの節約につながる。

一方ルールＧ５は、これを導出ルールとして用いると、たとえば"２０は１５０より小さい"というような情報を導くものとなってしまう。これは自明な情報であり、したがって、この種のルールはインテグリティルールとして用いる方がずっと有効となる。

それでは、リレーションＬＥＳＳ（ｘ，ｙ）とリレーションＧＦ（ｘ，ｙ）との差異は何かということが問題となる。ＬＥＳＳ（ｘ，ｙ）のような自明な定義自体は、その外延がアルゴリズムという手段によって既知となっている。このように、アルゴリズムによってか又はエクスプリシットにユーザが定義することによって、その外延がシステムにとって既知となっているようなリレーションをプリミティブリレーションと呼ぶことにすれば、プリミティブリレーション上での情報を推論するようなルールはインテグリティルールと考えた方が良いことになる（例中、Ｉを付したもの）。このようなルール以外のルール（例中、Ｄを付したもの）は、導出のためのルールとなり、例中※を付したタプルは、エクスプリットに格納しなくても済むこととなる。

このようにして、このアプローチでは、導出ルールとファクト集合とが混在するデータベースと、インテグリティルール集合とでシステムが構成されることになる。

ところで、別リレーションを用いてあるリレーションの外延を定義する方法としてデータベースにはビュー定義という機構が備えられている。これは、キュアリ言語表現によりベースリレーシヨン（外延がエクスプリシットに定義されたリレーション）から仮想リレーションを生成するものである。これに対し、ここで示している導出ルールを用いる方法は、リレーション中の各タプルの演繹を行なって、導出リレーションを生成する。

後者の前者に対する利点は、まず後者は、インプリシットとエクスプリシット両方で定義されているようなリレーションを扱うことも可能ということである。例えば例中のリレーションＧＦ（ｘ，ｙ）において、ＧＦ（１８，１９）はルールからは導出できない。これはエクスプリシットに与えざるをえず、従ってＧＦにはエクスプリシットな部分とインプリシットな部分とが混在することになる。また後者は、部分リレーションのみ定義するようなルールも導出ルールとして使用可能なこと（ｅ．ｇ，"Ａ５はＣ１１が作る全ての製品を扱っている"）及び例中、Ｇ２のような再帰的なルールの定義も可能である等の利点を持っている。

以上は、一階述語によるデータベースの形式化に際してのＧａｌｌａｉｒｅ及びＮｉｃｏｌａｓによる議論であるが、この他にも演繹データベース構築に際して問題となる技術課題がいくつかある。

４．３．２　演繹データベース構築に関する技術課題

一つは、述語論理とデータベースとのインタフェースのとり方をインタプリティブに行なうかコンパイルドで行なうかという議論である[16]：

インタプリティブなアプローチは、図４．４に示すように、ルールを用いた推論が進行するに従って逐次データベースをアクセスするという特徴を持っている。

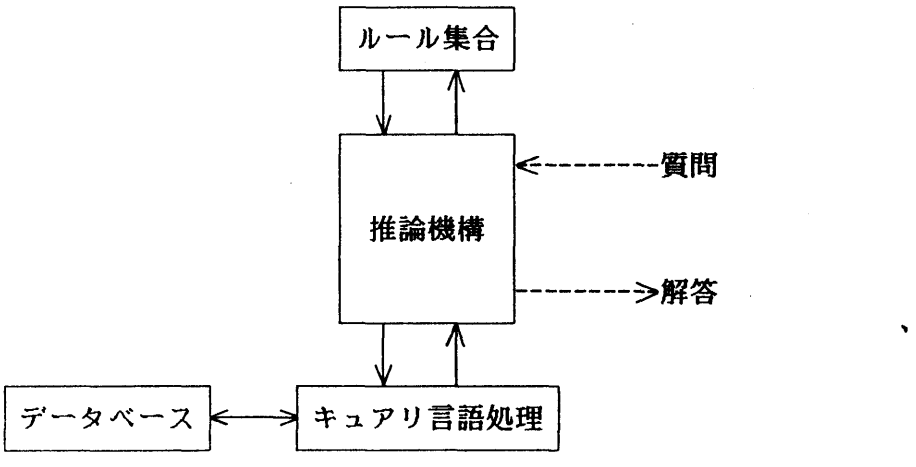


図4.4　インタプリティブアプローチ

このアプローチは、Ｍｉｎｋｅｒ等により提案されたもので推論機構とデータベース及びそのキュアリ言語プロセッサとが密に結合し、推論機構からデータベースへのアクセスが頻繁に行われる際には極めて有効となる。

これに対し、Ｎｉｃｏｌａｓ，Ｇａｌｌａｉｒｅ他によって提案されている方法にコンパイルドアプローチがある。これは、図４．５に示すように、問合せが行なわれると、それをルール集合を用いて一連のキュアリ言語表現に変換し（コンパイルし）、キュアリ言語処理機構に渡して実行させる方法である。

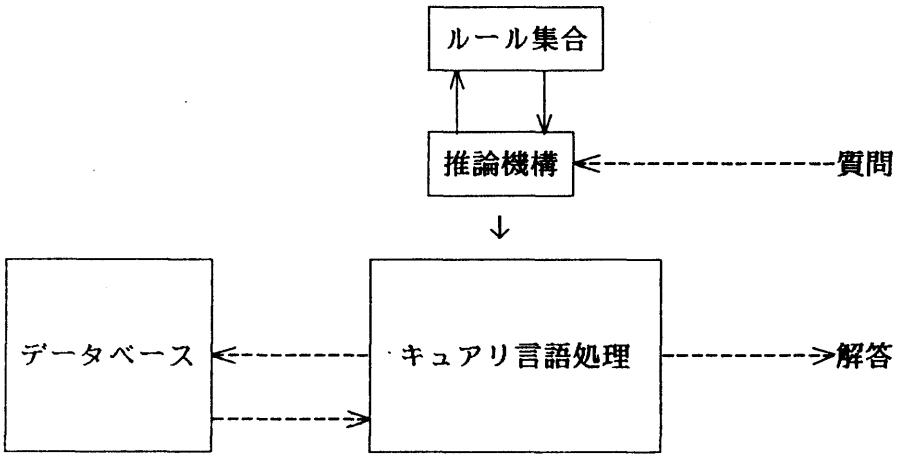


図4.5　コンパイルドアプローチ

このアプローチは、既存のデータベースを使用する場合とか、推論機構とキュアリ処理を含むデータベース管理システムとの間がＬＡＮのような疎結合のインタフェースとなっている場合、特に有効とされている。

また、システムの実現上の技術課題として、ルール集合を便宜的にせよデータベース管理システムの管理下に置くか否かは、処理の効率、管理の容易性等を決定する上で極めて重要な課題となる。例えばメリーランド大学で開発された論理データベースMRPPS3.0では、データの内部表現にテンプレートを使用することによって、ルールもDBMSの管理下に置くアプローチを取っているD13)。

既に開発された演繹データベースの代表例としては、以下のものが上げられるD16)。

①MRPPS(Maryland Refutation Proof Procedure System):メリーランド大学のＭｉｎｋｅｒ教授グループにより開発済のシステム。

②DBGEN:ONERA-CERTのＮｉｃｏｌａｓグループが開発中の演繹データベース管理システム。

③DEDUCE:IBMのＣｈａｎｇが提案しているRDB向の演繹的問合せ言語。

④DADM(Deductively Augmented Data Management System):SDCのＫｅｌｌｏｇらが開発中のシステム。プロトタイプは開発済で、現在、知識ベースシステム拡張中。

⑤KAUS(Knowledge Acquisition and Utilization System):大須賀教授(東大)グループが開発中の多層論理に基づく知識ベースシステム。

⑥AIDS(An Intelligent Database System):ベル研のＮａｑｖｉらが開発中のRDBMSインタフェースを持つ知的データベースシステム。

このうち①は証明論的かつインタプリティブなアプローチ、他はモデル論、又は拡張モデル論的かつコンパイルドなアプローチを採用しているといわれている。

４．４　分散型問題解決システム

演繹推論機構を持つデータベースを、ローカルエリアネットワークを介して複数個結合させ、効率の良いエキスパートシステムの構築を目指す本研究のアプローチは、一種の分散型問題解決システムとして位置づけることができる。このアプローチは、音声理解システムや、車のモニタリングのようなセンサベースシステムあるいはロボットの制御システム等を目的として最近開始されたばかりの極めて新しい研究領域であり、従って、その重要性が認識されつつあるのにもかかわらず、未だ余り研究事例は多くはない。国外での事

例としては、分散型Ａ．Ｉ．研究のフレームワークとしてスタンフォード大学で開発されたＣｏｎｔｒａｃｔ　Ｎｅｔ D4,15) および、ＣＭＵにおける分散型Ｈｅａｒｓａｙ－ＩＩ D11) の経験をもとにＬｅｓｓｅｒ（現マサチューセッツ大学）他により提唱されたアプローチD5,14) とをあげることができる。

Ｃｏｎｔｒａｃｔ　Ｎｅｔは、マネジャとコントラクタという動的に変化する役割を持つ２者間のプロトコルを規定したものであり、一般に問題の部分問題への分割が比較的容易に行えるものには適しているが、そうでない場合には制御が複雑となってしまうことが予想される。これに対し、Ｌｅｓｓｅｒ等のアプローチは、入力データや中間処理結果をプロセッサ間で相互に交換し、それを次の段階における処理やアルゴリズムにフィードバックするものであり、問題を明確に部分問題に分割できない場合や、入力データにあいまいさがある場合には効果的であるが、そうでない場合には、重複処理の発生頻度が極めて高く、非効率なものとなってしまうことが予想される。

また、いづれの事例においても、各プロセッサの保持する知識量あるいは知識の分割の仕方に対するシステムのパフォーマンスといった、分散知識ベースの設計問題に関する定量的な考察は試みられていない。

本研究は、分散型問題解決システムにおけるこのような未解決の問題に対しても考察を試みたものである。本研究の位置づけを一層明確にするため、以下では簡単に、ＬｅｓｓｅｒとＣｏｒｋｉｌｌにより考察された分散型問題解決の論点を概説するD5)。

一般に分散システムは、図４．６のモデルで表すことができる。ここで、分散型問題解決システムにおいては、各処理ノードは、知識ソース（ＫＳ）と呼ばれる一種のエキスパートシステムとなる。

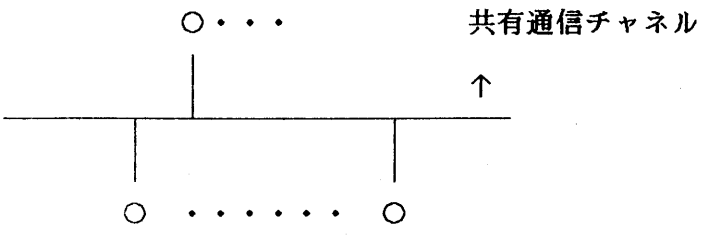


図４．６分散システムのモデル

Ｌｅｓｓｅｒ等は、既存の分散システムに於ては、各ノードは、他ノードによる援助を必要としないで各々の処理を完結できるため、これを Completely Accurate / Nearly Autonomous(CA/NA) と呼んでいる。つまり、各ノードは、完全な正しい情報（Completely Accurate）に基づいて処理を実行し、さらに各ノードは通常、与えられた処理を終結させるために必要な全情報を自身のデータ中に持っている（Nearly Autonomous)からである。

一方、そうした情報が自身のデータ中にない場合は、他ノードに対して要求を出し、処理結果を返してもらうことになる。ＣＡ／ＮＡシステムにおいては 、このような形式のノード間インタラクションは、１つのノードがマスタで他はスレーブであるような形態で実現されている。

このＣＡ／ＮＡアプローチは、アルゴリズムやデータを明確に分割できないような応用分野には適さないものとなる（例：車のモニタ等のセンサベースシステム）。

このような応用に対しては、既存の方式とは異なる新しい分散システムの構成法が考えられる。このアプローチは、各ノードが不完全な入力データを用いながら、他ノードと中間処理結果を交換し合うことにより、協同して完全な解を導くことができるようにシステムを構成する方式である。不完全データに対しても有効な処理をノードが実行できるようにするための１つの方法は、それが常に正しい結果を生成しなくてはいけないという制限をゆるめることが必要となる。そのかわり、各ノードは不完全で不正確な中間結果あるいは、他ノードの中間結果とは相矛盾するような中間結果を生成することも有り得る。つまり、不正確で矛盾を含んだようなデータに対しても解が生成できるような問題解決の構造が必要となる。

このような問題解決機構を持ったシステムのことをFunctionally Accurate（ＦＡ）と呼ぶ。

ＦＡシステムでは、不完全な入力データに対しても、各ノードは処理を実行するのみならず、他ノードからの不完全かつ矛盾を含んだ中間結果に対しても有効な処理を行なう必要がある。このことは、各ノードが、協力し合うことによってエラーのある中間結果を除き、完全で無矛盾な解へ向って収束していくような問題解決のスタイルを意味することになる。このような分散システムは相互に関連するタスクの協調型ネットワークとしてとらえることができ、したがってこのようなＦＡシステムのことをFunctionally Accurate/Cooperative（ＦＡ／Ｃ）と呼ぶ。

ＦＡ／Ｃシステムは、不完全な入力データに対し、データのどの部分が消失したか、又は完全か、無矛盾かといったデータの不確かさを補間する機能を持つ必要がある。

一方、こうしたデータの不確かさの他に、選択の余地のあるタスクのうち、どのようなタスク群を実行させれば余分なノード間通信を行なわないで済み、全体として有益かといったコントロールの不確かさも存在する。

完全なコントロール情報を持たなくとも、各ノードがコントロールの決定を行なえるようにするためには、各ノードをより、self-direceted にする必要がある。このため、各ノードは、処理結果のローカルな評価値を用い、それを他ノードに転送する。ＦＡ／Ｃシステムにおけるこのような、ノードの self-directed 性は極めて大きなものとなる。なぜなら、すべての必要データが必ず入手できるとも、またそのデータが他ノードのデータと無矛盾であるとも限らない状態で、処理の方向を選択する機能を各ノードは持つことになるからである。したがって、各ノードによってなされる self-directed な決定により、冗長な処理、不必要な処理が導かれてしまう可能性があるけれども、このようなコントロールによって通信路やノード障害に対するシステムの robustness が向上し、不測の事態に対するシステムの応答性も向上する。

ここで、ＣＡ／ＮＡおよびＦＡ／Ｃにばかり焦点を絞ってきたが、もちろんCompletely Accurate/Cooperative（ＣＡ／Ｃ）あるいはFunctionally Accurate/Nearly Autonomous（ＦＡ／ＮＡ）といった分散システムも存在する。実際、ほとんどの分散システムは、この４つの極のどこかに位置づけられる。

入力データが不確かな場合にはＦＡアプローチが、ノードが何を実行すべきかといったコントロールが不確かな場合はＣアプローチが適切となる。しかしながら、ほとんどの分散システムは基本的にはＣＡ／ＮＡかＦＡ／Ｃかのどちらかであると考えられる（図４．７参照）。

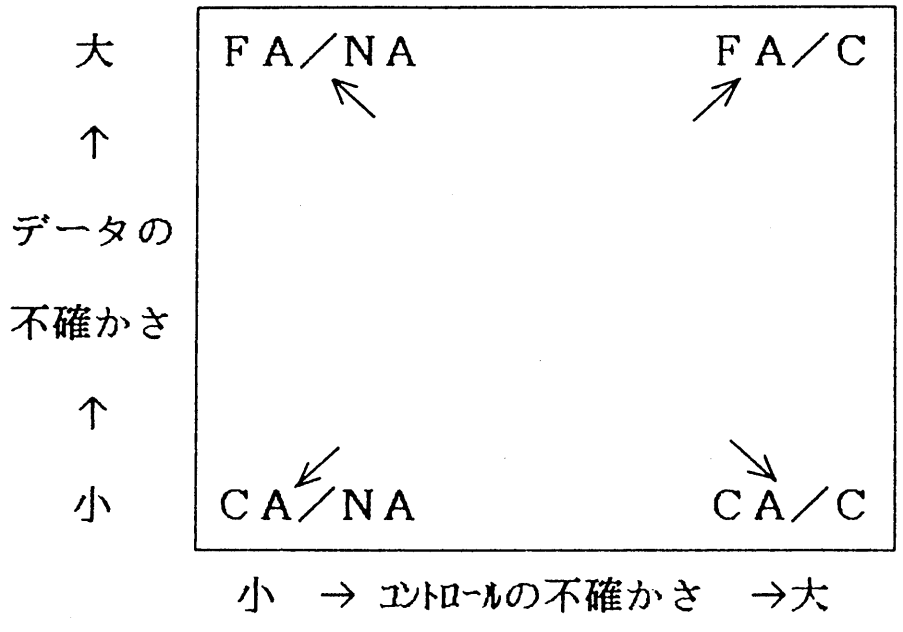


図４．７　データとコントロールの不確かさ

というのは、データの不確かさとコントロールの不確かさとは相互に関連し合って出現するからである。データの不確かさがあるとノード間での定形的なインタラクションのパタンを決定するのが困難となり、コントロールの不確かさを発生させる。一方、コントロールの不確かさがあると、中間処理結果が、不完全であったり、矛盾を含んでいたりするため、データの不確かさを伴うことになる。逆に、データの不確かさが少なくなれば、ノード間インタラクションの不確かさも減じられ、この場合にはＣＡ／ＮＡアプローチが適切となる。

４．５　本章の位置づけ

本研究の主目的は、汎用のマイクロコンピュタをベースとして実用的エキスパートシステムを構築することにあった。このため、本システムにおける演繹データベースではハードウェア上の特殊な機構を開発するというアプローチはとらず、探索アルゴリズムの工夫、並列サーチ機構の採用という方法で対処した。

一方、既存データベースのとり込みという課題に対しては、可能な限り汎用のデータベースを取り込むべく検討を行ったが、マイクロプロセッサのみを使用して高パフォーマンスを達成することは困難であった。このため、本システムに於るデータベースは、データベース設計時に内部スキーマを変更することができ、適用形態に応じてユーザが最も検索効率の良い内部構造を指定するという特殊なアプローチをとっている。このアプローチは、通常のマイクロコンピュータを使用して、多量のファクトデータを扱う実用的エキスパートシステム構築に対する１つの可能な解決策であると考えている。

また本システムでは、拡張モデル論的アプローチを採用したが、インテグリティ規則は、現在の所入れていないため、証明論的アプローチと同様のものとなってしまっている。

一方分散型問題解決のシステムとしてみたときの本システムは、意味で本研究におけるアプローチは、Ｌｅｓｓｅｒ等が Completely Accurate/Cooperative と呼ぶ範ちゅうに入れられることになり、また各ノードの self-directed性の極めて強いアプローチとなる。なぜなら、演繹データベースによる分散型問題解決においては、入力データと中間処理結果とは全く分離して扱われており、各知識ソースの保持する知識や与えられる入力データである問合わせ情報そのものには誤りや不完全さはない。この意味で Competely Accurate なアプローチといえるが、一方不完全な中間処理結果は、次に何を実行すべきか、又いつどのような情報を他知識ソースに渡すべきかというコントロールに関する不確かさを生成することになり、そのために生ずるあいまいさを協調的方法で解決せざるを得ない。

このような観点からみれば、本研究はＬｅｓｓｅｒ等の云うＣＡ／Ｃ分散システムに焦点を絞り、システムのモデル、性質、分散知識ベース設計問題とその評価、具体的実現例等の考察を試みた研究であると位置づけることができる。

４．６　分散型推論サブシステムに関する基本的考察[D24)]

前節で論じたように、本システムにおける推論サブシステムは、大容量ファクト集合の格納場所としてのデータベースに、述語論理に基づく演繹推論機構を組み込み、さらに分散型問題解決を実行するためのプロトコルを導入したものとなっている。

なお本サブシステムは、応用プログラムに対し、推論機構を介して検索を行う演繹検索と、直接データベースをアクセスする単純検索との２種のインタフェースコマンドを設けている。

このような方式をとった理由を以下に示す：

① ＳＤ^{3}システム全体を、推論システム（演繹検索）としても、又データベースシステム（単純検索）としても利用することが可能となる。意志決定支援、計画策定支援といった実際の応用においては、この２つのシステムを同時に利用する場合が多い。

② 両者を同一コマンドインタフェースで統合した場合、演繹推論機構側で集合(Set)の概念を扱う必要が生ずる。これは、一階述語の範囲では扱えなくなり、例えば多層論理(many sorted logic)等の概念拡張が必要となり、それを実現する機構も複雑となる。

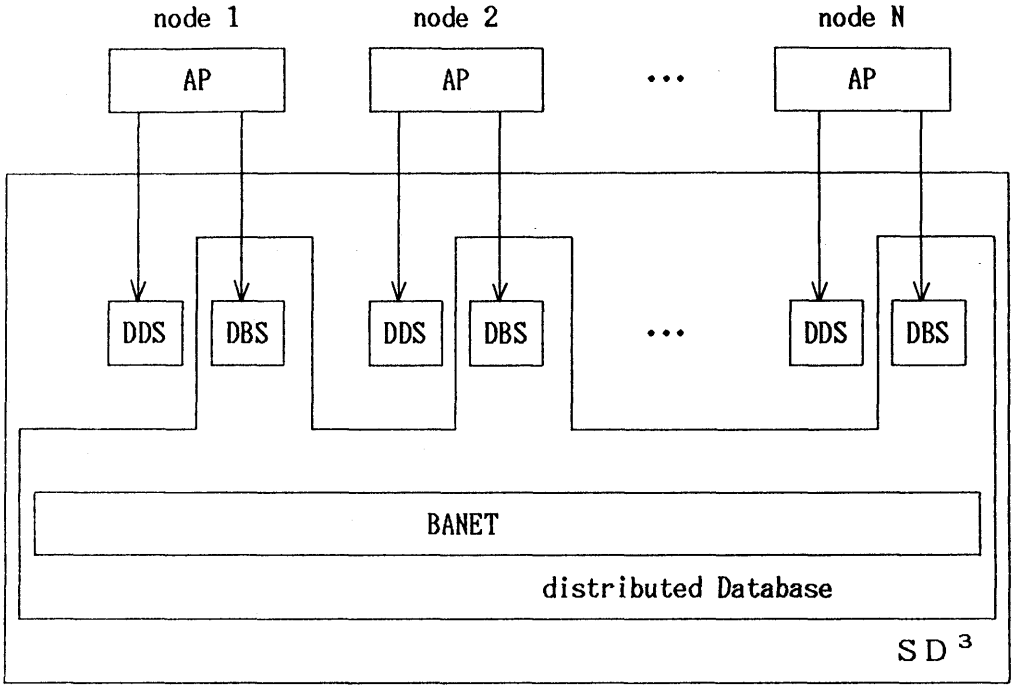
③ 両者を統合した場合、単純検索であっても、演繹推論機構を起動することになるため、直接応用プログラムがデータベースを参照する場合に比べて効率が低下する。

４．６．１　システムの論理構成

ＳＤ^{3}の全体構成を図４．８に示す。図４．８において、ＡＰは具体的問題に依存した手続き部分である。各ノードは演繹推論を実行するＤＤＳ(Data Deduction System)とデータベースの検索を実行するＤＢＳ(DataBase System)とからなっている。

各ノードのＤＢＳを統合することによって分散データベースサブシステムが構成され、

さらにＤＤＳを結合させることにより、分散型問題解決サブシステムが構成される。



AP　:Application Program

DDS　:Data Deduction System

DBS　:DataBase System

BANET :Broadcast Architecture NETwork

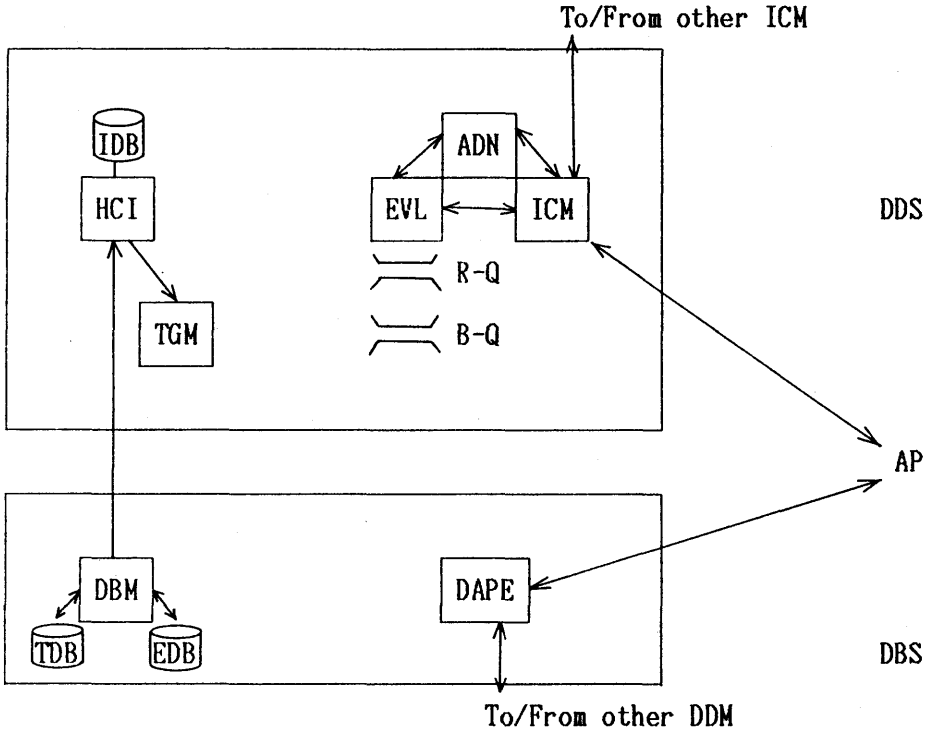
図４．８　ＳＤ³の構成概要

各ノードにおける論理モジュール構成を図４．９に示す。図４．９において、ＩＣＭ（Internode Communication Manager）は分散型問題解決のプロトコルを実行し、ＡＤＮ（Area Description）を 用いて他ノードからの受信情報が自ノードに関連するか否かの判定、他ノードへの自身の導出した情報の送出等を行う。

ＥＶＬ(evaluator)は実行のストラテジーをたてる。具体的には、次に展開すべき探索木上のノードを選択し、ゴール節形式としてＨＣＩに実行を指示する。展開する深さはＤＢＭへの書き込み内容により指示できるので、評価関数に沿った探索、全解探索のいずれも実行可能であり、種々の探索パターンに適用することが可能である。

ＨＣＩ (Horn Clause Interpreter) はＩＤＢを参照することにより、ホーン節で記述されたルールの実行を行なう。 外延的に定義された述語の実行が必要となれば、 キュアリを生成し、ＥＤＢへのアクセス要求としてＤＢＭに通知する。

もしルールの適用が不可能となったときは、FailをＥＶＬに返す。また、ルール実行の都度、ＨＣＩはＴＧＭに制御を渡し、トリガの実行を行なわせる。



IDB : Intentional DataBase

TDB : Tempolary DataBase

EDB : Extentional DataBase

HCI : Horn Clause Interpreter

TGM : Trigger Mechanism

DBM : DataBase Manager

EVL : Evaluator

ADN : Area Description

ICM : Inter-node Communication Manager

DAPE: Database Access Protocol Executor

R-Q : Ready Queue

B-Q : Blocked Queue

図４．９　ノード内論理モジュール構成

ＩＤＢ（Intentional DataBasa）にはホーン節で記述されたルールベース、関数定義等が格納されている。ＥＤＢ（Extentional DataBase）には導出された中間結果、およびＥＶＬにより指示された探索ストラテジーが格納される。

ＴＧＭ（Trigger Mechanism）は探索の途中で導出された中間結果をＴＤＢに書き込むためのトリガ機構である。

４．６．２システムの動作概要

ＳＤ^{3}の各ノードのモジュール間の関係をより明確に示すため、ある人（太郎とする）の孫を求める問題を例にとり、システムの動作を概説する。本システムでは、この問題に対して以下のデータベースが必要となる。但し、ｇ（ａ，ｂ）はリストｂに要素ａを追加する関数、ｆ（ｘ）はリストｘの最初の要素を取り出す関数とする。

ＩＤＢ

Rule 1 ：A (h,x) ← EA (h,x)

Rule 2 ：A (h,g(f(x),y)x) ← HPC(i,h), A(i,y), P(f(y),f(x)),
UNUSE-CHECK(i,h,g(f(x),y))

トリガ ： [Insert to EA] on Rule 2

ＥＤＢ

P(x,y) ： xがyの親であることを示す。
レコードとして右表の内容を仮定する。

親	子
太郎	二郎
太郎	花子
二郎	三郎
花子	四郎

図4.10 データベース例

ＴＤＢ

ＥＡ(h,x) ： 識別子ｈを持つ中間結果の内容がリストｘであることを示す。

ＨＰＣ(i,j) ： 識別子ｊを持つ中間結果が識別子ｉを持つ中間結果の直接の後続結果であることを示す。

本システムにおいて特徴的なことは、導出の過程で用いられる中間結果に識別子が付されることを前提とし、それに基づきいくつかの述語が定義されることである。 述語A(h,x) は識別子hを持つ中間結果の内容がリストxであることを示し、ＥＡ (h,x)は同じ関係を示すが、既に導出済みで、ＴＤＢに格納されていることを表す。xは例えば、リスト（三郎．二郎．太郎）であり、太郎から始まる親子関係の例である。

ＩＤＢ内のルール１はもしＴＤＢの中にすでに中間結果ｈとリストの関係がＥＡ (h,x) として格納されていれば、述語Ａ (h,x) も成り立つことを示す。ルール２は中間結果ｉをさらに１ステップ展開して後続の新しい中間結果hを導出するためのルールである。ルール２においてUNUSE-CHECKは、同一の親中間結果ｉを持つ導出中間結果ｈの内容xが、既に導出してＥＡ中に格納されている中間結果と重複しないことを保証する組込み述語である。

今、太郎の孫を２人だけ求めるには、以下の手順が実行される。

① ＥＶＬは、探索のストラテジーとして、ＴＤＢ内のＨＰＣに、レコード (h0,h1)、(h0,h2)、(h1,h3)、(h2,h4) を書きこむ。また、ＥＡにはレコード (h0,太郎) を初期値（探索のルート）として書き込む。

② ＥＶＬはＨＣＩに対し、ゴール節 ← Ａ (h3,x)を出す。

③ ＨＣＩはルールを順次適用し、まずルール２よりＡ (h1,(二郎.太郎)) を求める。

④ トリガが起動され、レコード(h1,(二郎.太郎))が中間結果としてＥＡに書きこまれる。

⑤ ＨＣＩはルール２を再帰的に実行しているから、再びルール２によりＡ (h3(三郎.二郎.太郎) を求める。

⑥ トリガが再び起動され、レコード (h3(三郎.二郎.太郎)) がＥＡに書き込まれる。

⑦ ＨＣＩはＥＶＬに解ｘ＝（三郎.二郎.太郎）を返す。

⑧ ＥＶＬはＨＣＩにゴール節 ← Ａ (h4,x) を出す。

⑨ 同様の処理が行なわれるが、ルール２の末尾の述語UNUSE-CHECKにより、Ａ (h2,(二郎．太郎)) は解から除かれる。

⑩ 上記③～⑦ と同様の処理が実行され解ｘ＝（四郎.花子.太郎）を返す。

このように本システムは、データベースの演繹検索を木探索の問題に置き換えて実行するものであり、ＥＶＬがＨＣＩに与えるゴール節は、中間結果の識別子を指定して、その内容を問うという形式になっている。ここで、問われるのは中間結果の内容である必要は必ずしもなく、場合によってはその評価値であっても良いということに注意されたい。

通常、評価値による探索問題においては、中間結果自体は探索ストラテジーの決定には何も寄与しない。このため、現在開発中のアプリケーション（略地図発生システム）においては、評価値のみを得て次の探索ストラテジーを決定し、中間結果はＴＤＢに蓄積のみ行う方式をとっている。

４．６．３　分散型演繹検索機構

ＳＤ^{3}分散型推論システムにおいては各ノードは木探索を実行しているが、同時にまた、システム全体でも、組織的な一つの木探索を行っている。

以下では、こうしたＳＤ^{3}システムにおける探索方式を形式的に扱うため、まずいくつかの用語を定義する。

［１］定義

① 中間結果が Fail もしくは Success 値のみを直接の後続中間結果として持つ場合、これを『終端結果』もしくは『解』と呼ぶ。また先行中間結果を持たない中間結果を『初期値』もしくは『問題』と呼ぶ。

② システム内のあるノードに解でない一つの中間結果が与えられたとする。その直接の後続中間結果の少なくとも一つがそのノードで導出可能なとき、与えられた中間結果はそのノードにとって『展開可能』(expandable)であるという。

さらに、組織的な一つの探索木全体におけるすべての後続中間結果をそのノードで導出できるとき、その中間結果は、そのノードにとって『全展開可能』(totally expandable)であるといい、そうでない場合は、『半展開可能』(partially expandable)であるという。

③ あるノードが展開可能な中間結果の集合をそのノードの『関連域』（Area of Interest）と呼び、さらに全展開可能な中間結果の集合をそのノードの『達成域』（Area of Completion)と呼ぶ。

関連域と達成域という概念を用いて、分散システムの編成形態を類別することができる。つまり、一つは、任意の中間結果に対し、その中間結果を達成域に持つようなノードがシステム内に少なくとも一つ存在する場合で、ここではこれを『全編成』された（totally organized）システムと呼ぶことにする。

これに対し、与えられた中間結果を自分の関連域内に持つすべてのノードがその展開を実行して直接の後続中間結果を導出し、その和集合をとれば、全体の探索木上の与えられた中間結果の直接の後続中間結果をすべて導出したことと等価となるシステムもある。このようなシステムを『半編成』された（partially organized）システムと呼ぶ。

一方、各ノードの保持するルールあるいはファクト集合を与えられた問題に応じて適宜移動させなければ、すべての解の導出が不可能な場合もある。このようなシステムを『未編成』（disorganized）システムと呼ぶ。したがって未編成のシステムではそのまま演繹検索を行っても必ずしも解に到るという保証はない。この保証を得るためには未編成もしくは全編成のシステムに構成し直す必要がある。

次に、分散データベースにおけるジョイン演算の例を用いて、これらの差異を論ずる。

例えば、Ｐ及びＱをバイナリリレーションとする。

今、Ｐの第１カラムをある値によって制限し、Ｐの第２カラムとＱの第１カラムとのジョインを行なうキュアリ（e.g. P(a,y) ∧ Q(y,x)）を考える（これは深さ１の木の全解探索とみなすこともできる)。今、ＰとＱとを３つの部分Ｐ１，Ｐ２，Ｐ３，Ｑ１，Ｑ２，Ｑ３，に分割し、対（Ｐｉ，Ｑｉ）をノードＮｉに格納したとする(図4.11）。このように配置したシステムでは、格納データの移動を伴わないで解を求めることはできない。それ故、これは未編成システムとなる。

一方、Ｐはそのままでｑを重複して全ノードに割り当てたとする。この場合は各ノードのローカルなジョイン結果をあわせれば解になるという意味で半編成システムである（図4.12）。

次に、Ｐを第１カラムでソートし、第１カラムが同一値を持つタプルは同一ノードに収容されるようＰを再分割したとする（図4.13）。これは、一つのノードでのジョイン演算の実行により、必ず解が得られるから、全編成システムとなる。

ＣＡ／Ｃ分散システムにおいて、未編成を全編成もしくは半編成に構成し直す際に特に問題となるのは、ファクトのみならずルールの転送の必要が生ずるということである。これは、ファクトの転送のみ扱うのが従来の分散データベースであったのに対し、いわば広義の分散データベースの構築ということができる。

なお全編成システムにおいては、各ノードの関連域を実際の関連域よりも縮小させ、達成域と同一の領域に定義し直してもシステム全体の機能は変わらない。このとき、関連域、達成域あわせて単にエリアと呼ぶ。

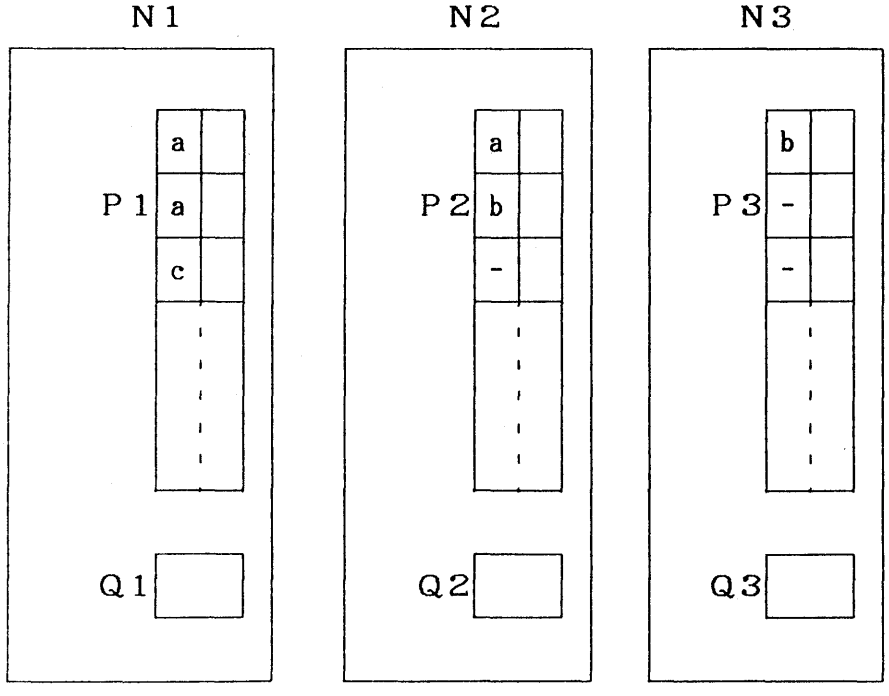


図４．１１　未編成システムの例

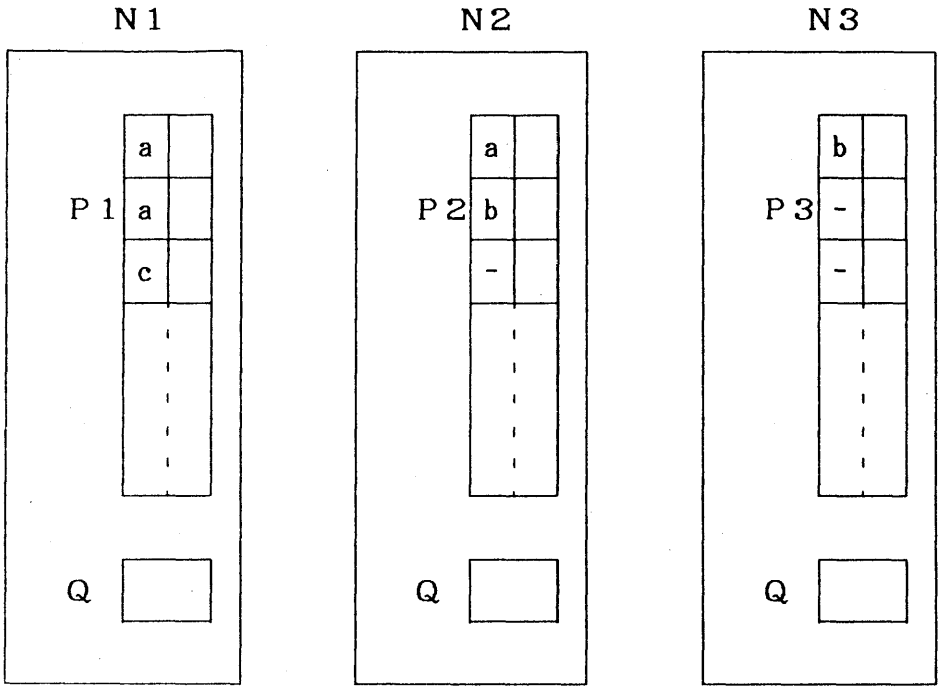


図4．12　半編成システムの例

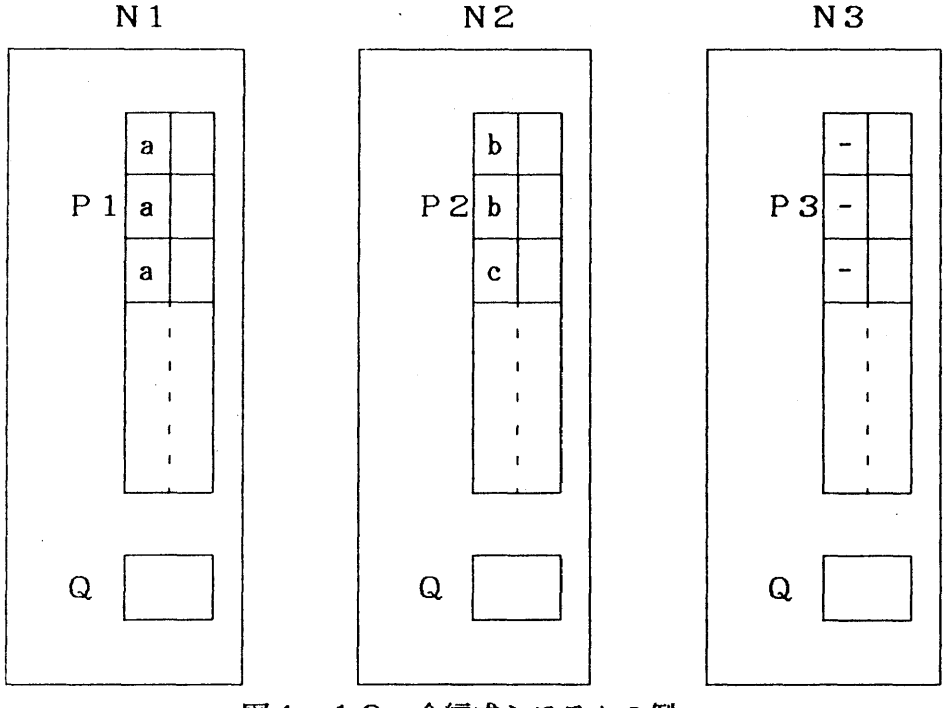


図4．13　全編成システムの例

以下ではＳＤ^{3}システムにおける分散型推論のためのプロトコルを概説する。なおシステムは全編成もしくは半編成を仮定する。

［２］プロトコル概要

① イニシエータノードに問題Ｓが与えられる。イニシエータノードはＳを協調メッセージにより全ノードにブロードカストする。

② Ｓｉ（初めはｉ＝ｏ）を含む協調メッセージを受信したノードＮｉは、それがＮｉの関連域にあれば、優先度Ｐｉを付加して競合メッセージを全ノードにブロードカストする。付加する優先度はシステムが全編成か半編成かによって異なる。全編成の場合、Ｓｉの自ノードのＲ－Ｑ中の順位Ｒｉとノード番号Ｎｉとの対とする。半編成の場合、Ｓｉが達成域にあれば１、そうでなければＯとする。

③ 競合メッセージを送出したノードは直ちにＳｉをＲ－Ｑにおき、展開の対象とする。

④ ＮｉはＳｉを順次展開してゆき、導出した中間結果ＳｊがＮｉの達成域を越えた場合には、Ｓｊを協調メッセージにより全ノードにブロードカストする。またＮｉの達成域を越えない場合でも、新たに導出した中間結果の数があらかじめ規定された個数に達した場合には、その時点でＲ－Ｑの２番目にある中間結果を協調メッセージでブロードカストする。

⑤ Ｎｉは自身の関連域に解以外の中間結果が存在しなくなるか、またはイニシエータノードから終結要求が出されるまで、Ｒ－Ｑ内にある中間結果の展開を続行する。

⑥ 他ノードＮｋからのＳｉに対する競合メッセージを受信したＮｊは、優先度を比較し、自身の方が低ければＲ－Ｑより、Ｓｉおよびその後続中間結果を取り除く。優先度が同じか、自身の方が高ければ処理を続行する。但し優先度は、全編成の場合ＲｊとＲｋの少ない方が優先し、もし同じであればノード番号が比較される。半編成の場合は１がＯに優先する。

４．６．４　ＣＡ／Ｃ分散システムの評価

本項では評価対象として、全編成システムにおける全解探索問題をとりあげる。このとき、システムの実行効率に特に影響を与える要因として重要なものは、各ノードの保持する達成域の大きさである。つまり、達成域を大きくし過ぎると並列実行による効果がほとんど得られない（問題が単一ノードに閉じてしまう）し、また達成域を小さくし過ぎると通信オーバーヘッドが極めて大となってしまうことが予想される。

この場合の最適な達成域の大きさを、本項では展開の結果生ずる直接の後続中間結果が、もとの中間結果と同一達成域に入る確率という観点から論ずる。

［１］前提

① １つの中間結果を展開して直接の後続中間結果を１つ生成するのに要する時間を一定値　u sec とする。

② １つの中間結果を他ノードに転送するのに要するデータのビット長をＢ、通信チャネルの実効スループットをＴ bit/secとする。ｒ＝Ｂ／Ｔは、１つの中間結果を別ノードに転送するため通信チャンネルを専有する時間となる。従って通信・処理比Ｃは、Ｃ＝ｒ／ｕとなる。

③ 中間結果の展開処理と送受信処理は、完全に並列に実行されるが、通信チャンネルは単一であり、すべてのノードが共有する。

④ 確率変数Ｚを、１つの中間結果から発生する全ての直接の後続中間結果の総数とする。もし与えられた問題が解を持つのであれば　Ｅ(Ｚ) ＝ρ＜１である。

⑤ 確率変数Ｘ，Ｙをそれぞれ、先行中間結果と同一達成域に属する直接の後続中間結果の総数、および異なる達成域に属する中間結果の総数とする。

⑥ Ｅ(Ｘ) ＝λ，Ｅ(Ｙ) ＝μとし、σ＝λ／(λ＋μ) ＝λ／ρとおく。σを探索のローカリティと呼ぶ。これは、直接の後続中間結果が、先行中間結果と同一達成域に属する確率である。

⑦ 与えられた問題を第０世代、それから直接生じた中間結果の集合を第１世代、・・・と呼び、各世代集合の要素の数の分布をZ_0, Z_1, ・・・, Z_n, ・・・であらわす。

但し、$Z_0=1$, $Z_1=Z$である。

［２］探索ローカリティの最適値

$\phi(z)$ をZの分布の母関数 、$\phi x(z)$, $\phi y(z)$ を それぞれX，Yの分布の母関数とする。今、Zn の分布の母関数を$\phi(n)(z)$ であらわすと、 変数Z_0, Z_1,・・Z_n・・は分技過程 (Branching Process) [E1,2] を構成するから、

$\phi^{(n)}(z) = \phi(\phi^{(n-1)}(z))$, が成り立つ。

したがって、第ｎ世代集合の要素の数の平均$\bar{Z}n$ は、

$$\bar{Z}n = \left[d\phi^{(n)}(z) / dz \right] = \phi'(\phi^{(n-1)}(1)) \cdot \phi^{(n-1)}(1),$$

$\phi'(1) = \rho$であるから $\bar{Z}n = \rho^n$となる。したがって、第０世代から解に到る迄に生じた要素の総数の平均$\bar{z}$は、

$\bar{z} = \Sigma \bar{Z}n = 1/(1-\rho)$ となる。したがって単一ノードで全探索を行った時要する平均時間 Tmax は、

$Tmax = u/(1-\rho)$ eq.(1) である。

一方、Z＝X＋Yであり、また展開された中間結果の属する達成域が先行中間結果のそれとは異なる場合、１回の通信を伴なうから、ｎ－１からｎ世代を生じたとき要する通信回数Ynは、変数Yの独立なZn-1回の和となる。したがってYnの分布の母関数は

$\phi^{(n-1)}(\phi y(Z))$で与えられる。故にYnの平均値$\bar{Y}n$は eq.(2) で与えられ、

$$Yn = d\phi^{(n-1)}(\phi_y(Z))/dZ = \phi^{(n-1)'}(\phi_Y(1)) \cdot \phi_Y'(1) \qquad \text{eq.(2)}$$

探索が終結する迄に要した通信時間の平均値 Tcom は、eq.(3) で与えられる：

$$Tcom = C \cdot u \cdot (\Sigma Yn) = C \cdot u \cdot \mu/(1-\rho) \qquad \text{eq.(3)}$$

一方、ある達成域に属する１つの中間結果が同一達成域に後続中間結果を生成しなくなるまでの平均時間を Tmin とすると、ρ を λ で置き換えた時の Tmax と同様にして、Tmin $= u/(1-\lambda)$ を得る。これは探索に要する最小時間の平均値を与えているとみなすことができる。$\sigma = \lambda/\rho$ から、eq.(4)、eq.(5) を得る：

$$Tcom = c \cdot u \cdot \rho\ (1-\sigma)/(1-\rho) \qquad \text{eq.(4)}$$

$$Tmin = u/(1-\rho\sigma) \qquad \text{eq.(5)}$$

図4.14に $C=10$, $\rho=0.99$ とした場合の Tmax, Tmin, Tcom を示す。σ の最適値は通信時間と処理時間の等しい所、図4.14の例においては、0.895 から 0.975 までの間にあり $\sigma > 0.975$ で処理ネック、$\sigma < 0.895$ で通信ネックとなることがわかる。なお、これはノード総数、達成域の重複の程度に無関係で、ρ および C の値によってのみ定まる不変的な性質であることに注意されたい。

一般に最適解 σ_{opt} は、不等式 $\sigma_0 \leqq \sigma_{opt} \leqq \sigma_1$ を満たす。但し、σ_0 は方程式 Tmax=Tcom の σ についての解、σ_1 は Tmin=Tcom の σ についての解である。

eq.(4) および eq.(5) と、$\sigma<1$ とから、eq.(6) を得る：

$$\sigma 1 = (1+\rho((1-\rho)^2 + 4(1-\rho)/C)^{1/2})/2\rho \qquad \text{eq.(6)}$$

同様に、$\sigma_0 = 1-1/(C\rho)$ となる。$0 \leqq \sigma$ だから、$C\rho < 1$ の場合、σ_0 は定義されない(i . e .どのような σ の値に対しても通信ネックとはならない)。したがって σ_{opt} の範囲は、一般に ineq.(7) であらわされる：

$$\max(0,(1-(C\rho)^{-1}) \leqq \sigma_{opt} \leqq (1+\rho-((1-\rho)^2 + 4(1-\rho)/C)^{1/2})/2\rho \qquad \text{ineq.(7)}$$

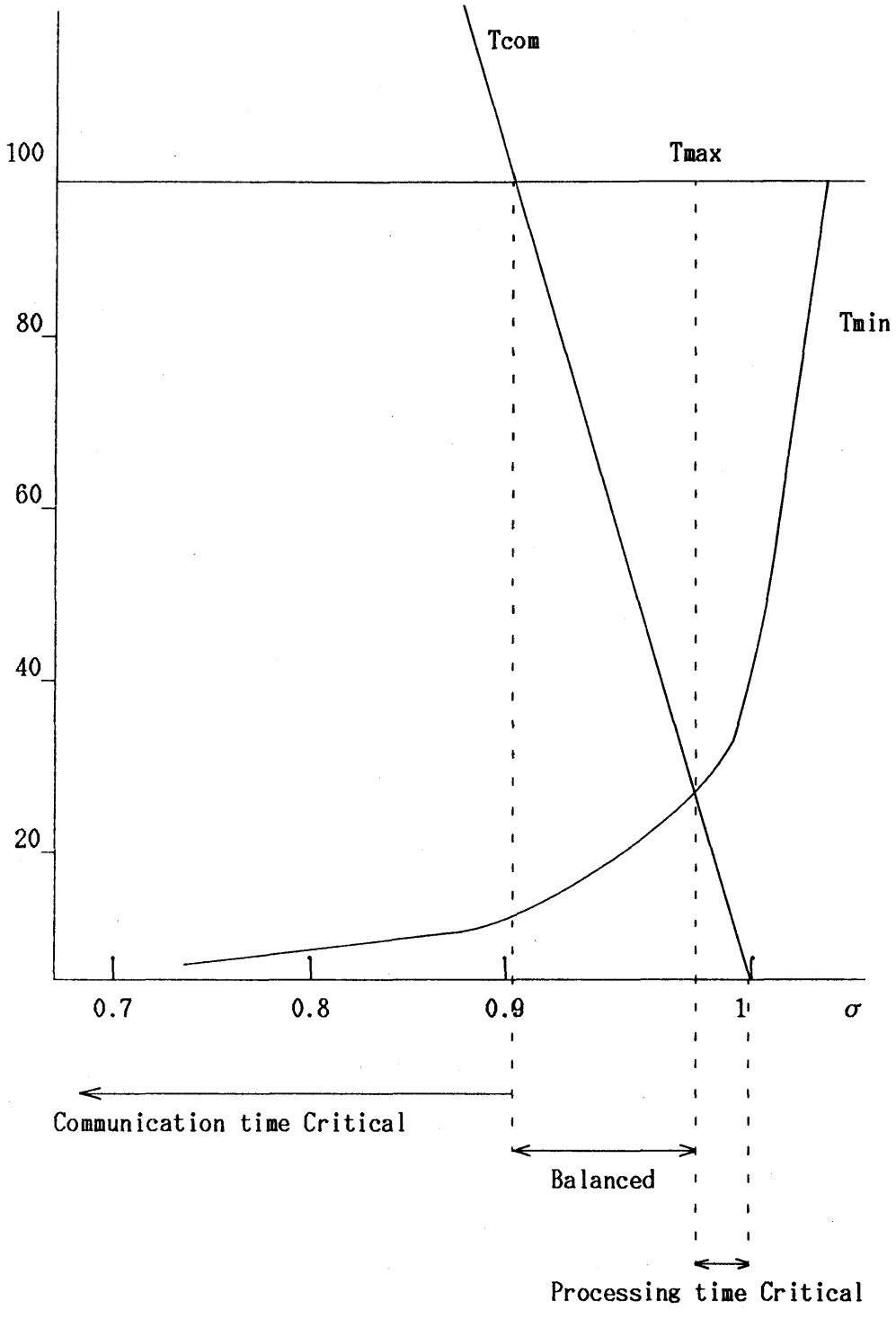


図4．14　C＝10，ρ＝0.99 の場合の Tmax，Tmin，Tcom

第　5　章

本システムの具体的応用事例

本章では、SD^3システムの具体的応用例である略地図生成システムのハードウェア／ソフトウェア構成および動作概要を述べることにより、SD^3システムにおける検索と推論の方式およびルールとデータベース等をより具体的に議論する。

５．１　システムの目的

前述のようにＳＤ^{3}システムは、汎用のマイクロプロセッサを用いて実用的なエキスパートシステムを実現するためのフレームワークを研究することを主目的として開発された分散型問題解決システムである。一方、ＳＤ^{3}システム開発の過程で実現されたローカルエリアネットワークや分散データベースは、サブシステム自体でも十分実用に供し得るものであり、それぞれ分散データ処理向きＬＡＮとして、又定型トランザクション処理向きのデータベースシステムとして、実システムへの適用が検討されている。

またＳＤ^{3}システムは、汎用的な問題解決を目指したものであるが、システムを真に汎用的なものとするためには、いくつかの具体的応用例を設定し、システムの評価、実験を行う必要があると判断した。これにより、システムのパフォーマンスを評価できるだけでなく、さらに具体的応用に依存するソフトウェアモジュールと汎用的に適用できるソフトウェアモジュールとを一層明確に区分できるということも期待された。このような具体的応用の１つとしてＳＤ^{3}システム上にインプリメントされたものは略地図生成システムである。これは、エンドユーザにより指定された地図上の任意の２点間の（準最適な）径路を求め、その２点を結ぶ略地図を生成するというもので、一種のプラン生成システムといえる。

このような応用を設定した理由は以下の通りである：

（１）特定地区における現実の地図情報をデータベース化することにより、応用例自身が実用的価値を持つ。

（２）この応用においては、システムは比較的大規模なファクト集合（地図データ）と、比較的小規模のルールとを持つことになり、演繹データベースの応用例としては適切なものとなる。

（３）一定条件を満す全径路を探索することも、あるいは１つの経路のみ探索することも自由に設定することができ、各種評価データを収集する為には都合が良い。

（４）解決すべき問題自体にはあいまいさがなく、また保持するデータベースの内容にも誤りはない。したがって典型的なＣＡ／Ｃシステムの例となる。

なお以降では、このＳＤ^{3}上の略地図生成システム ── ＳＤ^3・ＭＧ（Map Generator）のインプリメンテーションを述べることにより、あわせてＳＤ^{3}システムの具体的なハードウェア、ソフトウェア構成、および応用プログラムとＳＤ^{3}システムとのインタフェース等を論ずることとする。

５．２　システムの概要

ＳＤ^3－ＭＧシステムは、図５．１に示すように、５つのノードとそれらを結合するローカルネットワークとから成っている。

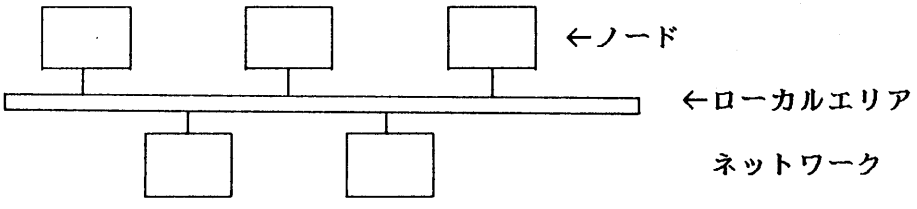


図５．１　ＳＤ^3－ＭＧシステムの構成

ユーザは、どのノードからでも問題の入力を行うことができ、問題解決のプロセスは、５つのノードによって分散型で実行されることになる。但し、現行のシステムでは、１つのノードに於て問題が入力されると、その問題解決が完了するまで他のユーザからの入力はどのノードにおいても受け付けられない。

各ノードのソフトウェアは、汎用的な問題解決を実行する部分（データベース検索、演繹推論）とユーザインタフェースを実行する応用ソフトウェア部とから成る。したがって、応用ソフトウェアは、ユーザの問題入力があったノード（以降イニシエータノードと呼ぶ）においてのみ実行されることになる。

５．２．１　ハードウェア構成

各ノードのハードウェア構成を図５．２に示す。各ノードはＤＰ（Data Processeor）、ＮＰ（Network Processeor）によるマルチプロセッサ構成をとっている。ＤＰは各ノードローカルな演繹推論およびデータベース検索を、ＮＰは他ノードとの通信およびプロトコル処理をそれぞれ実行する。

なお、現在、ＤＰには汎用のパーソナルコンピュータ（ＯＫＩ IF800/M50）を、ＮＰには、I8086ベースのネットワーク処理専用コンピュータを使用している。

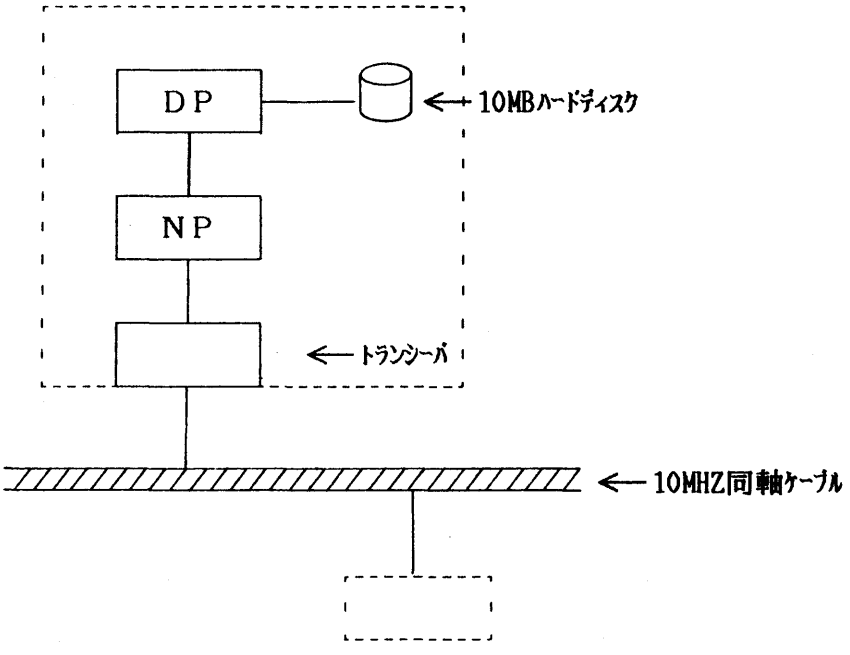


図５．２　各ノードのハードウェア構成

またＤＰとＮＰとはＩＥＥＥ４８８ケーブルを介して結合されており、ＮＰどおしは、トランシーバを介して10MB/secの同軸ケーブルに相互接続されている。

なお、ＤＰおよびＮＰのハードウェアブロック構成および諸元を各々図５．３、５．４および表５．１、５．２に示す。

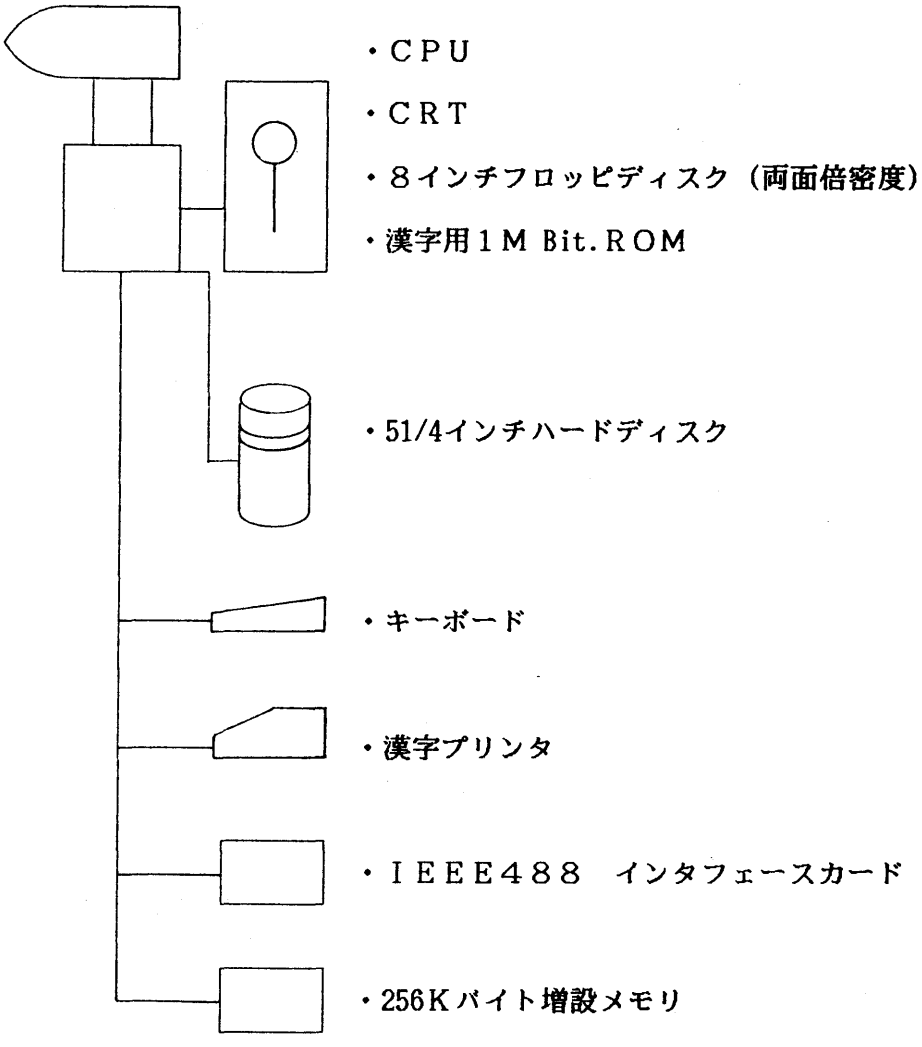

・ＣＰＵ
・ＣＲＴ
・８インチフロッピディスク（両面倍密度）
・漢字用１Ｍ Bit.ＲＯＭ
・51/4インチハードディスク
・キーボード
・漢字プリンタ
・ＩＥＥＥ４８８　インタフェースカード
・256Ｋバイト増設メモリ

図６．３　ＤＰ部の構成

＜ＣＰＵ＞

使用ＣＰＵ	8086-2
ＣＬＯＣＫ周波数	7.9872 MHz

＜メインメモリ＞

使用素子	64KBIT/CHIP　ﾀﾞｲﾅﾐｯｸRAM
容量	256KBYTE 実装

＜ＣＲＴ及びＣＲＴインタフェース＞

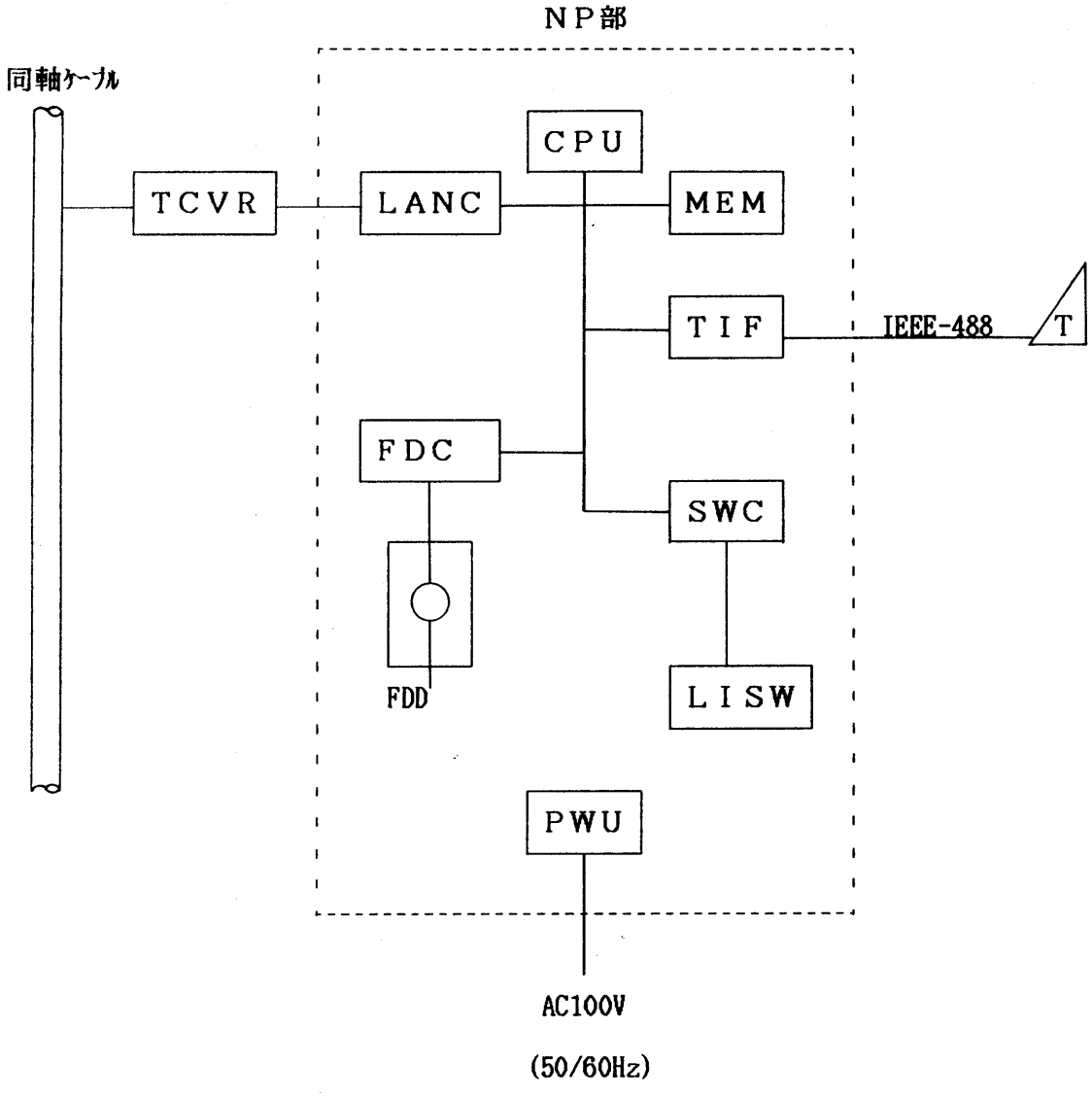
画面構成	80字×25行　40字×25行　20字×25行 80字×20行　40字×20行　20字×20行 英数、カナ　漢字
画面	12ｲﾝﾁ　ｸﾞﾘｰﾝ又はｶﾗｰ
文字エリア	8×19ﾄﾞｯﾄ（英、数、ｶﾅ）、16×19ﾄﾞｯﾄ（漢字）
グラフィック機能	640×475ﾄﾞｯﾄ
カラー機能	黒、青、赤、紫、緑、水色、黄、白（８色）
カーソル	点滅表示
ライトペン（オプション）	ｲﾝﾀﾌｪｰｽ内蔵。ｽｲｯﾁ信号、光検出信号分離。 40×200のｴﾘｱ判定機能。

＜フロッピディスク＞

ディスクドライブ	両面倍密度FDD（8ｲﾝﾁ）
平均アクセスタイム	216msec
内分け	
・セトリング時間	15msec
・平均シーク時間	3msec×38 ﾄﾗｯｸ＝114msec
・平均回転待ち時間	83msec
・転送時間	16μsec/BYTE

記憶容量	1MBYTE/台
＜ハードディスク＞	
ハードディスクドライブ	５ 1/4ｲﾝﾁ磁気ﾃｨｽｸ
平均アクセスタイム	75msec
・セトリングタイム	15msec
・平均回転待ち時間	8.3msec
・転送時間	3.7μsec/BYTE
記憶容量	7.3MBYTE/台（実効）
接続台数	1台
＜キーボード＞	
方式	
エンコーダ内蔵スキャン	
ＫＥＹ	
JIS配列準拠、ファンクションキー、編集機能及びテンキー付、打鍵時クリック音発生、	
スロープスカルプチャー型	
インタフェース	
シリアルインタフェース	
＜プリンタ＞	
印字構成	
80，40字/行	
文字構成	
12×24，又は24×24ドットマトリックス	
ハードコピー可能	
印字ヘッド	
24ピン　ドットインパクト型	

表５．１　ＤＰハードウェア諸元



ＣＰＵ	：中央処理装置	ＦＤＣ	：フロッピーディスク制御部
ＭＥＭ	：メモリ	ＳＷＣ	：スイッチ／ランプ制御部
ＬＡＮＣ	：ＬＡＮ送受信部	ＬＩＳＷ	：操作部
ＴＩＦ	：端末インタフェース部	ＰＷＵ	：電源部

図５．４　ＮＰ部の構成

＜ＣＰＵ＞

使用ＣＰＵ	8086
ｃｌｏｃｋ周波数	5MHz

＜メインメモリ＞

使用素子	64KBITS/chip　ﾀﾞｲﾅﾐｯｸRAM
容量	512KBYTE 実装

＜端末インタフェース部＞

インタフェース	IEEE488　std. 488-1978に準拠

＜フロッピーディスク＞

ＤＰと同様

＜操作部＞

前面スイッチ	Power ON/OFF用ﾛｯｸｽｲｯﾁ
	Power Indicator (Green)
	Alarm 表示　(Red)
保守用スイッチ	Reset
	Test mode
	Mem Dump
	Select
	System error 表示
	ROM/RAM error 表示
	LAN error 表示
	ON-Line 表示
	Status 表示

＜ＬＡＮ送受信部＞

通信速度	10Mﾋﾞｯﾄ/S
伝送方式	ﾍﾞｰｽﾊﾞﾝﾄﾞ伝送

伝送制御方式	CSMA/CD
誤まり制御	32ビットCRCチェック
最大距離	500m/単位網
最大ステーション数	100個/単位網

表５．２　ＮＰハードウェア諸元

次に、ＤＰ部およびＮＰ部の装置外観をそれぞれ図５．５、図５．６に示す。

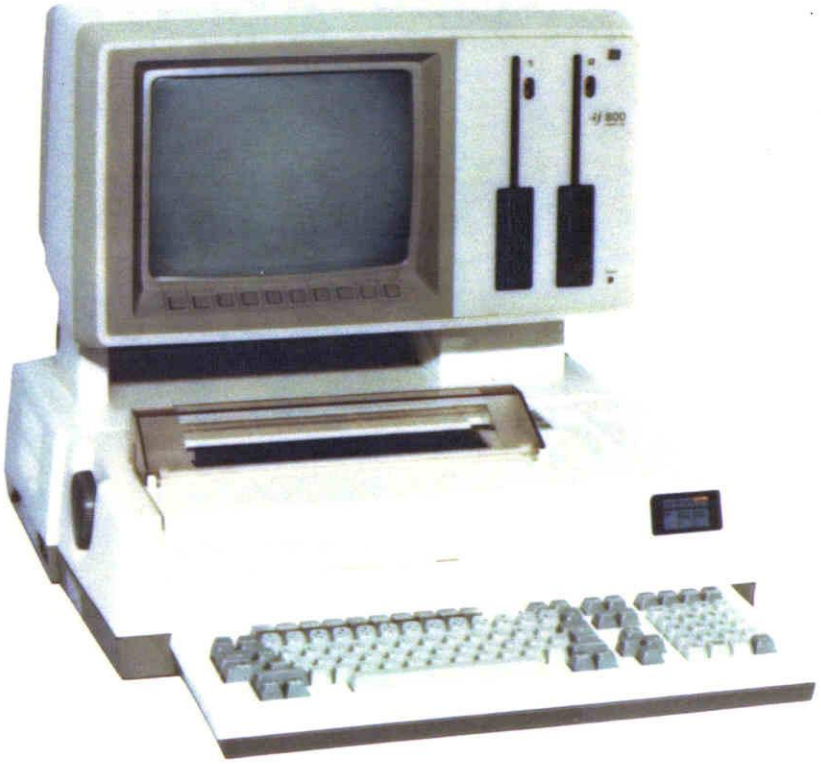


図５．５　ＤＰ部装置外観

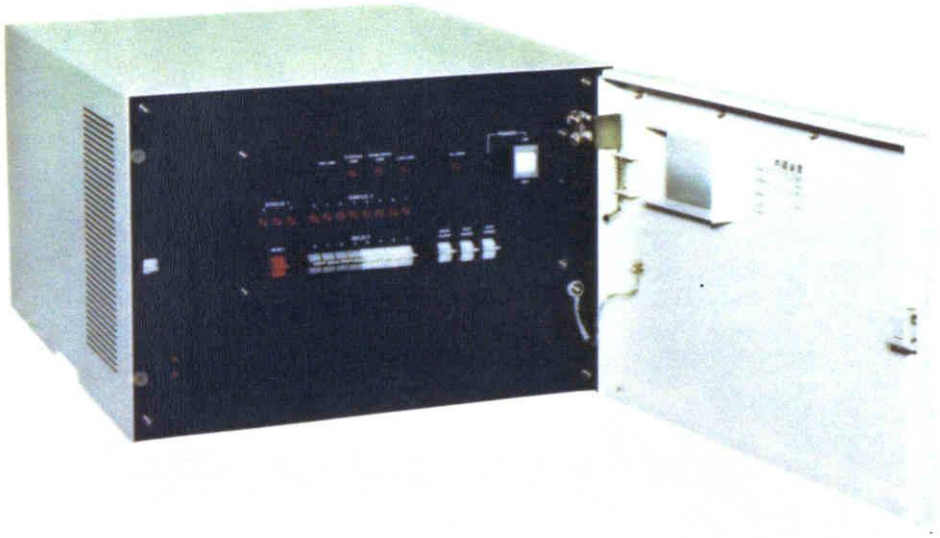


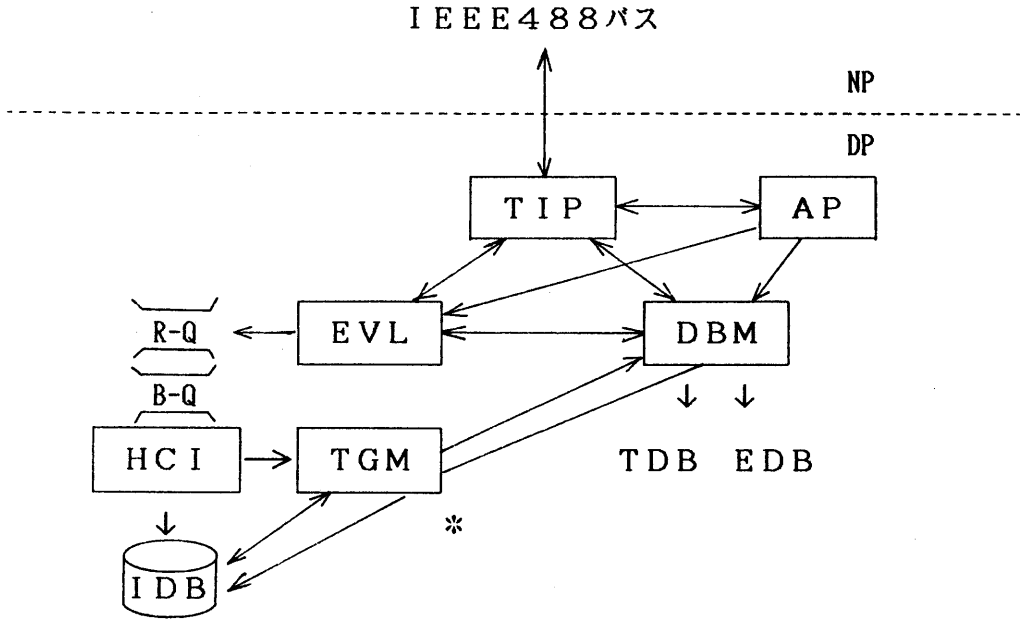
図５．６　ＮＰ部装置外観

５．２．２　ソフトウェア構成

ＳＤ^{3}－ＭＧの各ノードにおけるソフトウェアの概要をＤＰ、ＮＰのそれぞれについて以下に示す。

〔１〕ＤＰ内ソフトウェアの構成および機能概要

図５．７にＤＰ内ソフトウェアモジュールの構成を示す。なお現システムにおいてはＤＰ内Ｏ．ＳにＣＰ／Ｍを使用している。したがってＤＰにおいてはマルチプロセシングの概念はなく、各モジュールはすべて逐次的に実行されることになる。



AP　：Application Program　　TDB　：Temporary DataBase

EVL：Evaluator　　TGM　：Trigger Mechanism

HCI：Horm Clause Interpreter　　R-Q　：Ready Queue

DBM：DataBase Manager　　B-Q　：Blocked Queue

IDB：Intensional DataBase　　*updateのみ

EDB：Extensional DataBase

図５．７　ＤＰ内ソフトウェア構成

各モジュールの機能を以下に概説する。

① ＡＰ

出発地、目的地に関するエンドユーザからの入力が１つのノード(イニシエータノード)に対してなされる。イニシエータノードのＡＰは、これを受け付けそれらを分散データベースの検索を介して識別番号（ＩＤ）に変換する。次に自ノードを含む全ノードに対し径路導出の指示を出す。

また各ノードから返されたいくつかの径路の中から適切なものを選択する（例えば最も評価値の高いもの）。

最後に得られた径路情報から略地図を生成し画面に表示する。

なお各ノードは、与えられた出発地、目的地に対しその径路を探索し結果をイニシエータノードのＡＰに返すことになるが、全解探索以外ではＡＰは一定個数以上の径路情報を受け取ると全ノードに対し停止メッセージを送信する。したがって各ノードは、この停止メッセージを受け取るか、導出し得るすべての径路を導出し尽した場合、処理を終了する。

② ＴＩＰ

ＩＥＥＥ４８８バスの制御を行い、ＮＰより送信される要求の受信、ＮＰへの要求の送信を行う。またＮＰから受信された要求を、演繹検索に関するものはＥＶＬに、単純検索に関するものはＤＢＭにそれぞれ振り分け、対応モジュールに起動をかける。

③ ＥＶＬ

このモジュールは、略地図生成という特定問題向きに手続きが記述されており、イニシエータノードからの指示メッセージや他ノードからの協調メッセージにより、径路の探索と導出を行う。ただしここで得られる径路とは、自ノードで導き得る部分径路である。（部分径路とは、出発地点から目標地点に至る径路の一部分もしくは全径路のことである。）

図５．８ではノードα、β、γが協調して出発地点から目標地点の径路を導く様子を例示している。ノードα、β、γのいずれも、単独では出発地点から目標地点までの全径路を導出できない。たとえばノードαのＥＶＬは出発地点から地点Ｌ1 までの部分径路を導出できるがＬ1 以降の径路については導出できない。そこでαはＬ1 以降の導出を依頼すべく協調メッセージをブロードカストする。ノードβ、γはαからの依頼を受けるが、Ｌ1 以降の導出を請け負うことができるノードはβだけである。そのため、ノードβはノードαから協調メッセージによりＬ1 以降の導出を請け負い、βのＥＶＬは目標地点に至ると考えられる部分径路Ｌ1 ～Ｌ2 を導出する。最後に、Ｌ2 から目標地点に至る径路はノードγにより導出される。なお、協調プロトコルについては第４．６．３節（２）項で既に述べた。

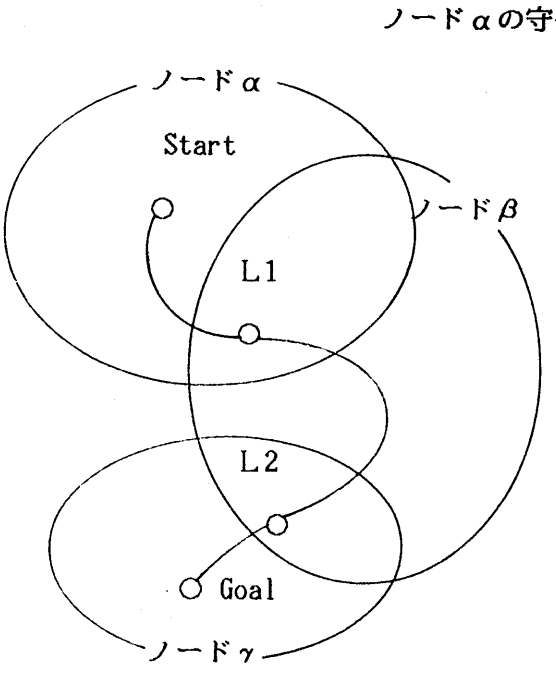


図５．８　径路導出例

ＥＶＬは、まずR-Q中にある中間結果のうち次に展開する対象となるもの（例えば最も評価値の高い中間結果）を取り出す。次に、展開のためのストラテジ（対象中間結果のＩＤと展開の深さ）を設定し、これをＤＢＭを介してＴＤＢに書き込む。次にＨＣＩを起動させることにより、展開後の中間結果を取り出しこれをR-Qに追加する。

４　ＨＣＩ

ＨＣＩはＥＶＬからの起動要求を受け付け、ＩＤＢを参照しながら、演繹推論を実行する。もし推論の途中でデータベース参照が必要となった場合は、ＤＢＭに対するキュアリを生成し、検索結果を受取る。また、推論の過程で使用したルールにトリガが表示されていればＴＧＭの起動を行う。最後に求める中間結果が得られたか、あるいは途中でＦａｉｌとなった場合、これをＥＶＬに返す。

５　ＤＢＭ

ＥＤＢおよびＴＤＢに対する単純検索、および更新を実行する。ＤＢＭに対するアクセ

スコマンドは、通常のデータベースと同様、Retrieve、Insert Deleteおよびデータ定義に伴なう各種コマンドを用意している。既存のデータベース管理システムと異なる点は、ユーザがデータ定義時にデータの細部構造（内部スキーマ）まで指定できるよういくつかの指示コマンドを設け、検索効率の向上を設っている点にある。

なお、ＤＢＭはリレーショナルモデルに準拠した汎用的なモジュールとしてインプリメントされており、ＴＤＢ、ＥＤＢに対する処理上の区別は行っていない。ＳＤ^3－ＭＧにおいては、ＤＰ内各モジュールからＤＢＭに様々な要求が出される（ＴＩＰからはエンドユーザが入力した出発点、到達点に対する検索要求、ＥＶＬからは解に至った中間結果のトレース要求、ＴＧＭからは中間結果のＴＤＢへの書き込み要求、ＨＣＩからはＥＤＢおよびＴＤＢに対する検索要求等である）。

⑥ ＴＧＭ

ＨＣＩによって起動されたルールが成功し、かつそのときの変数の値を問題解決の完了まで確保しておきたいときこのモジュールが起動される。ＴＧＭは、ＤＢＭに対し、ＴＤＢへの挿入要求を生成し、そのときの変数値のＴＤＢへの書き込み指示を行う。なおトリガ起動のための機構の詳細はＩＤＢの説明とあわせて次項で行う。

次に、ＩＤＢ、ＥＤＢおよびＴＤＢには、特定問題向きの情報（ＳＤ^3－ＭＧにおいては地図データおよび径路導出に要する各種のルール等）が格納されることになる。以下にその詳細を述べ、あわせてＳＤ^{3}におけるＩＤＢ、ＥＤＢ等の役割を論ずることとする。

⑦ ＥＤＢ

ＳＤ^3－ＭＧにおけるＥＤＢは、特定地域の地図情報を格納するデータベースとなっており次の３つのテーブルから成る。

・ＬＮＡＭＥ

文字列で表された地点名とそれに対する識別番号（ＩＤ）との対応表であり、ユーザ入力の地点名をそのＩＤに変換するために用いられる。ＬＮＡＭＥテーブルの構成例を図５．９に示す。

地点名	地点識別番号
田町駅	００３
三田ビル	０１１
・	・
・	・

図５．９　ＬＮＡＭＥテーブル構成例

・ＣＯＮＣ

地点ＩＤとその隣接地点ＩＤおよびそれらを結合する道路の識別番号とその間の距離等によって構成され、径路導出の過程で参照される。図５．１０にテーブル構成例を示す。

地点ＩＤ	隣接地点ＩＤ	道路ＩＤ	距離	境界フラグ
００１	００４	２１２	５	
００１	００５	３３４	１	１
００４	００１	２１２	５	
００４	０１１	１０１	３	１

図５．１０　ＣＯＮＣテーブル構成例

・ＥＬＯＣ

地点ＩＤとその地点のｘ座標ｙ座標が与えられる。なお座標値は、特定地点を原点として与えたものである。図５．１１にテーブル構成例を示す。

地点ＩＤ	ｘ座標	ｙ座標
０１２	１４６	２５２
０３３	３６	１９８

図５．１１　ＥＬＯＣテーブル構成例

⑧ ＩＤＢ

ＩＤＢの説明に先だち、以下の用語の定義を行っておく。

・径路　：　複数の地点およびそれらを結ぶ道路のリストである。

・仮説　：　出発地点から中間地点までの最適径路の候補となる径路のことである。

・到達地点　：　ある仮説Ｈは出発地点から（目標地点に至る）中間地点Ｌに至る径路を表すとする。このときＬを仮説Ｈの到達地点という。

・評価値　：　仮説Ｈの評価値とは出発地点からＬに到るまでに要した距離とＬから目標地点までの推定距離を加えた値である。

・親仮説、子仮説　：　Ｌpから目標地点へ向けて、Ｌpと隣接する地点Ｌcへと径路を延ばすと、出発地点からＬcに至る径路を表す新しい仮説が得られる。このときHpを親仮説、Hcを子仮説という。

なお、同一の親からいくつかの子仮説を導出する場合もある。

・境界フラグ　：　ある地点について、その地点がノード上で達成域にあるか否かを示すフラグである。「地点Ａがノードの達成域にある」とは、地点Ａ自身およびその隣接する地点がすべてノードＮのデータベースに格納されているということを表している。また、ＳＤ^3－ＭＧの扱えるすべての地点は、ＳＤ^3－ＭＧのいずれかのノードのＥＤＢにおいて達成域にあるように構成されている。つまりＳＤ^3－ＭＧでは適当なノードを選択すれば、ある地点に隣接するすべての地点がそのノード内だけで導出できるようにデータベースの設計がなされている。

既に述べたように、このように設計されたＳＤ^3－ＭＧは全編成であるという。

以降ではＩＤＢに格納されるルールおよび述語について概説する。

なお到達地点、道路、および仮説はすべて識別子によって表現されており、ここではそれぞれを LocationＩＤ、StreetＩＤ、仮説ＩＤと呼ぶ。

ＩＤＢには図５．１２に示すルールが格納される。

```
Rule.1  HLOC (p.h.q.d.f) ← EHLOC(p.h.q.d.f)
Rule.2  HLOC (p.h.q.d.f) ← HLOC(o.p.l.r.g) ∧ CONC(l.q.s.d.f)
                           ∧ UNUSE-CHECK(p.h.q) ∧ GDIS(q.x)
                           ∧ GDIS(l.y) ∧ lt(x.y)
Rule.3  VAL(h.t.s.f) ← EVAL(h.t.s.f)
Rule.4  VAL(h.t.s.f) ← EPCH(p.h) ∧ VAL(p.q.r.g)
                       ∧ HLOC(o.p.l.e.i) ∧ HLOC(p.h.m.d.f)
                       ∧ A D D(q.d.t) ∧ GDIS(m.s)
Rule.5  GDIS(q.x) ← ELOC(q.x1.y1) ∧ EGOAL(x2.y2)
                    ∧ ABS(x1.x2.z1) ∧ ABS(y1.y2.z2)
                    ∧ ADD(z1.z2.x)
```

図５．１２　ＩＤＢ内ルール

ルール内の各述語は、以下の様に定義される；

ＨＬＯＣ(p.h.q.d.f)

p:親仮説ＩＤ。

h:子仮説ＩＤ。

q:子仮説 h のLocationＩＤ。

d:親仮説 p のLocationＩＤから子仮説 h のLocationＩＤまでの距離。

f: q が達成域にある場合は０、ない場合は１が入れられる（境界フラグ）。

ＥＨＬＯＣ(p.h.q.d.f)

p.h.q.d.f の意味はＨＬＯＣと同様であり、ＴＤＢ内にリレーションとして外延的に定義される。

ＶＡＬ(h.t.s.f)

h:仮説ＩＤ。

t:出発地点から仮説 h の到達地点までに要した距離。

s:仮説 h の到達地点から目標地点までの推定距離（直線距離）。

f:仮説 h の到達地点に対する境界フラグ。

ＥＶＡＬ(h.t.s.f)

h.t.s.t.f の意味はＶＡＬと同様であり、ＴＤＢ内のリレーションとして外延的に定義される。

ＥＰＣＨ(p.h)

p:仮説ＩＤ。

h:仮説 p の子仮説ＩＤ。

但しこの述語はＴＤＢ内のリレーションとして外延的に定義される。

ＣＯＮＣ(l.q.s.d.f)

ＥＤＢ内のリレーションとして外延的に定義される。l、q、s、d、f の意味は前述した。

ＵＮＵＳＥ－ＣＨＥＣＫ(p.h.q)

p:親仮説ＩＤ。

h:子仮説ＩＤ。

q:子仮説 h のLocationＩＤ。

この述語は同一の親仮説 p を持つ子仮説 h' のLocationＩＤが仮説 h の Location ＩＤ q と一致しないことをチェックするためにある組み込み述語である。

ＥＬＯＣ(q.x1.y1)

この述語はＥＤＢ内のリレーションとして外延的に定義される。q、x1、y1 は前述のようにそれぞれ地点ＩＤ、その地点のＸ座標、Ｙ座標を表す。

ＥＧＯＡＬ(x2.y2)

x2:目標地点のＸ座標。

y2:目標地点のＹ座標。

この述語は問題が与えられたときＡＰがＬＮＡＭＥリレーションおよびＥＬＯＣリレーションを検索して求める。内容はＴＤＢに格納される。

ＧＤＩＳ(q.x)

q:LocationＩＤ。

x:地点 q から目標地点までの推定距離。

但し (x1 y1) と (x2 y2) 間の推定距離は、ｄ＝|x1-x2|＋|y1-y2|により定義される。

ＡＢＳ (X1,X2,Z): X1 と X2 の差の絶対値を Z で表す組み込み述語。

ＡＤＤ (X1,X2,Z): X1 と X2 との和を Z で表す組み込み述語。

ｌｔ (X,Y)　　: X<Y ならば真、X≧Y ならば偽となる組み込み述語。

図５．１２における各ルールは、言語ｐｒｏｌｏｇと同様の動作を実行する。云いかえればＨＣＩに対しＧｏａｌ節を与えると、後向き推論によりまずヘッド部のマッチングが、ついでボデイ部のマッチングがとられることになる。ＨＣＩはどれを引数として与えられても動作可能なように汎用的に構成されているが、以下ではシステムの動作概要を説明する都合上、ＳＤ^3－ＭＧにおけるルールの使われ方、動作についてのみ述べることとする。

・Rule.1

仮説ＩＤ（ｈ）を確定された引数として起動されるルールであり、既にＴＧＭによりＴＤＢへ登録されているＥＨＬＯＣを参照し、親仮説ＩＤ（ｐ）、仮説ｈの到達地点ｑ、親仮説ｐの到達地点からｑまでの道のりｄおよびｑに対する境界フラグｆを返す。

なお Rule.2 や Rule.4 でＨＬＯＣ(ｏ,ｐ,･･･) を評価する場合に Rule.1 が起動される。

・Rule.2

仮説ＩＤ（ｈ）およびその親仮説ＩＤ（ｐ）とを確定された引数として起動されるルールである。

ただし、仮説ｈの内容（到達地点、評価値、･･･）はこのルールの起動により、導出されるわけであるから、Rule.2 の起動時にはその仮説ＩＤ（ｈ）が与えられているだけである（このルールが失敗すれば、ｈの仮説ＩＤ自体もＴＤＢより削除される）。Rule.2 の起動により各述語が起動され、それらがすべて成功すれば、仮説ｈの到達地点ｑ、親仮説ｐの到達地点からｑまでの道のりｄ、および仮説ｈに対する境界フラグｆが求まる。ただし、到達地点ｑは、親仮説ｐから既に導出されている仮説ｈ' の到達地点ｑ' と比較して、ｑ≠ｑ' となるように決定される。これは述語 UNUSE-CHECK で調べることになる。

次に各述語の評価時の動作について詳細を説明する。

まずＨＬＯＣの起動で親仮説ｐの到達地点ｌが求まる。次に、ＣＯＮＣの起動でｌに隣接する地点ｑ、ｌとｑと通る道路ｓ、ｌとｑの道のりｄおよびｑの境界フラグｆが求まる。UNUSE-CHECK の起動では、先に述べたようにｑが既出であるか否かを調べる。

ＧＤＩＳ Λ ＧＤＩＳ Λ ｌｔの一連の起動により、親仮説の到達地点ｌよりも子仮説の到達地点ｑの方が目標地点との推定距離が短いか否かが調べられる。

なおRule.4 でＨＬＯＣ（ｐ,ｈ,･･･）を評価する際にも Rule.2 は起動される。

・Rule.3

仮説ｈを確定された引数として起動されるルールであり、既に（ＴＧＭにより）ＴＤＢへ登録されているＥＶＡＬを参照し、出発地点から仮説ｈの到達地点までの道のりｔ、仮説ｈの到達地点から目標地点までの推定距離ｓおよび境界フラグｆを返す。

Rule.4 でＶＡＬ（ｐ,･･･）を評価する場合に Rule.3 は起動される。

・Rule.4

仮説ＩＤ（ｈ）を確定された引数として起動されるルールである。ただし、仮説ｈの内容（到達地点、評価値、･･･）はこのルールの起動により、導出されるわけであるから、Rule.4 の起動時にはその仮説ＩＤが与えられているだけである。Rule.4 の起動により各述語が起動されそれらがすべて成功すれば、出発地点から到達地点までの道のりｔ、到達地点から目標地点までの距離ｓおよび境界フラグｆが求められる。

次に各述語評価時の動作について詳細を示す。

まず、ＥＰＣＨの起動により、仮説ｈの親仮説ｐが求まる。次に、ＶＡＬの起動により、出発地点から親仮説ｐの到達地点までの道のりｑが求まる。一つめのＨＬＯＣ（ｏ，ｐ，･･･）の起動により二つめのＨＬＯＣ（ｐ，ｈ，･･･）起動の際に評価されるＨＬＯＣ（ｏ，ｐ，･･･）の値を保証する。ＨＬＯＣ（ｐ，ｈ，･･･）の起動により、仮説ｈの到達地点ｍおよび仮説ｐの到達地点からｍまでの道のりｄが求まる。ただし、ｍが自ノードの達成域になければ、フラグｆが設定される。組み込み述語ＡＤＤにより「出発地点～親仮説ｐの到達地点までの道のり」ｑと「親仮説ｐの到達地点～仮説ｈの到達地点ｍまでの道のり」ｄとを加えた値ｔが求まる。最後にＧＤＩＳにより、「仮説ｈの到達地点ｍから目標地点までの推定距離」ｓが求まる。

・Rule.5

LocationＩＤ（ｑ）を確定された引数として起動され、「ｑから目標地点までの推定距離」ｘを求めるルールである。

まず、ＥＬＯＣにより地点ｑの座標（x1, y1）を導出する。次に、ＥＧＯＡＬにより目標地点の座標（x2, y2）を検索する。最後に、ＡＢＳ ∧ ＡＢＳ ∧ ＡＤＤ により「ｑ～目標地点の推定距離」ｘが評価される。

なおＩＤＢには、次のようなトリガ定義も含まれている。

TG.1　ＥＨＬＯＣ　insert on Rule.2

TG.2　ＥＶＡＬ　insert on Rule.4

各トリガの機能は以下であり、その動作の実行はＴＧＭによりなされることになる。

・TG.1

Rule.2 の実行後ＨＬＯＣ（ｐ,ｈ,ｑ,ｄ,ｆ）の各値が確定したら、同じ値を持つレコード（ｐ,ｈ,ｑ,ｄ,ｆ）をＥＨＬＯＣに追加する（ＴＤＢに書き込む）ことを表す。

・TG.2

Rule.4 の実行後ＶＡＬ（ｈ,ｔ,ｓ,ｆ）の各値が確定したら、同じ値を持つレコード（ｈ,ｔ,ｓ,ｆ）をＥＶＡＬに追加することを表す。

⑨ ＴＤＢ

仮説を展開してゆく上で一時的にＥＶＬが生成するリレーションより成るデータベースであり、プレディケイトＥＨＬＯＣ、ＥＶＡＬ、ＥＰＣＨおよびＥＧＯＡＬの起動によりこのＴＤＢが参照される。ＥＨＬＯＣとＥＶＡＬは前述のＴＧＭによりＴＤＢに登録される。ＥＰＣＨは、ＨＣＩに対して仮説の展開を指示する際に、ＥＶＬによりＴＤＢに登録される。ＥＧＯＡＬは径路探索を開始する前に、ＡＰより各ノードへ送信され、各ノードはＥＧＯＡＬ値を受信すると同時にＴＤＢへ登録する。

［２］ＮＰ内ソフトウェアの構成および機能概要

図５．１３にＮＰ内ソフトウェアモジュールの構成を示す。なお図において Kernel として示した部分は、リアルタイム処理向きの専用オペレーティングシステムである。

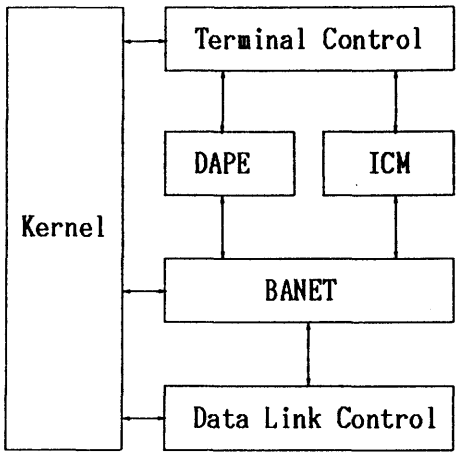
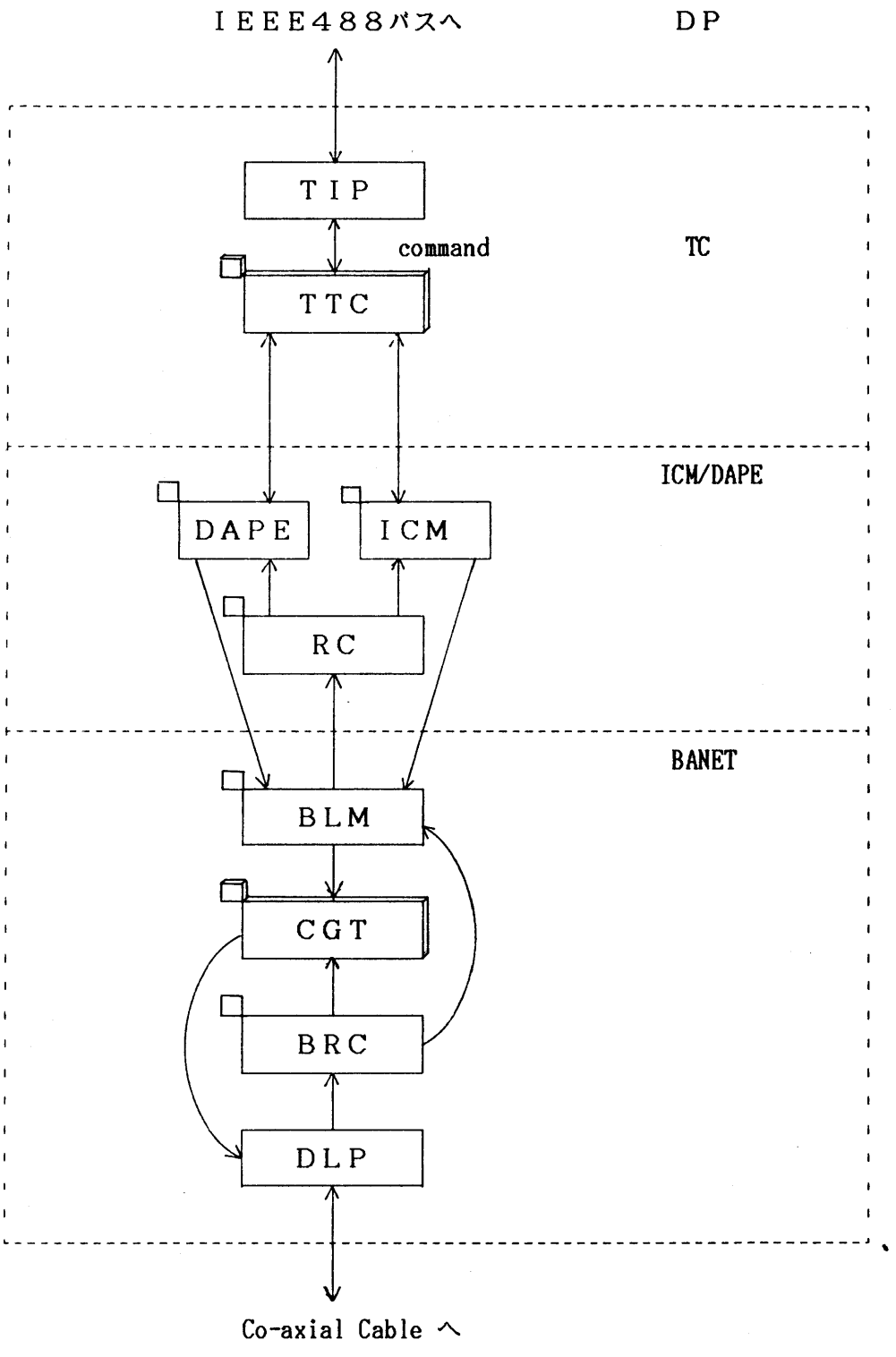


図５．１３　ＳＤ^{3}ノードＮＰ内ソフトウェア構成

次に、図５．１４は、Kernel 部を除くＮＰ内ソフトウェアモジュールの構成をより詳細に示したものである。


IEEE488バスへ
DP
TIP
command
TC
TTC
ICM/DAPE
DAPE
ICM
RC
BANET
BLM
CGT
BRC
DLP
Co-axial Cable へ

TC　: Terminal Control

DAPE : Data Base Access Protocol Executor

ICM　: Internode Cooperation Module

BANET: Broodcast Architecture Network

TIP　: Terminal Interface Program

TTC　: Terminal Task Control

RC　: Receive Control

BLM　: Broodcast Level Manager

CGT　: Communication Group Task control

BRC　: Banet Receive Control

DLP　: Data Link Program

図５．１４　Kernel を除くＮＰのソフトウェア構成

図５．１４における各モジュールの機能概要は以下のとおりである。

① ＴＩＰ

ＩＥＥＥ４８８バスを制御することにより、他ノードから受信されたメッセージのＤＰへの送信および他ノードへの要求メッセージのＤＰからの受信を行う。

② ＴＴＣ

ＤＰから受信された要求を分散データベースに関するものと分散型推論に関するものとに振り分け、それぞれＤＡＰＥおよびＩＣＭに渡す。また他ノードから受信されたメッセージをＴＩＰを介して、ＤＰに引き渡す。マルチタスキングにより実行される。

③ ＤＡＰＥ

第３．５．４節で述べたデータベースアクセスプロトコルを実行する。但し、ＳＤ^{3}－ＭＧにおいては、分散型推論実行中のデータベースまたはルールベースへのｕｐｄａｔｅは認めていない。

④ ＩＣＭ

第５．６．３節で述べた分散型推論のためのプロトコルを実行する。

⑤ ＲＣ

ＢＡＮＥＴからの非同期的な受信処理を実行する。また受信されたメッセージがＩＣＭへ引き継ぐものか、ＤＡＰＥへ渡すかの振り分けも行う。

⑥ ＢＡＮＥＴ

第２．４．３節で述べたローカルネットワークのプロトコルを実行する。ＢＡＮＥＴを構成するモジュールとその動作概要を以下に示す；

ＢＬＭ：通信グループの形成と消滅の指示、ネームサービス、ＲＣへのメッセージの転送等、ＢＡＮＥＴの持つ基本的な機能を実行する。

ＣＧＴ：通信グループ毎にタスクとして生成される。ＣＧＴモジュールは、ＢＬＭからの指示に基づき、通信グループの形成と消滅、データ転送等を実行する。

ＢＲＣ：データリンクレベルからの非同期的なデータの受信を行い、ＢＬＭあるいはＣＧＴに引き渡す。

ＤＬＰ：10Mb/sec の Ethernet 仕様データリンクレベルの送受信処理を実行する。このモジュールはその大部分がＬＳＩにより実現されている。

５．２．３ 分散型問題解決の動作概要

本節ではＳＤ^3－ＭＧにおけるユーザ要求の入力から略地図出力に至るシステムの動作概要を示すことにより、ＳＤ^{3}における問題解決の方式を論ずることとする。

［１］実行のフェーズ分けと動作概要

システムの実行過程は、以下の３つのフェーズに分割して論ずることができる：

ａ．問題入力

イニシエータノードが中心となり、以下の３つに分けられる。

① ユーザからの入力（出発地点、目標地点）を受けつける。

② 各ノードのＥＤＢを検索して、演繹を開始する上で必要な初期値を得る。

③ 各ノードに対して演繹検索の開始を依頼する。

b．演繹検索

各ノードは、適切と思われる仮説(径路)をいくつか導出しイニシエータノードへ渡す。

c．略地図導出

大きく以下の３つの処理を行う。

① イニシエータノードは他ノードより渡された仮説（径路）の内、最適と思われるものを一つ選ぶ。

② 他ノードに対し、仮説の内容である地点名と道路名の並びよりなる具体的な径路のトレースを要求する。

③ 出発地点～目標地点までの径路を表示する。

［２］各フェーズの詳細

a．問題入力

イニシエータノードが中心となり、ユーザや他ノードとやりとりを行い、演繹検索開始までに必要な前処理を行う。以下にその手順を述べる。

① ユーザはイニシエータノードに、出発地点と目標地点の地点名を与える。

② 地点名を受け取ったイニシエータノードは他ノードに対して、出発地点と目標地点のLocationＩＤ(以降 Start、Goalとする)およびＥＧＯＡＬの値を要求する。

③ Start、Goal のいずれか一方が得られない場合は、地点名がシステムで処理できる範囲を越えているものとして、メッセージを出して処理を終了する。

④ Start、Goal が等しければ、推論を実行する必要もないのでメッセージを出して処理を終了する。

⑤ ＥＧＯＡＬの値および Start → Goal という問題の実行依頼を他ノードに対してブロードカストする。

b．演繹検索

演繹検索は、協調プロトコルに基づき各ノード間で協調、競合メッセージを通信する処理と、個々のノード内での処理とに分けて考えることができる。

まず、各ノード間での処理について説明する。

そのためにＳＤ^3－ＭＧにおける協調および競合メッセージの意味を再度明確にしておく；

協調メッセージとは、あるノードＮｉで導出した仮説ｈのそれ以降の展開を他ノードＮｊに依頼するために、Ｎｉが他ノードに対しブロードカストするメッセージである。ノードＮｉが協調メッセージをブロードカストする際の契機は次の２つの条件のいずれかが満足された時である。

① Ｎｉで導出した仮説ｈはＮｉの達成域にない。つまり導出した径路がノードＮｉにおける守備範囲の境界部にまで到達した場合。

② Ｎｉでｋ回、仮説を展開する毎に、２番目に評価値の高い仮説について、その仮説以降の仮説の展開を他ノードに依頼する場合。

①の協調メッセージは、自ノードＮｉにおいては、仮説ｈ以降の仮説の正当な展開が期待できない場合である。つまりｈの到達地点に隣接する地点すべてをＮｉはその守備範囲とはしていないために、Ｎｉで導出できるｈ以降の仮説は限定されている。したがって、自ノードを除く全ノードに対して協調メッセージをブロードカストする必要がある。なおＳＤ^3－ＭＧにおいては全編成となるようデータベースが設計されているため、全展開可能なノードはシステム内に必ず存在する。

②の協調メッセージは、システム全体での並列処理の効率の向上を目的とするものであり、具体的には以下の事項につきその効果を期待することになる。

- 遊休ノードの数をできるだけ減らすこと。
- あるノードＮｉにおいて２番目の評価値を持つ仮説ｈの展開を他ノードに依頼することで、もし他ノードＮｊではその仮説ｈが１番目の評価値となるならば、ｈ以降の仮説の展開が、ノードＮｊにおいては最優先で実行される。つまり、評価値のよい仮説をできるだけ優先的に展開できるようになる。

こうして協調メッセージをブロードカストすると、ブロードカストされた仮説を達成域に持つノードはすべてその仮説を請け負い、展開を実行する。このため同一の仮説を

複数のノードで請け負った場合には、同一の仮説の展開が重複して行われることとなる。このような重複した展開を防ぐために、または既に重複して展開が実行されている場合にはできるだけ早く終結させ、１つのノードを残しその他のノードでの重複した展開を放棄させるために、競合メッセージがブロードカストされる。以下では競合メッセージ送出の契機および重複している展開の終結時の動作について概説する。

① まず、仮説ｈの協調メッセージがブロードカストされる。

② 仮説ｈを達成域に持つノードはすべてその仮説を請け負う。

③ ｈを請け負ったノードは競合メッセージをブロードカストする。ノードＮｉで仮説ｈを請け負った時に、Ｎｉがブロードカストする競合メッセージの内容は以下の通りである。

- 請け負った仮説の仮説ＩＤ（ｈ）。
- ｈを請け負ったノードの識別番号（Ｎｉ）。
- ノードＮｉにおいて、ｈが何番目の評価値を持つのかを示す値 qorder（Ｎｉ，ｈ）、（ノード内では最良評価値の仮説から順次展開されている）。

④ 競合メッセージを受信したノードＮｊは自身の中で仮説ｈを請け負っているか否かを調べる。もしｈを請け負っていればＮｊ自身での qorder（Ｎｊ，ｈ）値とブロードカストされた qorder（Ｎｉ，ｈ）とを比較し、ｈ以降の仮説の展開を実行（続行）するか停止するかを決定する。停止する場合は、ｈに関する自ノード内ＴＤＢの内容をすべて削除する。

出発地点から目標地点への径路の導出は、ソースノードより問題が協調メッセージとして他ノードへブロードカストされることで開始される。以後ソースノードより停止メッセージがブロードカストされるまで、適宜、協調メッセージおよび競合メッセージをブロードカストすることで、目標地点へ至る仮説を導出することができ、導出された仮説はソースノードへその評価値と共に渡される。ソースノードはｋ個の、目標地点へ至る仮説を受け取ると停止メッセージを他ノードへブロードカストする。

次に個々のノード内での演繹検索処理について説明する。今、出発地点 Start から中間地点Aにいたる径路は既に導出されていて、ノードＮｉにおいて仮説ｈａによって与えられているものとする。さらにｈａはＮｉの達成域にあるものとする。このときｈａの子仮説が導出される際の動作概要は次の通りである；

① ＥＨＬＯＣ（Hp，ｈａ，A，Da，false）およびＥＶＡＬ（ｈａ，Ta，Sa，false）が予めＴＤＢに格納されている。

② ＥＶＡＬはＴＤＢにＥＰＣＨ（ｈａ，hc）を追加し、ＨＣＩに対し
ゴール節 ← ＶＡＬ（hc，t，s，f）の実行を要求する。

③ ＨＣＩ内では、まず Rule.3 が起動されるが失敗し、引続き Rule.4 が起動される。

④ Rule.4 のＶＡＬ（ｈａ，q，r，f）の評価のために Rule.3 が起動されｑ＝Ta、ｒ＝Sa、ｆ＝false で成功する。

⑤ Rule.4 のＨＬＯＣ（hp，ｈａ，l，e，i）の評価のため、Rule.1 が起動されｌ＝a、ｅ＝Da、ｉ＝false で成功する。

⑥ Rule.4 のＨＬＯＣ（ha，hc，m，d，f）の評価のため、Rule.1 が起動されるが失敗し、Rule.2 が起動される。ｍ＝c、ｄ＝Dc、ｆ＝false で成功する。

⑦ Rule.2 の起動に成功したのでトリガが起動される。TG.1 によりＥＨＬＯＣにレコード（ｈａ，hc，Ｃ，Ｄ，false）が追加される。

⑧ Rule.4 のＡＤＤ（Ta，Dc，t）でｔ＝Tc が計算される。

⑨ Rule.4 のＧＤＩＳ（Mc，s）の評価のため、Rule.5 が起動されｓ＝Sc で成功する。

⑩ Rule.4 が成功したので、TG.2 が起動されＥＶＡＬにレコード（hc，Tc，Sc，false）が追加される。

⑪ ＨＣＩは、ゴール節 ← ＶＡＬ（hc，t，s，f）の結果としてＶＡＬ（hc，Tc，Sc，false）をＥＶＬに返す。

このような手順により、親仮説ｈａより子仮説ｈｃが導出される。なお手順１で、ＥＨＬＯＣ（hp，ｈａ,･･･）およびＥＶＡＬ（ｈａ,･･･）は予めＴＤＢに格納されているものとしているが、これは次の２つの要件による。

① ｈａ→ｈｃを導出したノードでｈｐ→ｈａも導出している場合、ｈｐ→ｈａを導出する過程で、手順７によりＥＨＬＯＣ（ｈｐ,ｈａ,･･･）が、手順10によりＥＶＡＬ（ｈａ,･･･）がＴＤＢに書き込まれる。

② 協調メッセージにより、他ノードまたはユーザインタフェース部より仮説ｈａの展開が依頼された時に、ｈａと共にＥＨＬＯＣ（ｈｐ，ｈａ,･･･）およびＥＶＡＬ（ｈａ,･･･）も渡される。

ｃ．略地図導出

演繹が終了した時点では最適仮説の候補がＫ個あるものとする。イニシエータノードは、このうち最も評価値のよい仮説h_{opt}を選び、その径路のトレース要求を他ノードにブロードカストする。

データベースを検索することで、一つの仮説につき、その仮説に対応する到達地点から親、子仮説に対応する到達地点への径路を導出できる。このため目標地点へ至る仮説より親仮説を順次たどることで、目標地点より出発地点へと径路を導出できる。

仮説h_{opt}を導出したノードは自ノードで導出した仮説がなくなるまで親仮説を順次トレースし、ＥＤＢを検索することで『径路』を導出する。トレースした親仮説が他ノードで導出されたものであればそのノードに対し『径路』を導出するように要求をだす。径路の導出要求を受け取ったノードは親仮説が自ノードで導出したものでなくなるまで親仮説をたどり径路を導出する、といった手順を繰り返す。

各ノードで導出された径路はイニシエータノードへと渡される。親仮説が初期値と一致すれば全径路が導出されたことになり、イニシエータノードは略地図を表示し、処理を終了する。

５．３　システムの現状

ＳＤ^3－ＭＧシステムは、現在、性能の詳細評価および改良のためのデータの収集を継続している段階である。

図５．１６～図５．２８は、ＳＤ^3－ＭＧによる出力および表示された途中結果の一例を示したものである。

処理時間は、生成中間結果数が１００以上ある相当複雑な経路探索で３～５分、２０以下の簡単なもので１０～３０秒を要しており、パーソナルコンピュータベースのプラン策定システムとしては、ほぼ満足のいく性能を得ることができた。

但し、現在のＳＤ^3－ＭＧシステムにおける出発点、目的点のエンドユーザによる入力は、簡易版のカナ漢字変換方式（音を入力して該当漢字を選び出す）を採用しており、ユーザにとっては余り便利な方式とはいえない。今後はメニュー選択方式、あるいは手書き文字認識のようなよりユーザフレンドリな方式に改良していきたいと考えている。

現在、ＳＤ^3－ＭＧにおける地図情報として試験的に格納されているファクトデータは、インデクス部分も含め総容量約１６０Ｋバイト、地点数約3,000ヶ所であり、東京都港区田町駅周辺の主要な建造物、ビル、企業および官庁等はすべて収録されたものとなっている。今後収録地点数、地域等を拡張することにより、略地図生成のコンサルテーションシステムとしても十分実用に供し得るものとすることを予定している。

一方ソフトウェアは、ＮＰ内モジュールおよびＤＰ内モジュールともにＣ言語で記述されており、ステートメント数は、表５．３のとおりである。

モジュール名	メモリ量	ステートメント数	ハードウェア
AP	18$^{K}_{byte}$	1.2^{K}	DP
EVL, TGM	12$^{K}_{byte}$	1^{K}	DP
HCI	9$^{K}_{byte}$	1^{K}	DP
DBM	34$^{K}_{byte}$	2^{K}	DP
共通ルーチン	10^{K}	1.5^{K}	DP
TC	3^{K}	1.6^{K} (但しアセンブラ)	NP
ICM/DAPE	10^{K}	12^{K}	NP
BANET	41^{K}	4.2^{K}	NP
DLP	2^{K}	1.5^{K} (但しアセンブラ)	NP

表5．3　SD^{3}－MG各モジュールの規模

なお、本システムの開発中に、米国SRI（スタンフォード研究所）より、パーソナルコンピュータベースのエキスパートシステムとしてSeRIES－PCが発表された。これは、単一のIBMPCもしくはXT上で動作可能となるもので、SD^{3}システムのような分散型の問題解決を目指したものではない。これはファクトデータは基本的には扱わず、またルール数も最大40程度と極めて制限されたものとなっている。したがって、収容する知識量、扱うことのできる問題の範囲等から判断し、SD^{3}システムの機能は、このシステムを大きく上回るものと考えている。

但し、SeRIES－PCにおいては、知識ベースエディタ等のエキスパートシステム記述のためのサポート用ツール、およびエンドユーザインタフェース用言語等が充実しており、SD^{3}システムにおいても本格的な実用化を想定するとこれら諸ソフトウェアの充実が必須の課題として残される。

特にＳＤ^{3}においては、特定問題向きに探索の戦略を決定するＥＶＬの部分の記述は、非専門家にとっては難しくなっているのが現状であり、この部分のユーザフレンドリな記述言語の開発が特に望まれる。

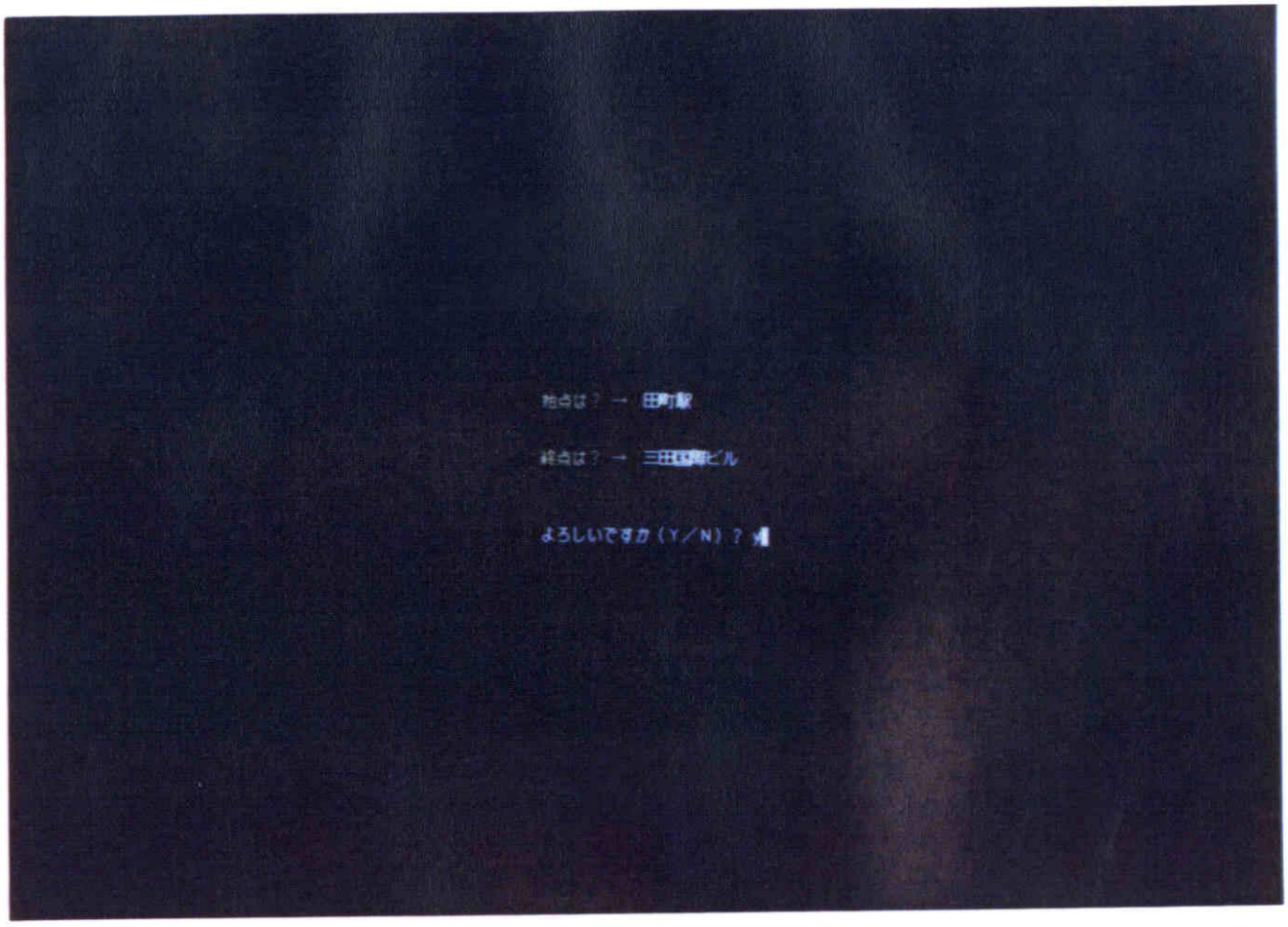


図５．１６　ユーザによる始点、終点の入力

図は、システムからの"始点は？"の問いに対しユーザが"田町駅"、ついで"終点は？"の問いに対し"三田国際ビル"と応じた時の画面を示している。次にシステムは、メッセージ"よろしいですか"を表示し、ユーザがＹ（Ｙｅｓ）を返すと、経路探索が開始されることになる。

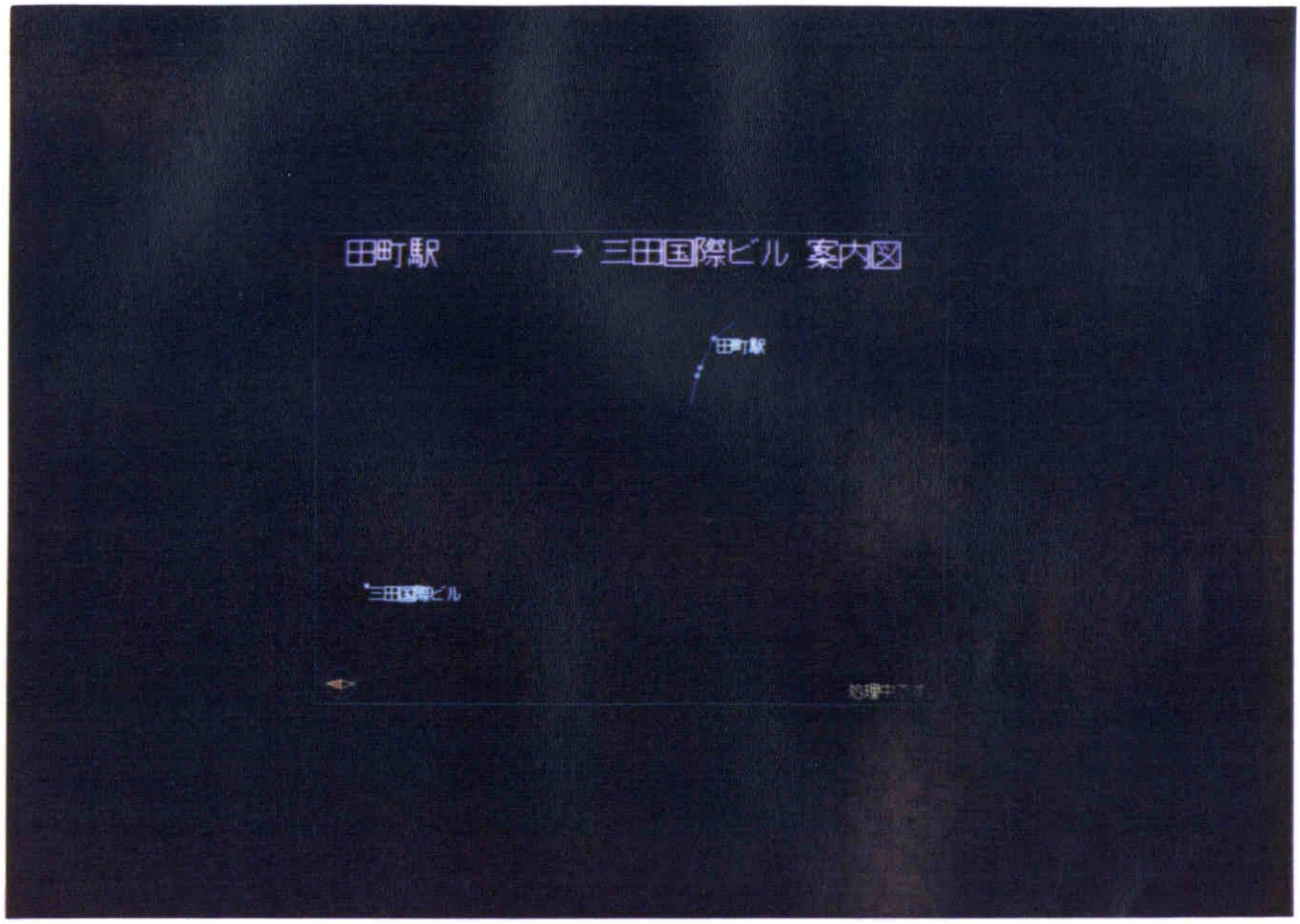


図５．１７　ＳＤ^{3}－ＭＧ経路探索の途中経過（１）

図は、田町駅を始点、三田国際ビルを終点として経路探索が開始された直後の画面をあらわす。なおこの画面は、処理の途中経過を説明する際のデモンストレーション用として組込まれたプログラムにより作成されるもので、通常の略地図導出においては、このような画面表示は行わない。

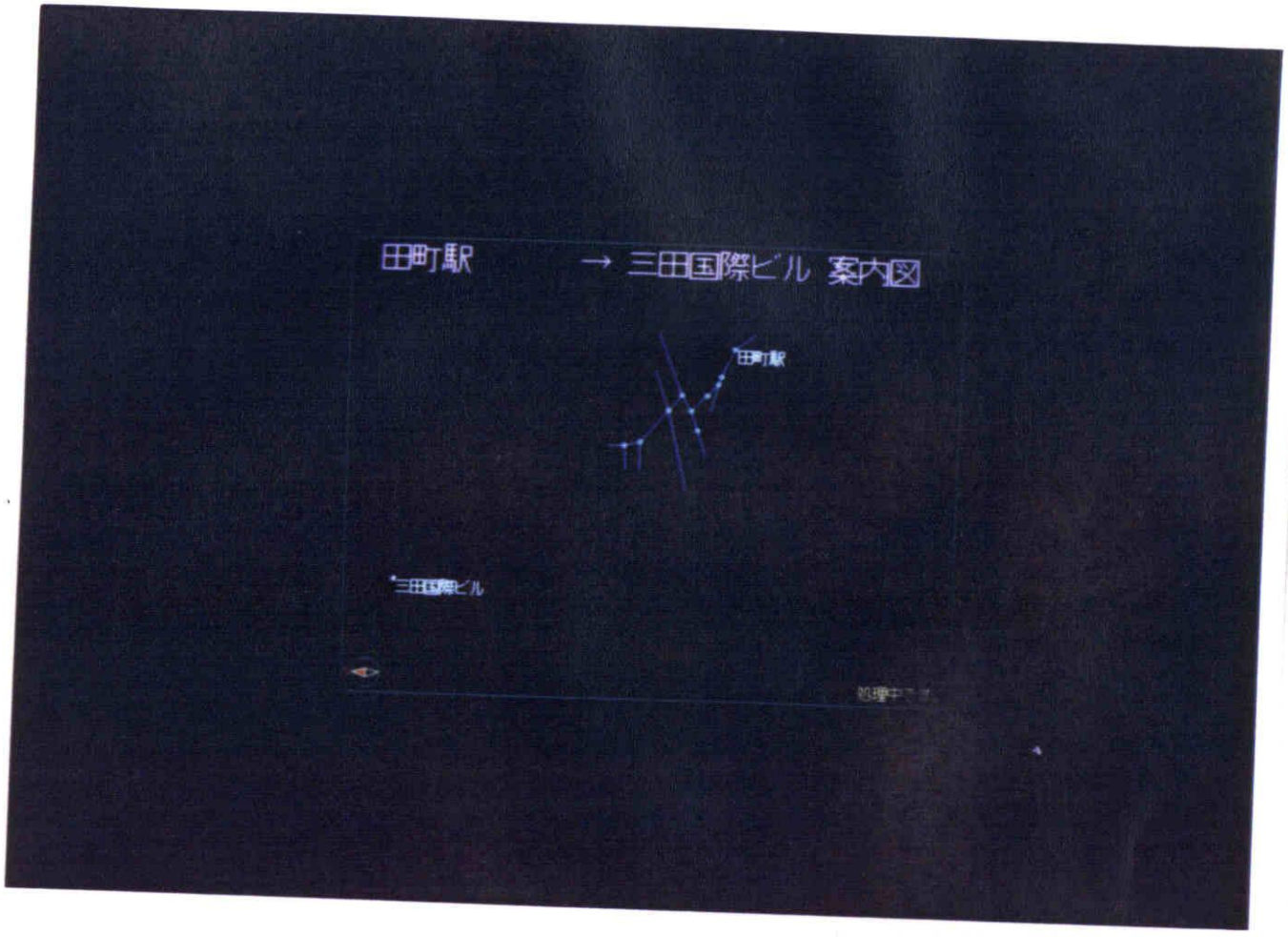


図５．１８　ＳＤ^{3}－ＭＧ経路探索の途中経過（２）

図５．１７と同様デモンストレーション用として表示された画面で、入力完了後１０秒程経過したときの処理の途中経過をあらわす。画面上プロットされている地点はすべて中間結果としてＴＤＢに登録され、評価値等の計算がなされる。

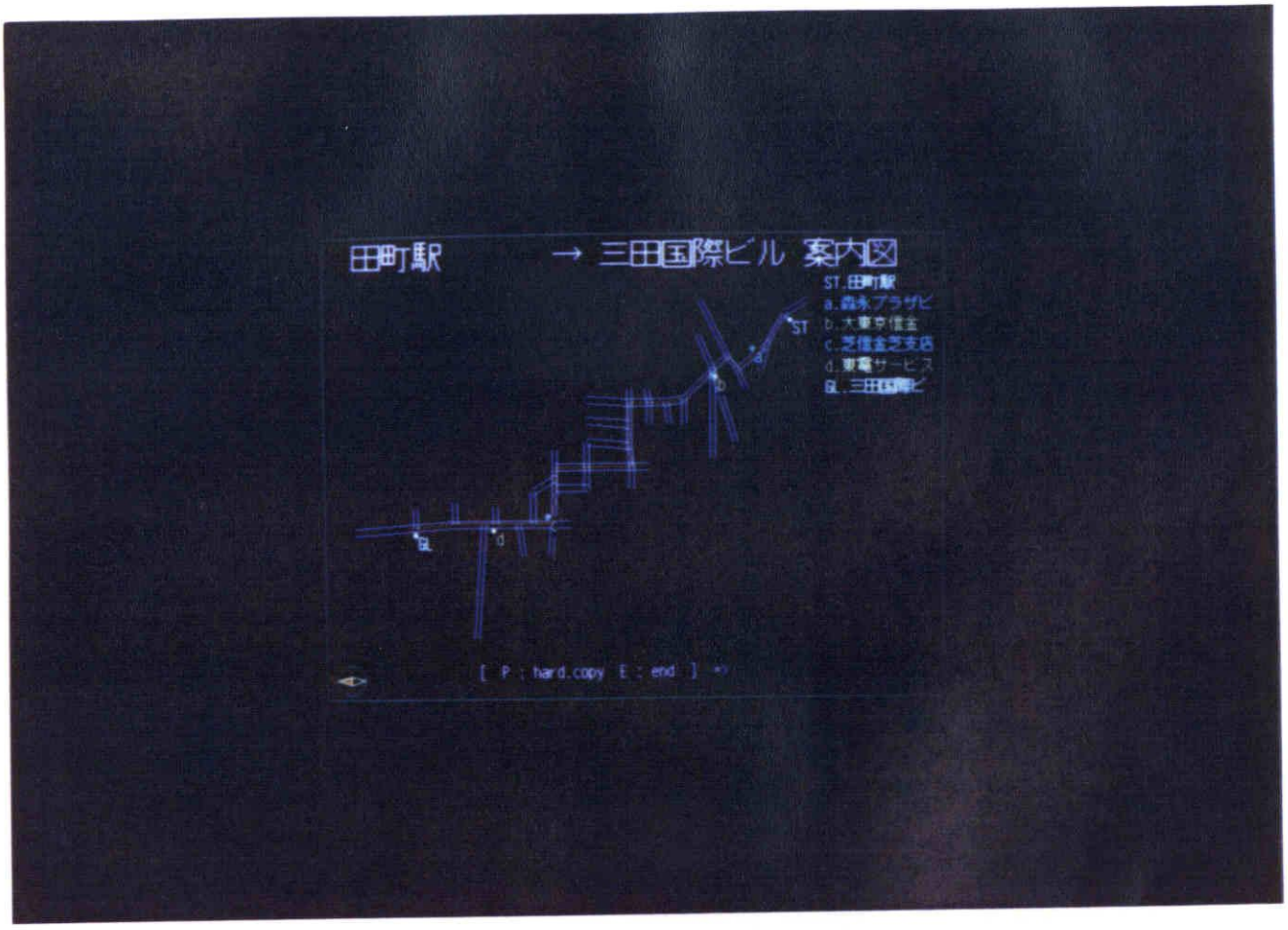
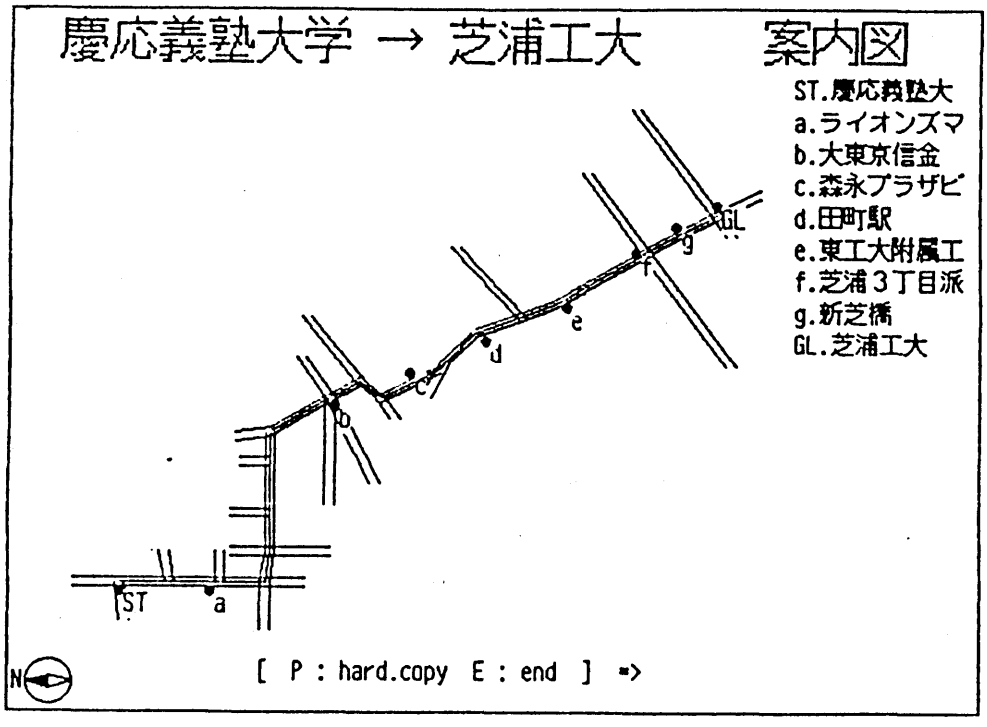


図５．１９　ＳＤ^{3}－ＭＧの出力結果（１）

図は、田町駅を出発点、三田国際ビルを目的点としたときのＳＤ^{3}－ＭＧシステムによる出力結果である。図中出発点はＳＴで、目的点はＧＬで示されている。経路上にある主要な建造物等は、それぞれａ、ｂ、ｃ・・のアルファベット記号で地点を、右の欄にその名称を表示する。但し、漢字混り文６文字以上の長い名称は、以降を削除して示されていることに注意。

(a)



(b)

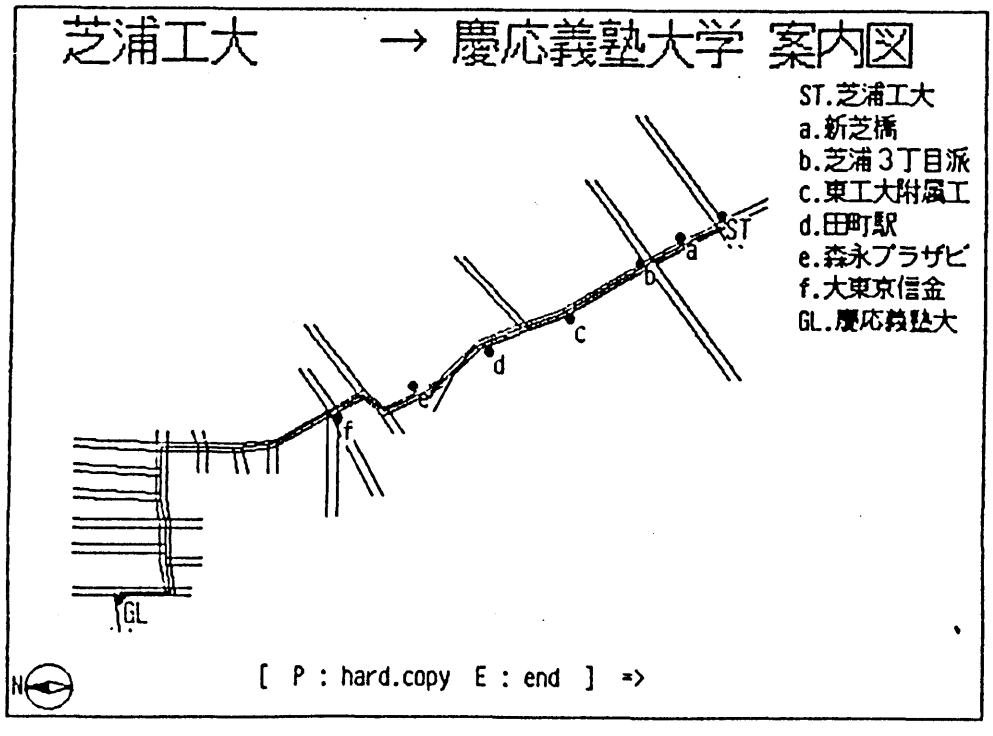


図５．２０　ＳＤ^{3}－ＭＧの出力結果（２）

図５．２０ａは、出発点を慶応大学、目的点を芝浦工大としたものであり、ｂは、出発点と目的点を入れ換えて実行させたときの出力結果の画面のハードコピーをとったものである。どちらも出発点と目的点を入れ換えただけの全く同じ略地図となっているが処理時間はａの場合の方が少し短い。

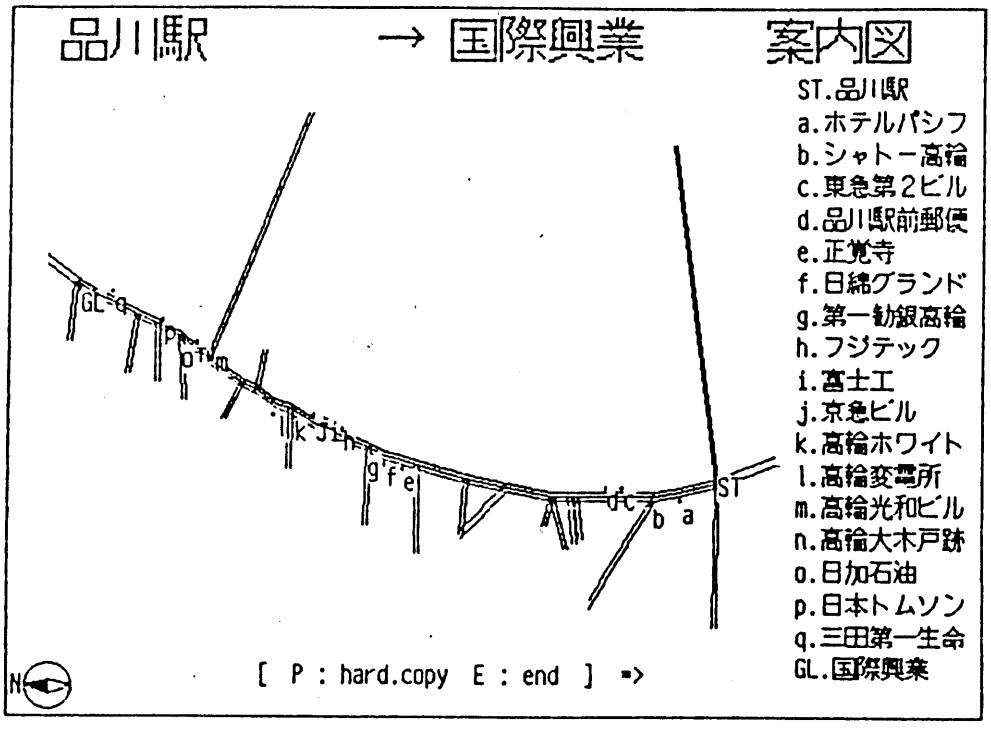
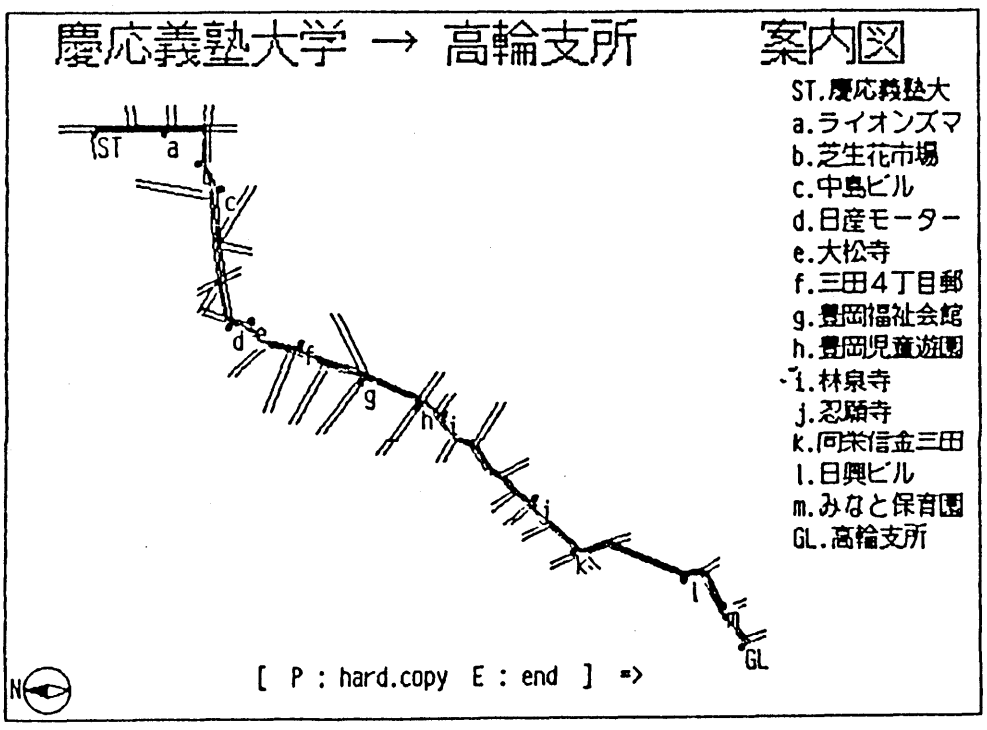


図５．２１　ＳＤ^{3}－ＭＧの出力結果（３）

図は品川駅を出発点、国際興業ビルを目的点とした時のＳＤ^{3}－ＭＧの出力結果である。

(a)



(b)

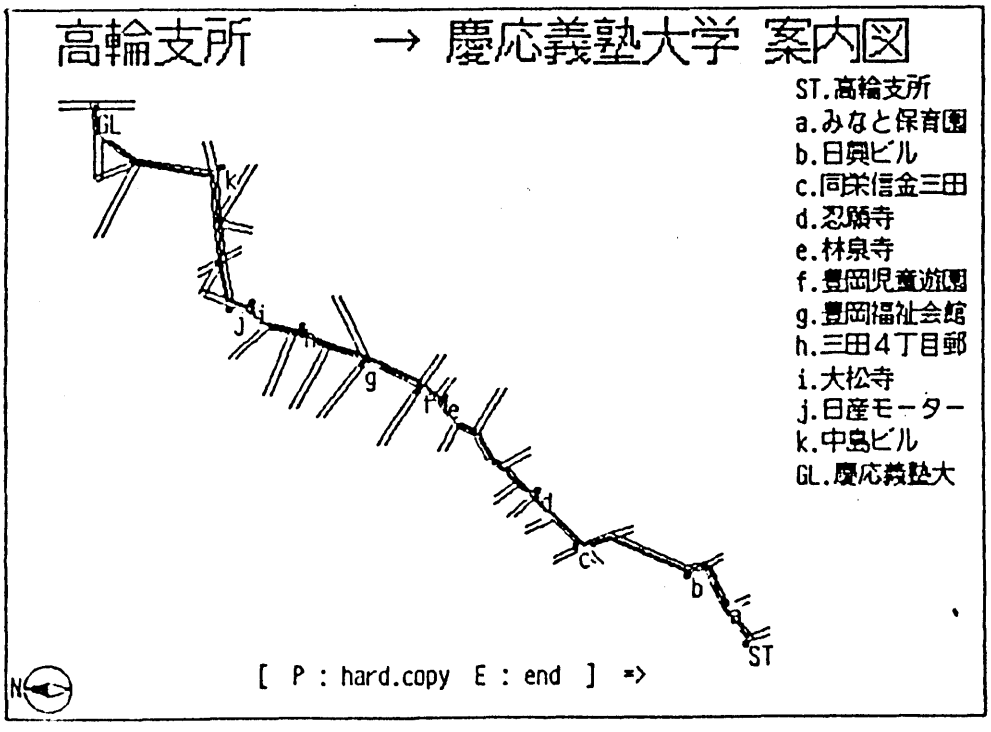


図5．22　SD³－MGの出力結果（4）

図５．２２．ａは慶応大学を出発点、高輪支所を目的点とした場合の出力結果を、ｂは出発点と目的点を入れ換えた場合の出力を示している。ａ、ｂとも全く同じ略地図が生成されているが、処理時間は異なっている。

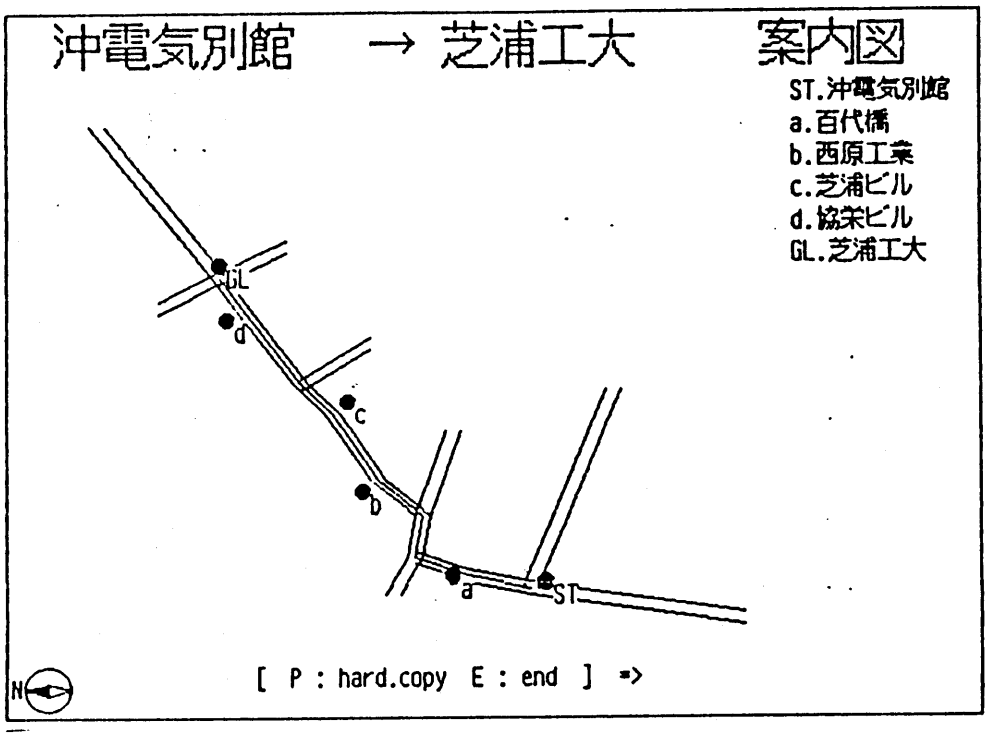
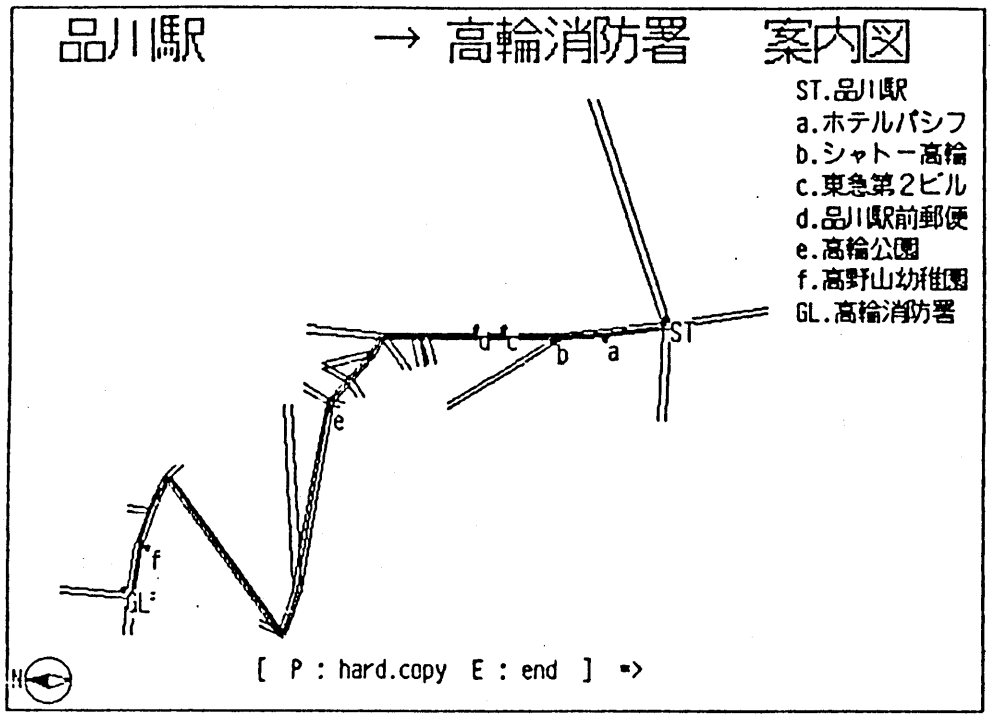


図５．２３　ＳＤ^{3}－ＭＧの出力結果（５）

図は沖電気別館を出発点、芝浦工大を目的点とした時の出力結果である。たとえば図５．２１と比較した場合、表示上の縮尺の違いに注意。

(a)



(b)

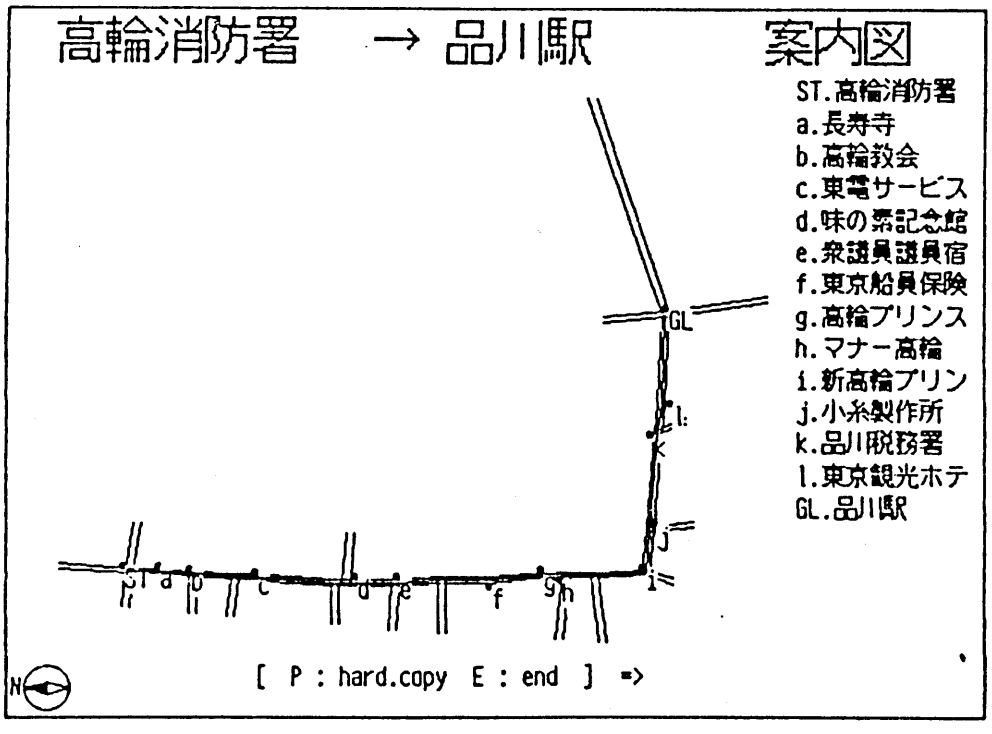


図5．24 SD³－MGの出力結果（6）

図５．２４．ａは、品川駅を出発点、高輪消防署を目的点とした時の出力結果であり、ｂは出発点と目的点を入れ換えた場合の出力結果である。ａからもわかるように経路が複雑であり、またＳＤ^{3}－ＭＧシステムは必ずしも最短経路を保証するものではないため、ａとｂとでは全く異なる経路導出が行われてしまっていることに注意。

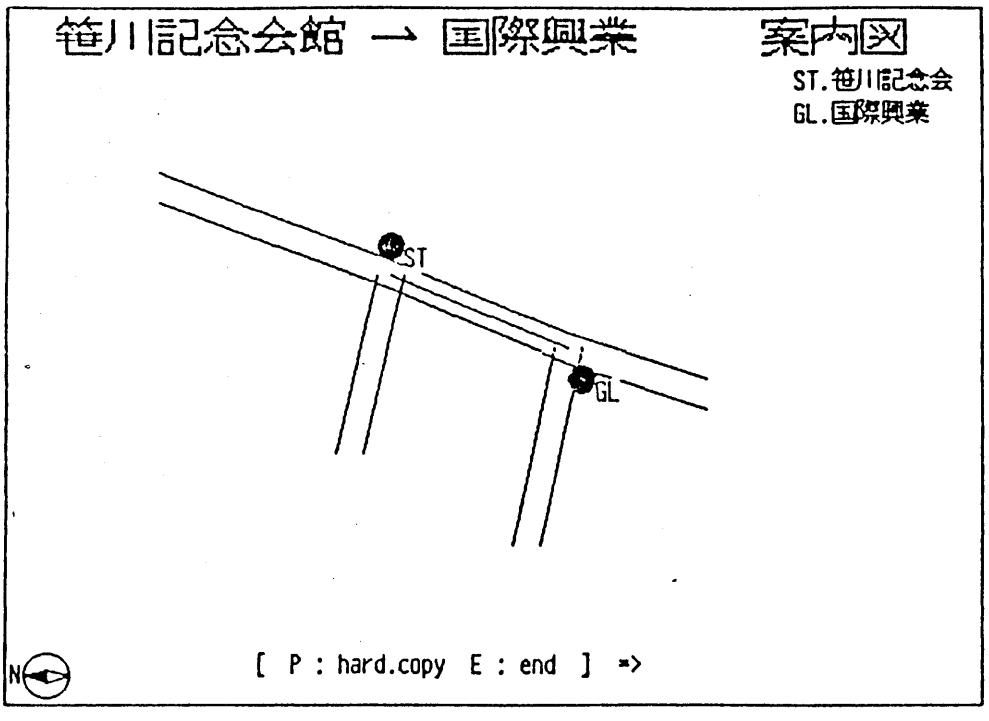


図５．２５　ＳＤ^{3}－ＭＧの出力結果（７）

図は、中継地点のない場合の経路を導出したものである。距離が短いため、たとえば図５．２１と比べて表示上の縮尺が大巾に異なっていることに注意。

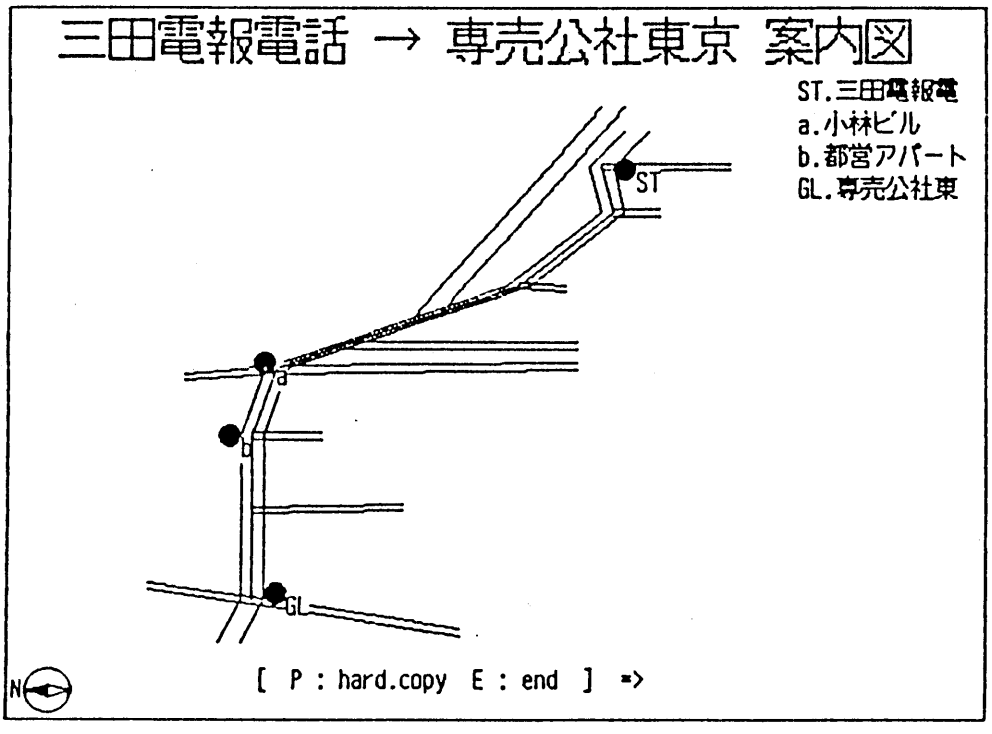
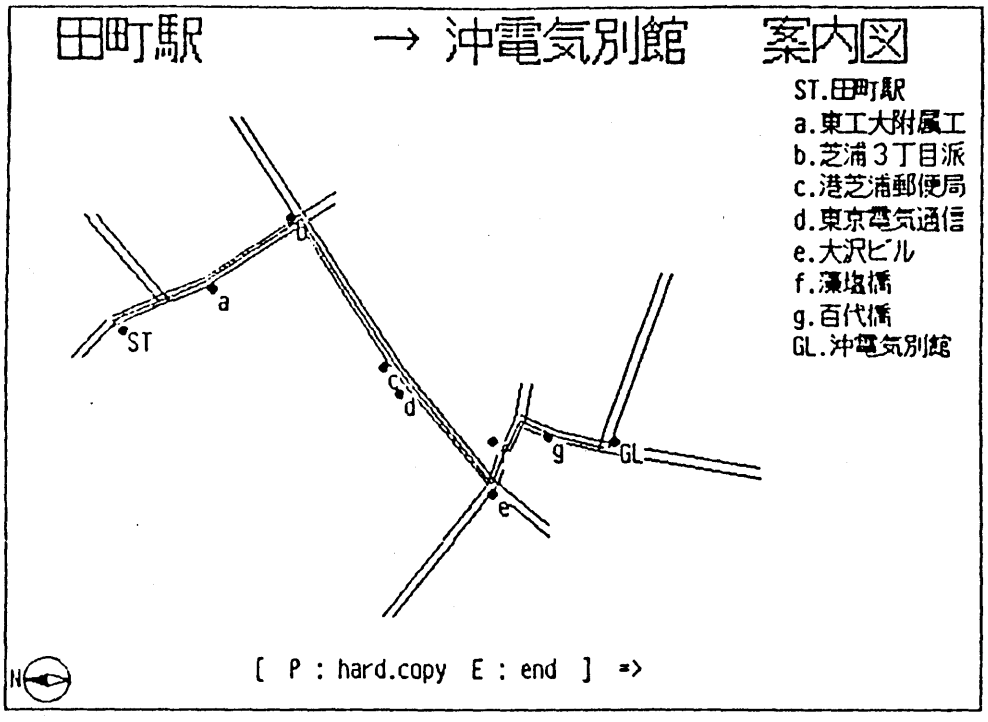


図5．26　SD³－MGの出力結果（8）

(a)



(b)

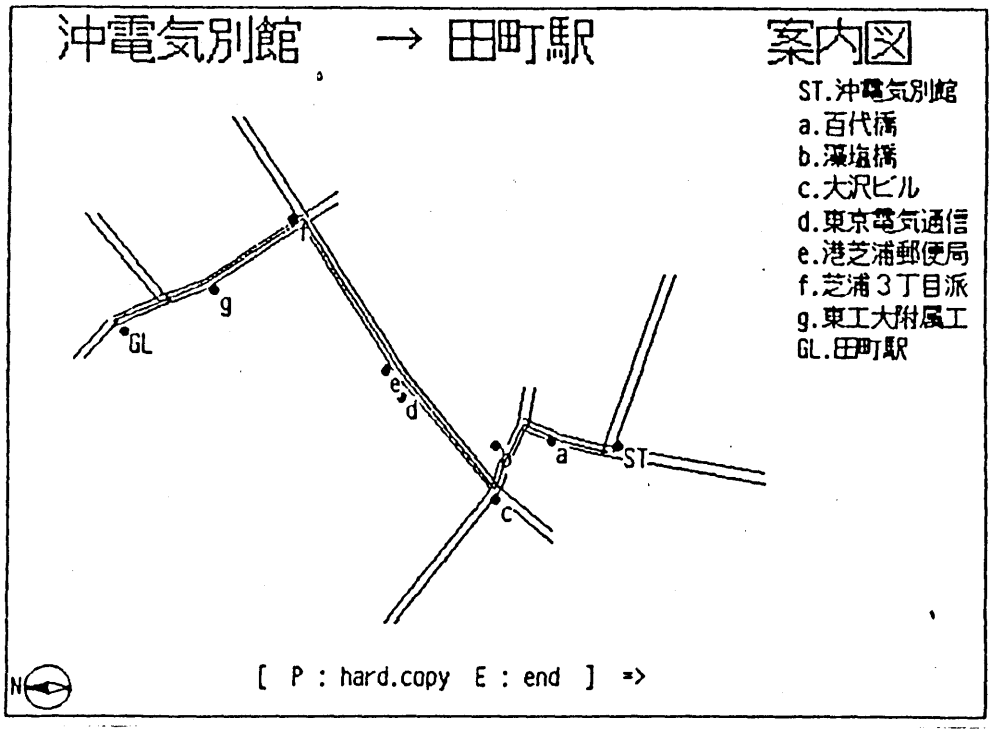


図５．２７　ＳＤ³－ＭＧの出力結果（９）

第　6　章

結論と今後の課題

本章ではＳＤ^{3}システムの特徴について要約し、あわせて今後の研究課題について概説する。

本論文では、分散型問題解決システムの構成法に関して、１つのアプローチであるSD^3システムとその具体的実現例である略地図発生システム（SD^3－MG）について、議論を行った。

またSD^3システムの構成要素であるネットワークサブシステム、分散データベースサブシステム、および分散型推論サブシステムについても、その基本的な特徴と研究開発の位置づけを論じた。それぞれのサブシステムの特徴を要約すると以下の様になる。

［１］ネットワークサブシステム

① データリンクレベルに Ethernet とコンパチブルな10M b/s 、ＣＳＭＡ／ＣＤ方式のローカルネットワークを採用。

② 分散データ処理指向の上位レベルネットワークアーキテクチャＢＡＮＥＴの実現。

③ 既存通信概念の拡張であるグループ内 Many to Many 通信の実現。

④ 従来アプリケーション層で実現されていたリライアブルブロードカスト機能をネットワーク側で実現。

［２］分散データベースサブシステム

① データの一貫性を維持するための同時実行制御をコミットメント制御（操作のアトミック性）とコンカレンシイ制御（順序づけの一貫性）とに明確に区別し、前者をネットワーク側で、後者を分散データベース側機能として実現。

② トランザクションの前解析方式の採用により、トランザクション種別に応じたコンカレンシイ制御とそれによる実行並列性の向上。

③ バス型ローカルネットワークの採用による容易で簡明なロケーショントランスペアレンシイの実現。

［３］分散型推論サブシステム

① 問題解決のための制御構造を述語のインタプリタから分離し、演繹推論を一般的な木の探索問題として表することによる問題に応じた柔軟な探索戦略の設定。

② データベースに中間結果を生成することによる膨大な中間結果への迅速なアクセスと重複した探索処理の回避。

③ キュアリ言語を備えた分散データベースとしても、又推論システムとしても動作可能とすることによる柔軟なＡＰインタフェースの提供。

④ 協調と競合プロトコルによる分散型並列推論機構の実現。

分散型問題解決システムのような高度の応用システムにおいては、これらのどのサブシステムの機能も実現のためには不可欠であることを、本プロトタイプシステムの開発の過程で確信した。

分散型問題解決システムという技術は極めて新しいものであり、未だ確立された方式も存在しないため、残された技術課題も多いが、筆者は、本論文で述べた基本方式のもとに、以下のような技術課題を今後の研究の対象としたいと考えている。

［１］マルチプロセッサ型システムの実現

ＳＤ^{3}システムにおいては、単独でも問題解決が可能な各ノードプロセッサがそれぞれローカルネットワークに結合された構成をとっていた。このため、ネットワーク上でのプロトコルのオーバヘッドが、高度の並列推論を実行するためには極めて大きな障壁となっている。また各ＤＰに汎用のパーソナルコンピュータを使用しているため、システム全体としては高価なものとなっている。今後は、各ノードプロセッサをプロセッサボードで代用させ、ボード間をローカルなバスで結合した、疎結合のマルチプロセッサアーキテクチャによりＳＤ^{3}を実現させることを計画している。

この場合、協調と競合のプロトコルは、バス上のプロトコルとなり手順も簡略化され、伝送速度もローカルネットワークの場合に比し大巾に向上するため、高速の並列推論実行が期待される。

［2］異なる応用プログラムの開発

SD^3システムの具体的適用であるSD^3－MGにおいては、問題解決の一形態であるプラン生成（略地図発生）を扱った。SD^3システムの基本アーキテクチャおよび方式を検証する意味でも今後は、診断システムや質問応答システムあるいは類似図形の検索システムといったCA／Cタイプの別の応用システムの開発を行う予定である。

［3］分散データベース機能の拡張

SD^3システムにおいては、IDBの更新は、現在ローカルな操作によってなされている。これはIDBが分散データベースシステムの管理下におかれていないことに起因するが、今後は分散データベースの機能拡張を行うことにより、ファクトデータだけでなくルールの更新をも含む広義の分散データベースによりこれらの統合を行うことを予定している。

［4］FA／Cシステムへのアプローチ

SD^3システムは既に述べたようにCA／Cシステムを前提としている。このため入力情報を補完したり、システムによる修正を加えなくてはならないような応用（e．g．談話理解、ソフトウェア作成支援etc．）に対しては、その基本アーキテクチャやノード間プロトコルを修正する必要が生ずるかもしれない。今後の極めて大きな技術課題は、FA／Cシステムとなる適切な具体的応用を見つけ出し、それを実現するための基本アーキテクチャとSD^3アーキテクチャとの差異を検討し、システムの開発を行うことである。

筆者の属する研究室においては、現在、典型的FA／Cシステムとなると考えられる「ソフトウェアの変換システム」についての基本検討を開始した処であり、今後の研究の進展が期待される。

［5］実用化、商品化

SD^3システムを真にエンドユーザに解放し、実用に供し得るものとするためには、知識ベースエディタやユーティリティ等の開発が必要となる。またEVLの記述言語やエンドユーザがシステムをアクセスするためのマン・マシンインタフェース部の充実が必須であり、今後の実用化に向けて、これらの検討を行っていく予定である。

なお、さらに発展的研究として、たとえばカーネギーメロン大学のＰｒｏｆ．Ｊｅｎｓｅｎ他により研究が進められているような「分散オペレーティングシステム中にこのような問題解決の機構をとり込む研究」も、今後のコンピュータ技術、ソフトウェア技術動向を考察すると見逃すことはできない課題であるということを付記しておく。

《謝辞》

本研究は、1974年４月より沖電気工業株式会社総合システム研究所において開始され、現在に到ったものである。本研究は、多技術分野にまたがったもので規模も大きく、また長期に渡ったため、研究の過程での多くの方々の励ましと助力なしには、所定の成果を達成し得なかったものと筆者は考えている。

特に本研究を進めるに当り、終始懇切な御指導と御支援をいただいた名古屋大学工学部情報工学科の福村晃夫教授ならびに沖電気工業株式会社システム本部開発第三部長推野努博士には深い感謝の意を表します。同時に、本研究の実施に際して様々な便宜と御助力をはかっていただいた長谷川潔前研究部長、松下温博士（システム本部開発第二部長）および佐野勝久現研究部長に心から感謝致します。

また、具体的研究課題を解決する過程で貴重な御教示、御示唆をいただいた慶応大学工学部相磯秀夫教授ならびにカーネギーメロン大学計算機科学科Ｐｒｏｆ．Ｊｅｎｓｅｎの両教授に心から謝意を表します。

さらに本研究の当初から終始検討に参加し、方式実現のための試案作りに多大の御協力をいただいた総合システム研究所の吉田勇君には深く感謝致します。

また、本プロトタイプシステムの開発に当り、プログラム作成を担当していただいた岸田一君（現カーネギーメロン大学訪問研究員）および総合システム研究所の古田香代里嬢、森久聡子嬢ならびにフジシステム（株）の皆様に心からの謝意を表したいと思います。

最後に本論文を編集するに当り、株式会社ＭＡＳシステムの皆様には多大の協力をいただきました。ここに紙面を借りて感謝の意を表します。

《参考文献》

マルチプロセッサ関連（分野A）

A-1] P. H. Enslow Jr., Multiprocessor Organization – a Survey, ACM Comp. Surveys, Vol. 9, no. 1, March 1977.

A-2] W. A. Wulf, C. G. Bell, C. mmp – A multi-miniprocessor, Proc. of Fall Joint Computer Conference, pp. 765-777, 1972.

A-3] H. Yamazaki, Performance Evaluation of multiprocessor Systems, Tech. report UIUCDCSR-77-891, Department of Computer Science, University of Illinois at Urbana – Champaign.

A-4] C. H. Sauer, K. M. Chandy, The Impact of Distribution and Disciplines on Multiple Processor Systems, Comm. of ACM, vol. 22, No. 1, Jan. 1979.

A-5] S. M. Fuller, J. K. Ousterhout, L. Raskin, P. et al., Multi-microprocessors: An Overview and Working Example, Proc. of IEEE, Vol. 61, no. 2, Feb. 1978.

A-6] J. Liu, C. Liu, N. Miyazaki, H. Yamazaki; Performance Evaluation of Multiprocessor Systems Containing Special Purpose Processors, PODSTAWY STEROWANIA, Tom 10, polish Academy of Science, pp. 201-222 (1980).

A-7] J. W. S. Liu, C. L. Liu, Performance analysis of multiprocessor systems containing functionally dedicated processor, Depart. of computer Science, University of Illinois at Urbana, Champaign, TR UIUCDCSR-835, 1977.

コンピュータネットワーク（分野B）

B-1] L. Pouzin "Virtual circuit v.s. Datagrams Technical and Political Issues", INWG General Note No. 106, January 1976.

B-2] V. Cerf, A. Mackenzie, R. Scantlebury, H. Zimmermann "Proposal for an Internetwork End-to-End Protocol", INWG General note No. 96.1, Jan. 1978.

B-3] H. Zimmermann "The Cyclades end-to-end protocol", Proc. 4th Data Communications Symposium, ACM-IEEE, Oct. 1975, pp. 7-21 to 7-26.

B-4] A. Mackenzie "A Host-Host Protocol for the ARPA Network", NIC No. 8246, 1972.

B-5] P. G. Cullum "The Transmission Subsystem in Systems Network Architecture", IBM Systems, Journal Vol. 15 No. 1, 1976.

B-6] J. M. McQuillan, D. C. Walden "The ARPA Network Design Decisions", Computer Networks Vol. 1 No. 5, August 1977.

B-7] D. C. Walden "Host-to-Host Protocols", Network Systems and Software, Infotech state of the art report 24, 1975.

B-8] Y. Matsushita, et. al. "An Overall Network Architecture Suitable for Implementation with either Datagram or Virtual Circuits Facilities", Computer Communication Review Vol. 8 No. 3, July 1978.

B-9] T. Kawaoka et. al. "A Logical Structure for Heterogeneous Computer Communication Network Architecture" ICCC - 78. September, 1978.

B-10] ISO 'Data processing – open systems interconnection – basic reference model' DIS 7498, Switzerland (1980)

B-11] J. B. Brenner, "Preliminary View on Administration and Use of Logically Related Sessions", Contribution to ISO/TC97/SC16/WG2 N60.

B-12] Metcalfe et al.: Ethernet: Distributed packet switching for Local Computer Networks, CACM, Vol. 19, No. 7 pp. 395-404 (July 1976).

B-13] The Ethernet, A Local Area Network: Data Link Layer and Physical Layer Specifications, Version 1.0, September 30, 1980

B-14] Y. Matsushita, H. Yamazaki, et al: An Evaluation of Virtual Circuits and lettergram Services, Computer Networks, pp. 287-294, North-Holland Publishing Company (1979).

B-15] Yamazaki, H. et. al: A Proposal for Broad cast Architecture Network (BANET). Proc. of 6th ICCC. (September, 1982)

B-16] 松下、山崎他：DONAにおける端末の考え方、情報処理学会、コンピュータネットワーク研究会17-5,
(1978年10月)

B-17] 松下、山崎他：DONAのハイレベルプロトコル、情報処理学会、コンピュータネットワーク研究会16-3,
(1978年7月)

B-18] 松下、山崎他：バーチャルサーキットとレターグラムサービスの適用領域について、電子通信学会研究会、
EC78-69, (1979年2月)

B-19] 岸田他：分散処理向きローカルネットワークについて、情報処理学会分散処理システム研究会14 - 4 (1982年7月)

B-20] 吉田他：分散処理向きローカルネットワークアーキテクチャBANET、情報処理学会「ローカルエリアネットワークシンポジウム」 (1983年9月)

データベースおよび分散データベース（分野C）

C-1] Rothnie, J. B. and Goodman, N.: A Survey of Research and Development in Distributed Database Management, Proc. of Int. Conf., 3rd VLDB, pp. 48-62 (1977).

C-2] Rothnie, J. B. et al.: Introduction to a System for Distributed Database (SDD-1), ACM Trans. on Database System, Vol. 5, No. 1, pp. 1-17 (1980).

C-3] Bernstein, P. A. et al.: The Correctness of Concurrency Control Mechanisms in a System for Distributed Database (SDD-1), ACM Trans. on Database System, Vol. 5, No. 1, pp. 52-68 (1980).

C-4] Stonebraker, M.: Concurrency Control and Consistency of Multiple Copies of Data in Distributed INGRES, IEEE Trans. on software engineering, Vol. SE-5, No. 3 (May 1979).

C-5] Fernandez, E. B., et al: Database Security and Integrity, Addison-Wesley (1981).

C-6] Date, C. J.: An Introduction to Database Systems (Second Ed.), Addison-Wesley (1977).

C-7] Gray, J. N., et al.: Granularity of Locks and Degrees of Consistency in a Shared Data Base, Proc. IFIP TC-2 Working Conference on Modelling in Data Base Management Systems, pp. 365-394 (1976).

C-8] Gray, J. N.: Notes on Data Base Operating Systems, Operating Systems- An Advanced Course, Springer-Verlag, pp. 393-481 (1978).

C-9] Eswaran. K. P., et al.: The Notions of Consistency and Predicate Locks in a Database System, Comm. ACM. Vol. 19, No. 11, pp. 624-633 (1976).

C-10] Ries. D. R., et al.: Effects of Locking Granularity in a Database Management System, ACM TODS Vol. 2, No. 3, pp. 233-246 (1977).

C-11] Ries, D. R., et al.: Locking Granularity Revisited, ACM TODS. Vol. 4, No. 2, pp. 210-227 (1979).

C-12] Gray. J. N., et al.: Granularity of Locks in a Shared Data Base, Proc. 1st Int'l Conf. on VLDB, pp. 428-451 (1975).

C-13] Verhofstad. J. S. M.: Recovery Techniques for Database Systems, ACM Computing Surveys, Vol. 10, No. 2, pp. 167-195 (1978).

C-14] Gray. J. N., et al.: The Recovery Manager of the System R. Data Base Manager, ACM Computing Surveys, Vol. 13, No. 4, pp. 223-242 (1981).

C-15] Downs, D. and Popek, G. J.: A Kernel Design for Secure Database Management System, Proc. 3rd Int'l Conf. on VLDB, pp. 507-514 (1977).

C-16] C. A. Ellis, "A Robust Algorithm for Updating Duplicate Data Bases" Proc of the SECOND BERKELEY WORKSHOP, 1977, pp. 146-158.

C-17] P. Alsberg, J. Day "A Principle for Resilient Sharing of Distributed Resources" proc 2nd International Conference on Software Engineering, 1976.

C-18] R. H. Thomas, "A Solution to the Concurrency Control Problem for Multiple Copy Data Base", Comcon 78, 1978, pp. 56-62.

C-19] P. A. Bernstein, J. B. Rothnie, N. Goodman, C. A. Papademitriou, "The Concurrency Control Mechanism of SDD-1. A System for Distributed Data Base (The Fully Redundant Case)", IEEE, Trans. Software, 1978, Vol. SE-4, No. 3, pp. 154-168.

C-20] P. A. Bernstein, D. W. Shipman, J. B. Rothnie, N. Goodman, "The Concurrency Control Mechanism of SDD-1: A System for Distributed Database (The General Case)" Technical Report, CCA-77-09, 1977.

C-21] C. H. Lee, R. Shastri, "Distributed Control Schemes for Multiple-Copied File Access in a Network Environment" Compsac 77, 1977, pp. 722-728.

C-22] R. Epstein, M. Stonebraker, E. Wong "Distributed Query Processing in a Relational Database Systems" ACM SIGMOD, 1978 pp. 169-180.

C-23] Lamport, L.: Time, Clocks, and the Ordering of Events in a Distributed System, CACM, Vol. 21, No. 7, July 1978, pp. 558-565.

C-24] Rosenkrantz, D. J. et. al.: System Level Concurrency Control for Distributed Database Systems, ACM Trans. on Database Systems, Vol. 3, No. 2, pp. 178-198 (June 1978).

C-25] Thomas, R. H., "A Majority Consensus Approach to Concurrency Control for Multiple Copy Data Bases", ACM Trans. on Database Systems Vol. 4, No. 2, 1979.

C-26] Chu, W. W., "Performance of File Directory Systems, for Data Bases in Distributed Networks", Proc. AFIPS 1976 NCC, Vol. 45, pp. 577-587.

C-27] H. Hammar, D. Shipman "An overview of Reliability Mechanisms For A Distributed Data Base Systems" proc. COMPCON 1978, spring.

C-28] Yamazaki, H. et al.: A Graph Theoretic Approach for Fault Detection and Recovery in a Distributed Database, Proc. of ICCC, pp. 237-242 (1980).

C-29] Yamazaki, H. et. al.: A Pre-analysis Scheme for Execution of Transactions on a Distributed Database. Proc. of COMPCON Fall '82 (September, 1982)

C-30] H. Yamazaki, Y. Matsushita, et al "A Hierachical Structure for Concurrency Control in a Distributed Database System" sixth Data Communication Symposium, Nov. 1979.

C-31] Y. Matsushita, H. Yamazaki et al: Cost Evaluation of Directory Management Schemes for Distributed Database Systems, Proc. of ACM-SIGMOD, pp.117-124, May, 1980.

C-32] 山崎他：分散データベースにおける同期制御のための階層型プロトコル，情報処理学会，分散システム研究会 2-1 (1979年9月).

C-33] 山崎：分散ﾃﾞｰﾀﾍﾞｰｽ － 更新ﾄﾗﾝｻﾞｸｼｮﾝの実行と制御に関する記号演算、情報処理学会論文誌、Vol 23, No 1, PP35-42, (1982年1月)

C-34] 山崎、吉田：ﾃﾞｰﾀﾍﾞｰｽの完全性と機密保護、情報処理、Vol 23, No 10, PP916-924, (1982年10月)

知識情報処理、演繹データベースおよび分散型問題解決システム(分野D)

D-1] H. Gallaire et. al.: An overview and Introduction to Logic and Data Bases, Logic and Data Bases, Plenum Press, pp. 3-30 (1978).

D-2] J. M. Nicolas and H. Gallaire: Theory vs. Interpretation, Logic and Data Bases, Plenum Press pp. 33-54 (1978).

D-3] D. Wright: Prolog As a Relationally Complete Data Base Query Language which can handle least fixed point operators. University of Kentucky, Tech. report No. 73-80, (1980).

D-4] Smith, R.: The Contract Net Protocol; High-level Communication and Control in a Distributed Problem Solver, IEEE Transactions on Computers, Vol. C-29, No. 12, pp. 1104-1113, Dec. 1980.

D-5] Lesser, V. and Corkill, D.: Functionally Acurate, Cooperative Distributed Systems, IEEE Trans. on Systems, Man, and Cybernetics. Vol. SMC-11, No. 1, pp. 81-96, Jun. 1981.

D-6] Barr, A. and Feigenbaum, E. A.: The Handbook of Artificial Intelligence, Vol. 1, 2, 3, William Kaufmann Inc, 1981.

D-7] Feigenbaum, E. A.: The Art of Artificial Intelligence; Themes and Case Studies of Knowledge Engineering, Proc. 5th IJCAI, 1014, 1977.

D-8] Hayes-Roth, F., Waterman, D. A. and Lenat, Douglas, B.: Building Expert Systems, Addison-Wesley, 1982.

D-9] Nilson, N. J.: Artificial Intelligence, Information Processing 74, North-Holland, 778, 1974.

D-10] Shorrliffe, E. H.: Computer-Based Medical Consultation; MYCIN, American Elsevier, 1976.

D-11] Erman, L. D., Hayes-Roth, F., Lessar, V. and Reddy, D.: The HEARSAY-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty, ACM Computing Surveys, 12,2,213, 1980.

D-12] E. Y. Shapiro: Inductive inference of Theories from facts, T. R. 192, Yale Univ., Dept. of Computer Science (1981).

D-13] J. Minker: An experimental Relational DataBase System Based on Logic, Logic and Data Bases, Plenum Press, pp. 107-147 (1978).

D-14] V. Lesser and L. Erman: Distributed interpretation: A model and Experiment, IEEE, trans. on Computer, Vol. 29, No. 12, pp. 1144-1163 (1980).

D-15] R. Smith: A. Framework for Distributed Problem solving, UMI Research Press, Ann Arbor, Michigan (1981).

D-16] 新世代コンピュータ技術動向調査委員会：知識ベースの動向、先端的コンピュータに関する調査研究報告書［技術動向編］PP19-PP110，昭和58年3月。

D-17] 新世代コンピュータ技術動向調査委員会：知識ベースシステムの動向、先端的コンピュータに関する調査研究報告書［技術動向編］-II，PP13-77，昭和59年3月。

D-18] 大須賀節雄：知識ベースとその応用，情報処理，Vol．21，No.12，pp.1231−1241，1980

D-19] 大須賀節雄：知識ベース技術の展望，情報処理，Vol.23，No.10，pp.967−974，1982.

D-20] 大須賀節雄：アドバンスト・データベース・システム－設計・診断・研究開発・意思決定のツールとして，シンポジュウム論文集，情報処理学会（1981）。

D-21] 大須賀：知識ベース技術の展望，情報処理，23巻10号（1982）

D-22] 吉田他：分散型演繹データベースシステムSD^3、情報処理学会知識工学と人工知能研究会、37－3、1984年11月

D-23] 和田他：SD^{3}上の略地図発生システム、情報処理学会知識工学と人工知能研究会、37－4、1984年11月

D-24] 山崎：分散型演繹データベースシステムSD^{3}とそのプロトコル、情報処理学会論文誌、Vol 26, No.2, PP288～295 (1985年3月)。

数学　その他（分野E）

E-1] W. Feller: An Introduction to Probability Theory and Its Application (Vol. I), Wiley, New York (1950).

E-2] Feller "An Introduction to Probability Theory and Its Applications" Vol. II, Wiley (New York), 1966.

E-3] E. G. Coffman, P. J. Denning "Operating Systems Theory", Prentice-Hall, Englewood Cliffs, New Jersey.

E-4] E. Mendelson: Introduction to Mathematical Logic, Second Edition, D. Van Nostrand Company (1979).

E-5] 細井勉：論理数学、数理科学シリーズ1、筑摩書房（1974年9月）。

E-6] 高橋英俊：岩波講座・情報科学の歩み-1、岩波書店（1983年）。