

分散型演繹データベースシステム: SD²

吉田 勇、和田光教、古田香代里、山崎晴明

沖電気工業株式会社 総合システム研究所

1. はじめに

主として軍事上の信頼性向上のために研究が始められた分散データベースの技術は、その後マイクロプロセッサ技術の急速な進歩によって分散処理が指向されるに従い、信頼性の向上、応答速度の向上、拡張の容易性などのメリットが認識されて、軍事以外の広範な目的をめざして研究が盛んに行なわれている 1), 2), 3)。特に最近ではパーソナルコンピュータ、オフィスコンピュータ等の普及により、これらのコンピュータ上に簡易なデータベースを構築する試みが盛んに行なわれているが、これらのデータベースがオフィス等への応用において、真にその効果を発揮するためには、これらをネットワークによって有機的に統合することが必要であり、分散データベース技術はこの時発生する様々な技術的課題を解決するための技術としても注目を集めている。

一方、関係データベースに推論機構を取り入れて全体として知識データベースを構築しようという試みもいくつかなされている 4), 5), 6)。また、広く問題解決や意思決定支援等を行なうことが可能なように、知識ベース化、エキスパートシステム化する試みも盛んに行なわれつつある 7)。従って、将来はネットワーク上に分散された個々のデータベースを統合して分散データベースシステムが構築されたように、こうした知識ベース、エキスパートシステム群を統合、管理するシステムが必須となることが予想される。このようなシステムを分散型問題解決システムと呼び、その代表的な例として分散型意思決定支援システムをあげることができる。

分散型意思決定支援システムでは、通常はローカルの意思決定が各サイトに置かれた知識ベース支援のもとに行なわれているが、しばしばこれらの知識ベースを統合したグローバルな意思決定を行なう必要が生ずる (図1)。例えば、ある企業のプロジェクト計画と管理というアプリケーションを想定すると、統合化オフィスシステムにおいては、各プロジェクト対応にローカルな知識ベースを持ち、それぞ

れの人員、進捗、予算等、プロジェクト遂行上の種々の管理と意思決定サポートが行なわれている。

これに対し新規プロジェクトの発生により経験 n 年以上、 $\times\times$ 関係の技術者 m 人が必要となったとする。

全体プロジェクトへの影響が最も少ない m 人の選定といった問題にはグローバルな意思決定サポートが関与することとなる。

このような分散型問題解決システムは、重要であるにもかかわらず、まだ余り考察されておらず、スタンフォード大学のコントラクト・ネット 8) やカーネギーメロン大学の分散型解釈モデル 9) 等わずかの研究事例しか報告されていない。コントラクト・ネットは分散型問題解決システムを構築する際の枠組みとして、マネージャとコントラクタと呼ばれる動的に変化する役割を担う二者間でのプロトコルを規定したもので、システムの編成形態が明確で部分問題への分割が比較的容易に行なえる場合には適しているが、そうでない場合には制御がきわめて複雑になってしまうことが予想される。

一方、分散型解釈モデルは分散型 Hearsay II 等の経験をもとに提唱されたもので、情報をノード間で相互交換し、その情報を自ノードへフィードバックするという形で問題解決が進行するため、明確に部分問題に分割できないようなものに対しては効果的であるが、部分問題が比較的明確に定義できるような問題に対しては、重複処理の発生の可能性が極めて高いため、適切なアプローチとはならなくなってしまう。

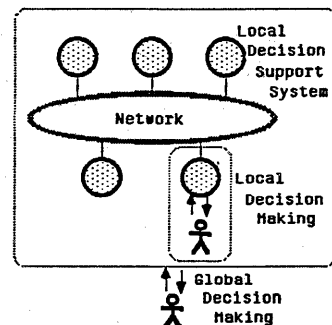


図1. 分散型意思決定支援システム概念

I DB (Intentional DataBase) 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。E DB (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戦略が格納される。

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。

2.2 システムの動作概要

SD³ 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。

I DB

ルール1: $A(h, x) \leftarrow EA(h, x)$

ルール2: $A(h, x) \leftarrow HPC(i, h), A(i, y), P(f(y), f(x)), C.EA(h, x)$

トリガ1: [Insert to EA] on Rule 2

トリガ2: [Delete from C.EA] on Rule 2

E DB

$P(x, y)$: 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

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列である。 $f(x)$ はリスト x の最初の要素を取り出す関数である。例えば、 $f(\text{三郎, 二郎, 太郎}) = \text{三郎}$ となる。

I DB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、もちろん述語 $A(h, x)$ 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C.EAは外延的に定義された述語EAの補集合としてトリガによって定義される。

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は探索の戦略として、TDB内のHPCにレコード (h_0, h_1) 、 (h_0, h_2) 、 (h_1, h_3) 、 (h_2, h_4) を書き込む。また、EAにはレコード $(h_0, \text{太郎})$ を初期値(探索のルート)として書き込む。
- (2) EVAはHCIに対し、ゴール節 $\leftarrow A(h_3, x)$ を出す。
- (3) HCIはルールを順次適用し、まずルール2より $A(h_1, (\text{二郎, 太郎}))$ を求める。
- (4) トリガが起動され、レコード $(h_1, (\text{二郎, 太郎}))$ が中間結果としてEAに書き込まれる。
- (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 $A(h_3, (\text{三郎, 二郎, 太郎}))$ を求める。
- (6) トリガが再び起動され、レコード $(h_3, (\text{三郎, 二郎, 太郎}))$ がEAに書き込まれる。

- (7) H C IはE V Lに解 $x = (\text{三郎}, \text{二郎}, \text{太郎})$ を返す。
- (8) E V LはH C Iにゴール節 $\leftarrow A(h_4, x)$ 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C. E Aにより、 $A(h_2, (\text{二郎}, \text{太郎}))$ 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 $x = (\text{四郎}, \text{花子}, \text{太郎})$ を返す。

このように本システムは、木探索の問題に置き換えて、データベースの演繹検索を実行するものであり、E V LがH C I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T D Bに蓄積のみ行なう方式をとっている 12)。

3. 分散型演繹検索機構

S D³の各ノードは木探索を実行しているが、同時にまた、S D³システム全体でも、組織的な一つの木探索となっている必要がある。

以下では、こうした分散型の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) 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

として『展開可能』(expandable)であるという。

さらに、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分類することができる。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、その和集合をとれば、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。

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. $P(a, y) \wedge Q(y, x)$)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 $P_1, P_2, P_3, Q_1, Q_2, Q_3$ に分割し、対 (P_i, Q_i) をノード N_i に格納したとする(図4)。このように配置した

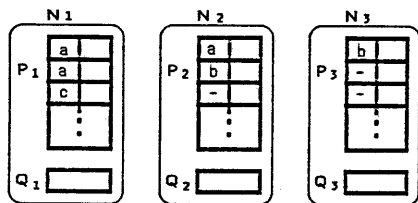


図 4 . 未編成システムの例

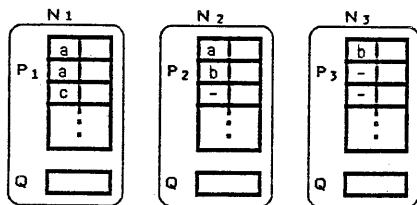


図 5 . 半編成システムの例

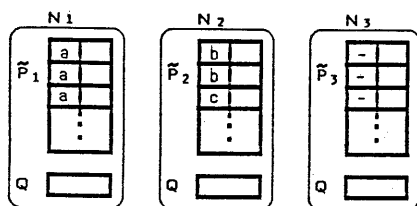


図 6 . 全編成システムの例

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5)。

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6)。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。

このように定義したとき、コントラクト・ネットは全編成システムに最も良く適合し、また分散型解釈モデルでは未編成もしくは半編成のシステムを扱っていると考えられる。

分散型探索システムにおいて、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といえることができる。

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よぶ。

4. SD³ の協調プロトコル

4.1 プロトコル設計上の留意点

分散型問題解決システムにおいて、真に分散化のメリットを活かすためには、ノード間の効率のよいプロトコルを設計しなければならない。SD³の協調プロトコルの設計にあたっては主に以下の点に留意した。

(1) コントロールの強さと処理の並列性

一般に、コントロールを強くする、すなわち、どのノードでどの処理を実行しているのかをすべて管理しようとする、各ノード間で交換されなければならないメッセージの数が増えて、通信オーバーヘッドが増大し、システム全体の処理の並列性が低下する。一方、コントロールを弱くして、他のノードでどんな処理を実行するかにはあまり干渉しないようにすると、通信オーバーヘッドは減少し、処理の並列性が向上して全処理量も増大するが、逆に各ノードで同一の処理を実行してしまう可能性も大きくなる。最適なコントロールの強さは、ノード数や各ノードに格納されているデータの重複度に依存すると考えられる。

SD³ の協調プロトコルの設計においても、以下の2つの方式が考案された。

- ① あるノードが他ノードに処理を依頼する協調メッセージを1回ブロードキャストするだけで、その処理を行なうか否かは各ノードの独自の判断にまかせられ、またおこなうとしてもそのことを依頼ノードには通知しない。この方式では各ノードが同一の処理を実行してしまう可能性が大きい。
- ② あるノードが他ノードに処理を依頼する協調メッセージに対して、それを請け負うことを競合メッセージで応答する。複数ノードが競合メッセージを応答した場合には、プライオリティにより実行ノードを選択し、処理の重複を抑える。

上記の2方式は明確な優劣があるわけではなく、SD³ 上にのるアプリケーションの性質によって、その適、不適が変わるものであり、今後どのような条件のもとではどちらの方式が適しているかの定量的な評価を行なう必要がある。しかし、現時点では現在開発中のアプリケーションの編成形態が全編成であるので、②の方式を採用している。

(2) ブロードキャスト・ネットの有効利用

SD³ はBANETという分散処理向きに新たに開発されたブロードキャスト型のLAN上に構築されている。従って、SD³ の協調プロトコルの設計にあたっては、このブロードキャスト・ネットの利点を大いに有効利用できるように考慮した。

従来の1対1通信に基づくネットワークで上記②の方式の協調プロトコルを実現しようとする、以下のような手順となる。

- ① あるノードが他ノードに処理を依頼する協調メッセージをブロードキャストする。
- ② 処理を請け負うノードは競合メッセージを依頼ノードに返す。
- ③ 依頼ノードは競合メッセージを応答してきたノードの中から最適なノードを選択して、そのノードに実行指示メッセージを、その他のノードには実行拒否メッセージを送る。

ブロードキャスト・ネットの特徴は、あるノードが依頼ノードに送る競合メッセージをすべてのノードで受信できる、ということである。SD³ ではこの性質を有効に利用して、以下のようにネットワーク内に流れるメッセージの量を減少させている。

- 各ノードは他ノードの競合メッセージを取り込み、もし自身が先に受信した同一の協調メッセージに対してその処理を請け負うべく、競合メッセージを準備していたとしても、もし受信された他ノードの競合メッセージのプライオリティの方が自身のプライオリティより高ければ、競合メッセージの送出を中止する。これにより余分な競合メッセージがネットワーク内を流れるのを防止することができる。
- 複数の競合メッセージがネットワーク内に送出された場合にも、実行ノードの選択は依頼ノードが行なうのではなく、各ノードがそれぞれプライオリティに基づいて行なう。従って依頼ノードから実行指示メッセージや実行拒否メッセージを発行する必要がなくなる。

(3) 協調メッセージの送出契機

あるノードが他ノードに協調メッセージにより処理を依頼する契機としては、もちろん自身のエリア記述(ADN)の範囲を越えてしまう場合がある。

この場合だけを協調メッセージの送出契機とする案もあるが、もし与えられた問題が一つのノード内だけで解決してしまう場合には、そのノードだけが動作し他のノードは遊んでしまうことになり、分散システムの利点を活かさない。このような場合にもうまく問題が分割されて、複数ノードで並列処理を行なえば効率がずっとよくなる。しかし、例えば木探索の1ステップの展開毎に協調メッセージを送出したのでは通信オーバーヘッドが大きくなりすぎて、望まれる並列処理の効率が得られなくなってしまう。協調メッセージ送出の最適な頻度は通信処理と各ノードのローカルな推論処理との処理時間の比率に大きく依存している。

SD³ ではNをシステム定数とし、木探索をNステップ展開し終った時点で、たとえまだすべての中間結果が自身の達成域にあっても、協調メッセージ

を送出する方式をとった。最適なNの値は試作システムを評価して求めて行く予定である。

4.2 協調プロトコルの概要

前述の留意点を考慮して設計されたSD³の協調プロトコルの概要を以下に示す。

- (1) APに問題S_iが与えられる。APはS_iを協調メッセージにより全ノードにブロードキャストする。
- (2) S_i (初めはi=0)を含む協調メッセージを受信したノードN_iは、それがN_iの関連域にあれば、優先度P_iを付加して競合メッセージを全ノードにブロードキャストする。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、S_iの自ノードのR-Q中の順位Γ_iとノード番号N_iとの対とする。半編成の場合、S_iが達成域にあれば1、そうでなければ0とする。
- (3) 競合メッセージを送出したノードは直ちにS_iをR-Qにおき、展開の対象とする。
- (4) N_iはS_iを順次展開してゆき、導出した中間結果S_jがN_iの達成域を越えた場合にはS_jを協調メッセージにより全ノードにブロードキャストする。またN_i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るから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キャストする。
- (5) N_i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AP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N_kからのS_iに対する競合メッセージを受信したN_jは、優先度を比較し、自身の方が低ければR-Qより、S_i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Γ_jとΓ_k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0に優先する。

5. 終りに

本稿では演繹検索機構を持った分散データベースシステムSD³の概要紹介と、分散型問題解決のためのプロトコルの提案を行った。本システムの特徴を要約すると以下のようになる。

- (1) 通常のデータベースに対するクエリ処理をファクト集合に対する演繹推論機構として一般化し、すべての演繹検索を一般的な木の探索問題として表現している。
- (2) 推論機構の基本部であるHCIからなるべく制御機構を分離して、その上のメタレベルで柔軟に探索ストラテジーを決められるので、全解探索や一解探索、あるいは評価値による探索等、様々な探索パターンに適合できる。
- (3) トリガ・メカニズムの導入により中間結果をTDBに格納し、DBMの管理下に置いたので、膨大な中間結果への迅速なアクセスや、重複した探索の回避等が可能である。
- (4) 全編成、半編成のいずれのシステムにも適用可能で、ブロードキャスト・ネットの利点を有効に利用する協調プロトコルが開発された。

なお、現在試作中のアプリケーションでは静的に全編成となるようEDB、IDBの分散配置が行われている。今後は半編成や未編成システムとなるアプリケーションについても開発を予定している。

特に未編成システムの場合、ファクトデータのみならず、ルールの転送も必要となってくるので、それらを含めた分散型問題解決プロトコルの設計や、分散クエリの実行ストラテジーの検討、さらに各ノードへ割り当てられる知識の量とシステム全体のパフォーマンスの関係についての考察といった問題が今後の主要な研究課題として残されている。

[参考文献]

- 1) Rothie, J. B. and Goodman, N : A Survey of Research and Development in Distributed Data Base Management , Proc of Int. Conf.. 3rd V LDB, pp. 48-62 (1977)
- 2) Rothie, 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)
- 3) 山崎: 分散データベース——更新トランザ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, pp. 35-42 (1982)
- 4) Minker, J : An Experimental Relational Database System based on Logic . Logic and Database, Plenum Press, pp. 107-147 (1978)
- 5) 松尾他: 推論関係型データベース管理システム Ad b i s、情報処理学会論文誌 Vol. 24, No. 2, pp. 249-255 (1983)
- 6) 横田他: Prolog による推論機構と関係データベースの結合、知識工学と人工知能 32-3 (1983)
- 7) 大須賀: 知識ベース技術の展望、情報処理 Vol. 23, No. 10, pp. 967-974 (1982)
- 8) Smith R : The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver , IEEE trans. on Computer, Vol. C-29, No. 12, pp. 1104-1113 (1980)
- 9) Lesser V and Erman L : Distributed interpretation: A Model and Experiment , IEEE trans. on Computer, Vol. C-29, No. 12, pp. 1144-1163 (1980)
- 10) Yamazaki H , et. al : A Proposal for Broadcast Architecture Network (BANET) , Proc. of 6th ICCS, pp. 115-120 (1982)
- 11) Yoshida I , et. al : Banet - a local area network for distributed data processing , computer communications, Vol. 7, No. 1, pp. 3-11 (1984)
- 12) 和田他: SD³ 上の略地区発生システム: SD³ - MG、知識工学と人工知能 37-4 (1984)