データベースシステム 93－1
システムソフトウェアと
オペレーティング・システム 59－1
（1993．5．27）

並列データベース管理システム: Kappa-P

河村 元夫
(財) 新世代コンピュータ技術開発機構

並列推論マシン PIM 上で動作する並列データベース管理システム Kappa-P について述べる。このシステムは, データモデルとして非正規関係モデルを採用している. 機能的には、拡張関係代数を中心にした言語処理系、および並列処理のための分散データベース機能が拡充されている。処理面としては、KL1 に適したプロセス指向の設計、および言語の各階層に対応した各層での並列処理が特徴である。

A Parallel Database Management System: Kappa-P

Moto Kawamura
Institute for New Generation Computer Technology
Mita Kokusai Bldg. 21F, 4-28, Mita 1-chome, Minato-ku, Tokyo 108, JAPAN

The Kappa (Knowledge Application-oriented Advanced Database and Knowledge Base Management System) project is one of the knowledge base management software projects at ICOT. We are now involved in the design and development of a parallel database management system (Kappa-P) to run on a parallel inference machine (PIM) and on its operating system (PIMOS). Kappa-P is based on a nested relational model and the system consists of language processors including extended relational algebra and distributed DBMS for parallel processing, mainly. I describe an outline of Kappa-P system.

1 はじめに

第五世代コンピュータプロジェクトでは, 知識情報処理システム (KIPS) の中核的機能であるデータベース, 知識ベース管理機能を提供することを目的とし知識ベース管理システム (KBMS) の研究開発を行ってきた [9]. 知識ベース管理はデータベース管理機能を含むべきであり, それをデータベース管理の拡張として位置づけている. そのなかで, Kappa (Knowledge Application-oriented Advanced Database and Knowledge Base Management System) [8, 5] は, KIPS や KBMS のデータベース エンジンの役割を果たし, 知識表現言語/知識ベース言語 *Quixote* [6] はその上位層にあたる. Kappa プロジェクトでは, 中期に逐次推論マシン PSI で動作する逐次データベース管理システム (DBMS) Kappa-I,II を開発し, 後期には並列推論マシンとそのオペレーティング システム PIM/PIMOS の環境で動作する並列 DBMS Kappa-P の試作をおこなった.

逐次版の Kappa-II は, 自然言語処理のための辞書や, ゲノムや蛋白質などの分子生物学データベースなどの新しい分野で利用され, その有効性が確かめられている. ゲノムや蛋白質などのデータは, 解析技術の進歩により爆発的に増加しており, そのデータ解析や格納には並列マシンのパワーが必要で, Kappa-P の重要な応用と位置付けている.

Kappa-P は, 並列マシン上の DBMS であり応用プログラムと同じマシンで動作することを前提に設計している. このようなシステムは他に Bubba [1] や PRISMA [2] などがある.

以下、2 節で設計方針, 3 節で特徴, 4 節で問い合せ処理について述べる.

2 設計方針

KIPS が扱うデータや知識は複雑な構造をしており, かつ量も膨大なものがある. たとえば, 遺伝子情報処理の分子生物学データベース GenBank/HGIR database[4] は, 塩基配列と特徴の記述と関連する文献情報からなり, この塩基配列は短いものから非常に長いものまでさまざまである. このような複雑なデータを伝統的な関係モデルで扱うには, データ表現能力の点と効率的な問合せ処理の点で問題が多い. また, このデータベースは, 解析技術の進歩にともない近年飛躍的に増大している. さらに, このデータに対する操作は, 類似検索など計算能力を必要とするものが多く, そのデータ量と計算量から, 並列マシンの能力が必要になっている.

Kappa-P が動作する環境は, 並列推論マシン PIM とそのオペレーティング システム PIMOS の環境で, 実装言語は, 並列言語 KL1 である. PIM は MIMD 型の疎結合と密結合が混合されたハイブリッド並列マシンある. 10 台程度の要素プロセッサが, 共有バス/共有メモリにより結合され一つのクラスタとなり, それが, ネットワークで結ばれている. 共有メモリは, 数百メガ バイトで, ディスクは各クラスタに取り付けることができる.

この環境では、ハードウエアとしては、クラスタを構成する PE、ネットワークでつながれたクラスタ、クラスタに取り付けられたディスクなどが並列に動作する。ソフトウエアとしては、並列 DBMS 自身とそのアプリケーション プログラムがそれぞれ並列を意識して書かれており、両方とも同じ並列マシン上で動作する。
このような背景のもと、並列化にあたっては次の点を考慮した。

- データモデル
  複雑な構造データを扱えること
  大量データに対する考慮

- ハードウエア資源の有効利用
  疎結合並列処理, 密結合並列処理の両方を考慮
  大容量主記憶を生かした主記憶データベース機能
  ディスクに対する並列アクセス

- 並列マシン上の DBMS
  応用プログラムとの通信量の削減

3 特徴

前節の方針に基づき, PSI 上で実装された自然言語処理システムや遺伝子情報処理システムで有効利用されている Kappa-II の経験を最大限に生かして, Kappa-P を設計した. つぎのような特徴がある.

• 非正規関係モデル

複雑な構造データを効率的に扱うために非正規関係モデルを採用する. これは, 関係モデルの自然な拡張になっており, *Quixotε*の無限構造をもたないオブジェクト項のクラスに対応する.

データ構造としては, 属性が階層構造を持ち, 値として繰り返し値が許されることが, 関係モデルとの違いである. また, その意味論として, 非正規関係に固有の操作である行ネスト操作に対し独立した意味を与えている. つまり, 非正規関係 $\{[a/c_1, b/\{c_2, c_3\}], [a/c_1, b/\{c_3, c_4\}]\}$ と $\{[a/c_1, b/\{c_2, c_3, c_4\}]\}$ は同じ意味を持っている. これにより, ネスト構造を意識しない問合せが可能となる. 先の非正規関係に対し問合せ? $-[a/X, b/\{c_2, c_4\}]$ を出すと, どちらも $X = c_1$ が結果として得られる. これは, 意味論としては基本的に関係に基づきながら表現 (および蓄積構造) の効率化をめざしているという点で, 関係モデルの自然な拡張であり, 多値従属性をもつデータの表現と処理の効率化が可能である. この意味を反映して, 関係代数を, 行, 列のネスト, アンネスト操作を含む拡張関係代数として定義しなおした.

また, さまざまな知識を格納できるようにデータ型としてタームを追加し, 大量データに対する検索処理の効率化を目的に, 索引としてのみ存在する属性なども追加した.

• システム構成

DBMS は大量のデータを扱うので, 疎結合並列処理における通信量が大きな意味を持ってくる. そのため, 密接に関連するデータを同じクラスタに置くなどのデータの配置が重要で, さらに, 問合せ処理において通信量が少なくなるようなプランを立てる必要もある. これらをおこなうためには, 分散データベースの構成が向いている. 各クラスタにローカル DBMS(LDBMS) と呼ばれる DBMS を配置し, 全体で一つのデータベースを管理する. この LDBMS は, それ自身で DBMS としての全機能をもち, 複数 LDBMS が関与する問合せを処理するために, 二相コミットプロトコルに基づく分散トランザクションの機能を持っている. また, 複数の LDBMS により一つのデータベースを管理するためテーブル名などの大域情報の管理が問題になるが, それを管理するサーバ DBMS (SBDMS) の複製を作ることにより, アクセスの集中を回避する. クラスタに割り当てられた, ローカル DBMS は, その内部処理で密結合向きの並列処理をおこなう.

• データ配置と並列処理

効率的な並列処理を行うためには、各クラスタの負荷とクラスタ間の通信量をバランスさせてやる必要がある。DBMS の場合はデータが二次記憶上に保存されしかも大量であることから、データの配置と並列処理方式が密接に関係してくる。

- 分散配置
  分散配置は、複数 PE の計算能力を利用するもっとも単純な場合である。この場合、問合せ処理時の通信量と負荷を考慮しテーブルを配置する必要がある。

- 水平分割
  水平分割は、論理的には一つテーブルをレコードを単位に複数に分割しそれぞれ異なるLDBMSに配置することである. 分割された個々のテーブルには基本的に同じ演算が出され, 最後に個々の演算結果をまとめる. 一つテーブルが一つのクラスタでは扱い切れないほどの量の場合や特にデータ検索などのCPU負荷が大きい演算を重視する場合などに有効である。

- 複製
  関係の複製によって、可用性を高めたりアクセスの集中を回避することができる。現在の実装では大域情報管理でのみ実現されている。

問合せは, 演算子をノード、その間の依存関係をアークとしたデータフローグラフとみなすことができる。このグラフのうち入力が揃ったノードから順に評価していくことにより並列処理をすることができる。また、演算ノードをプロセス、アークを演算対象である組を流すストリームとすることで、パイプラインによる並列処理をおこなうこともできる。Kappa-Pはこの処理方式を採用しているが、これは実装言語であるKL1のプログラミング スタイルそのものといえるものである。ただし、ストリームを要求駆動にすることにより、処理や中間結果の片寄りをなくすことを狙っている。

• 主記憶データベース

PIMは各クラスタは, 数百メガ バイトの大容量主記憶を持つ. この大容量主記憶を使って, 主記憶データベースをサポートする. この主記憶データベースでは二次記憶へ反映はおこなわない. しかし, 多数の中間結果テーブルを生成するシステム, たとえば演繹データベースなどでは, これで十分である. また, 主記憶データベースと二次記憶データベースを組み合わせて, ログ付き主記憶テーブルも提供する. 主記憶データベースと二次記憶データベースの両方に同じ内容のテーブルを作っておき, 読み系問い合わせは主記憶データベースに出し, 更新系問い合わせは主記憶データベースと二次記憶データベースの両方に同じものを出す. これにより読み系問い合わせは主記憶データベースの性能でて, 更新系問い合わせは並列処理できるので二次記憶データベースの性能ぐらいでおこなうことができる.

4 問合せ処理

Kappa-Pは, 原始コマンドと*KQL*の二種類のコマンドを提供している.

原始コマンドは, 非正規関係のためのプリミティブな操作を提供している. それは, レコード識別子によるポインタ操作を基本にした操作で, 関係代数に比べ低レベルではあるが効率的な操作が可能である. 索引にはこのレコード識別子が格納され, 求まったレコード識別子の集まり間の演算も行なうことができる. 非正規関係は属性値に繰り返しが許されるため, このレコード識別子も単純な構造ではなく, その演算にも様々な工夫をおこなっている [7].

*KQL*は, 拡張関係代数からなる式の集まりからなり, 問合せ中に一時的な操作を定義したり, 推移閉包を処理するためのループを記述したりできる. このKQLは, 一つのトランザクション単位でまとめて受けとられ, 各種最適化され, 拡張関係代数処理のための中間言語に変換され, 通信量を考慮し各ローカルDBMSへの部分問合せに分割され, 各ローカルDBMSに送られ実行される.

• アプリケーション プログラムとの関係

並列マシン上のDBMSとしてアプリケーション プログラムと同じマシン上で動作するため、DBMSとアプリケーション プログラムの関係を次のように考えている。

- アプリケーション プログラムとの通信量の削減

- 一つのアプリケーション プログラムから並列に要求が出せること

アプリケーション プログラムもまた並列に動作するので、一つのアプリケーション プログラムが複数のインタフェース プロセスを作り DBMS に対し並列に問合せを出すことを許す。問合せの結果はインタフェース プロセスが生成されたクラスタに集められアプリケーション プログラムに渡される。

アプリケーション プログラムと DBMS 間の通信量の削減のためには、アプリケーション プログラムを、対象データの存在する LDBMS と同じノードで実行してやればよい。しかし、問題は CPU の負荷にも関連するので、アプリケーションプログラムの一部分の処理で大きく通信量を減らせる処理のみに限るべきである。たとえば、応用に依存した類似検索などを行う場合、DBMS に対しては一次検索を行い、その結果に対しアプリケーション プログラム側で二次検索を行うことがあるだろう。水平分割されていないテーブルに対しては、テーブルの存在するノードで二次検索を実行してやればよい。対象が水平分割テーブルの場合、実体は複数テーブルで複数ノードに分かれて存在するので、二次検索はその複数のノードで実行すべきである。しかし、水平分割テーブルはユーザには一つのテーブルとしてみせているので、水平分割テーブルとして扱うと、結果は DBMS の内部処理で一つにまとめられてしまう。もちろん、水平分割テーブルとしてではなく、それを構成するテーブルをその対象とすれば可能ではあるが、それでは構成テーブルの個数が増えた場合などプログラムを書き直さなければならない。これを解決するために、二次検索処理をレコードを選別するフィルタとして与えられるようにしたものが、フィルタ付きレコード読み出し機能である。図 1のように二次検索はレコードを選別するフィルタとして結果が一つにまとめられる前に行われる。

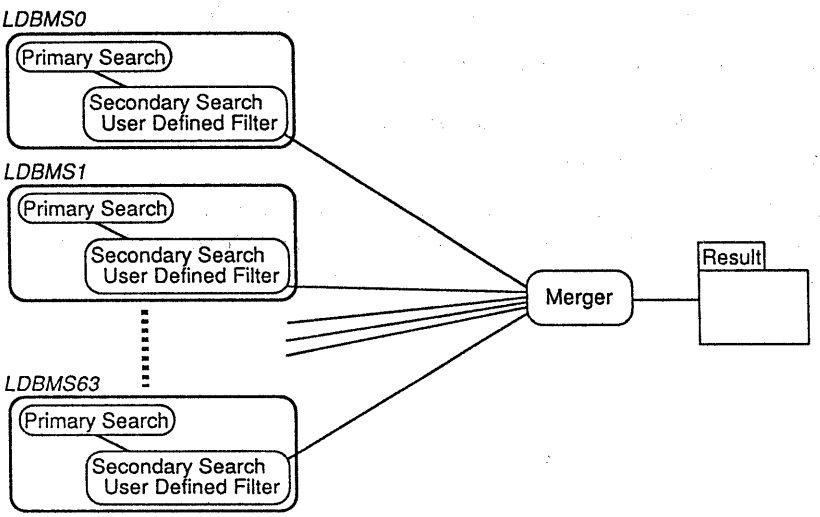

図 1: フィルタ付きレコード読み出し機能

5 まとめ

Kappa-P は現在 PIM 上で動作しており, 分子生物学データベースの一つである蛋白質データベースを格納し、類似検索の一種であるモチーフ検索などの応用プログラムが動いている. また, 性能評価として基本的なコマンドの評価と関係データベース用の代表的なベンチマークであるウィスコン

シン ベンチマークをおこなった. 今後, PIM 上により多くのデータベースを格納し Kappa-P のさらなる評価を行なうとともに, Unix 上の KL1 処理系 [3] 上に移植し, より広くつかわれ得るようにしていく予定である.

参考文献

[1] H. Boral, W. Alexander, L. Clay, G. Copeland, S. Danforth, M. Franklin, B. Hart, M. Smith, and P. Valduriez, "Prototyping Bubba, A Highly Parallel Database System", IEEE Transactions on Knowledge and Data Engineering, Vol. 2, No. 1, March 1990

[2] P. Apers, C. Berg, J. Flokstra, P. Grefen, M. Kersten, and A. Wilschut, "PRISMA/DB: A Parallel, Main Memory Relational DBMS", IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 6, December 1992

[3] T. Chikayama, T. Fujise, and H. Yashiro, "A Portable and Reasonably Efficient Implementation of KL1", submitted to ICLP'93

[4] "GenBank/HGIR Technical Manual", *LA-UR 88-3038*, Group T-10, MS-K710, Los Alamos National Laboratory, 1988.

[5] M. Kawamura, H. Sato, K. Naganuma, and K. Yokota, "Parallel Database Management System: Kappa-P", *FGCS'92*, 1992.

[6] H. Yasukawa , H. Tsuda, and K. Yokota, "Objects, Properties, and Modules in *Quixote*", *Proc. Int. Conf. on Fifth Generation Computer Systems*, ICOT, Tokyo, June 1-5, 1992.

[7] 川村ほか, "Kappa-P のアンネスト/ネスト処理", 第 45 回情処全国大会, 5R-08, 1992.

[8] K. Yokota, M. Kawamura and A. Kanaegami, "Overview of the Knowledge Base Management System (Kappa)", *FGCS'88*, 1988.

[9] K. Yokota and H. Yasukawa, "Towards an Integrated Knowledge-Base Management System — Overview of R&D on Databases and Knowledge-Bases in the FGCS Project", *FGCS'92*, 1992.