6Q-9 オブジェクト指向型マルチメディア知識ベース Jasmine のプログラムインタフェースの実現について

富士通株式会社

鈴木 文雄　石川 博　牧之内 顕文

1. はじめに

我々はオブジェクト指向型マルチメディア知識ベース管理システムを開発した。本システムを利用すれば、大規模な知識（データ）を効率良く処理するアプリケーションを容易に開発できる。

本稿では、2章で従来のシステムの問題点とその解決策、3章で本システムの全体構成、4章でオブジェクトの格納形式、5章で処理の高速化について述べる。

2. 従来のシステムの問題点とその解決策

従来のシステムには次のような問題点がある。

① 従来の関係データベースでは関係という単純なデータ表現しかないために、実世界の複雑な情報を直接的に表現し操作することができない。このためユーザにとってデータベースの設計が難しいものとなる。

② 従来のオブジェクト指向プログラミング言語では、遅延処理のため実行時に属性やメソッドを探索するので、実行速度が遅い。さらに、

③ 大規模かつ恒久的なオブジェクトの管理ができない。

上記の問題に対して、次のような解決策をとった。

① オブジェクト指向の概念に基づいた操作言語を提供して、実世界の複雑な情報を直接的に表現し操作できるようにした。この操作言語をシステムが自動的にデータベース操作言語に翻訳するので、データベースの設計や操作は不要になる。

② 翻訳時に遅延処理を施して、実行時のオーバーヘッドをなくした。これにより、属性操作やメソッド実行がより高速になる。

③ データベースを利用する上で、スペース効率の良い格納方式の採用した。また、オブジェクトの条件検索を効率の良いデータベース操作列に翻訳するので、大規模なオブジェクトを効率良く処理できる。

3. システム構成

本システムは次の3つの部分からなる。ユーザインタフェース管理部では、日本語やグラフィックス等による高度なインタフェースをサポートする。オブジェクト管理部では、オブジェクトを操作する枠組みを与え、プログラムインタフェースもサポートする。データ管理部では、拡張関係モデルにより2次記憶上にオブジェクトを格納する（図1）。

プログラムインタフェースを利用する時、次のような順序でプログラムをコンパイルする。まず、ユーザの記述したプログラムをプリコンパイラによりCプログラムに翻訳し、その結果をCコンパイラでコンパイルする。さらに、システムの提供する実行時サポートライブラリと結合して、実行プログラムができる（図2）。

プリコンパイラでは、ユーザプログラムを単語分割し、オブジェクト操作言語かどうか判定し、オブジェクト操作文についてはさらに解析し、最適化処理を施し、Cプログラムを生成する。途中、必要なオブジェクト情報を検索する。プログラムに定義がなく参照されるオブジェクトはデータベースより検索する（図3左）。

実行時サポートライブラリには、オブジェクトやメソッド等のハッシュ表を用いた管理やオブジェクト操作やデータベース中のオブジェクトの検索，削除，生成等のデータベース操作のためのプログラムがある（図3右）。

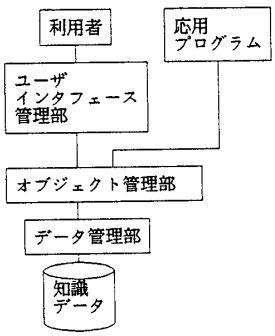

図1. システム構成

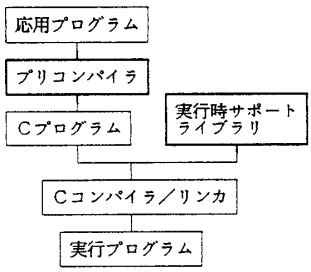

図2. プログラムのコンパイル

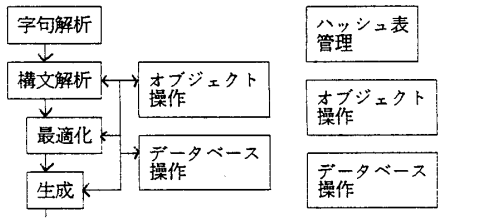

図3. モジュール構成

4. オブジェクトの格納方式

4.1. インスタンスの格納方式

各クラス毎に、各属性をフィールドとするようなテーブルを設ける。さらに、識別子属性にインデックスを張る。また、他の属性に

Implementation of Program Interface of the Object-Oriented Multi Media Knowledge Base System Jasmine

Fumio Suzuki　Hiroshi Ishikawa　Akifumi Makinouchi

FUJITSU LIMITED

対してユーザ定義のインデックスを張ることもできる。多値属性は非正規形テーブルに格納する（図4）。

他の格納方式として、正規形テーブルのものや全てのインスタンスに1つのスキーマを与えるものも考えれるが、本方式はスペース効率とアクセス効率の双方について優れている。

4.2. クラスの格納方式

クラスはクラス名をキーとするハッシュテーブルに格納する。ユーザ定義の属性は属性名と各ファセットをフィールドとする非正規形テーブルに格納するので、任意の属性を定義できる。各システム定義の属性はそれぞれ1つのフィールドに対応させ、アクセスの高速化を計った（図5）。

患者テーブル

#	名前	体重	身長	年齢	趣味
1	田中	NIL	172.0	28	読書 旅行

図4. インスタンス格納テーブル例

クラステーブル

クラス名	KB	上位		ユーザ定義属性				
		KB	クラス	属性名	KB	クラス	多値	if-needed
人間	医療	KERNEL	COMPOSITE	名前	KERNEL	STRING	NO	NIL

				体重	KERNEL	FLOAT	YES	float r;
								r=$self.身長;
								r=r-100.0;
								return(r);

図5. クラス格納テーブル例

5. 処理の高速化

オブジェクト操作を効率良く実行するために、応用プログラムをC言語プログラムに翻訳する。

5.1. 静的結合による属性操作とメソッド実行の高速化

従来のオブジェクト指向システムでは遺伝のための検索処理を実行時に動的に行なう。例えば、ある患者の体重を求めるとき、そのインスタンスの体重属性を探し、あればその属性値をとる。その値がニルならばデフォルト値やデモン等を検索して実行する。メッセージ送信においても、そのセレクタに対応するメソッドを検索して実行する（図6）。

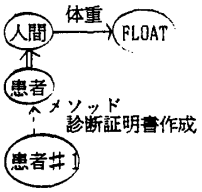

図6. 属性とメソッドの遺伝

本システムではプリコンパイル時に遺伝処理を施して、実行時の遺伝処理によるオーバーヘッドをなくした。すなわち、患者の体重は相対アドレスにより参照し、デフォルト値もプログラム中に展開する。デモン起動やメッセージ送信も対応するメソッド関数の呼出しに変換する。

また、デモン制御文やデモン制御パラメータを予めプリコンパイルするので、インタプリタのように実行時にデモン起動の有無等の判定をする必要がない。

5.2. オブジェクトの操作の高速化

オブジェクト管理部とデータ管理部とのインタラクションを少なくすることが重要である。そのために、利用時にはオブジェクトをデータベースより検索し、主記憶上にデータ構造として蓄える。以後は、そのアドレスをもとに直接アクセスしたり、システムの管理するハッシュ表を通して、識別子により高速にアクセスできる。また、データベース内での位置を管理するので、更新や削除を高速に行える。さらに、データベースからの第1回目の検索では識別子属性のインデックスを利用して高速に検索できる。

5.3. 問合せの最適化

次の3種類の最適化を行なう。

① 意味的最適化

モデルの意味に基づく最適化であり、カテゴリ属性を利用した検索テーブルの限定があげられる。

例えば、図7のように患者クラスが年齢により2つのクラスに分類でき、12才未満を小児、12才以上を大人とする。このような時、35才以上の患者は、35才以上の大人の患者と等価であり、検索テーブルを大人の患者に限定できる。

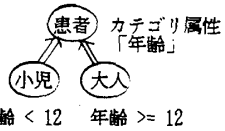

図7. 上位下位関係とカテゴリ属性

このように、カテゴリ属性に関する分類条件と検索条件をマッチングすることにより、検索テーブルの範囲を限定できる。

② 定性的最適化

データベース処理の最適化でよく言われる一般原則であり、(1)結合より選択を優先する，(2)射影の利用，(3)中間結果テーブルの利用等があげられる。これらの最適化は属性を辿る条件でかなりの効果を期待できる。

③ 物理的最適化

アクセスメソッドに関する最適化であり、インデックスの利用があげられる。クラス定義等でインデックスの指定をすれば、アクセス時にインデックスの利用指定がなくても、その利用効果が期待できれば、インデックスを利用するように翻訳する。

6. おわりに

以上のように、本システムは大規模な知識を効率良く格納し、さらに高速に処理できる。また、ユーザはオブジェクト指向のモデルで問題を直接表現できるので、プログラムの生産性も向上する。

謝辞

この研究は通産省工業技術院大型プロジェクト「電子計算機相互運用データベースシステムの研究開発」の一環として行なったものである。