5G−10

データベース指向OS XERO における複合オブジェクト管理

成田篤信　加藤和彦　益田隆司

東京大学　理学部　情報科学科

1．はじめに

従来のOS上でデータベース処理を行う上での最大の問題点は，ディスク装置に代表される永続的記憶装置の仮想化と管理の機構がDBMSとOSとの間で2重化し，結果として2重の永続記憶空間が存在している点である．XEROではこの問題に根本的に対処するため，永続記憶空間の再構成を行い，オペレーティングシステムの2次記憶管理モジュールが永続的複合オブジェクトの論理的な型管理を行う[2]．本稿では，2次記憶上での複合オブジェクトの管理法について述べる．

2．複合オブジェクト

XEROシステムでは，全ての永続データを，複数のオブジェクトが互いに参照し合う構造を持つ複合オブジェクトとして管理する[4]．複合オブジェクトは次のように定義される．全てのオブジェクトは、オブジェクトの物理的な位置によらないOIDを持ち，OIDのみによって，オブジェクトの値を参照することができる．また，オブジェクトは，atom型，tuple型，set型のいずれかの型を持ち，OIDのみによってそのオブジェクトの型も知ることができる．

（1）atom型

整数型，実数型，文字列型といったこれ以上分けることのできないデータ単位である．各基本型には、その型特有の演算子が用意されており，その型に属するオブジェクトの生成，削除，比較，更新等が行える．

（2）tuple型

属性名(Ai)と，OID(Ii)，の組の有限列を取る．

<A1:I1, A2:I2, ... , An:In>

tuple型オブジェクトに対しては、生成，削除のほか，属性名によって指定されるオブジェクトの参照，変更が行える．

（3）set型

同じ型を持つオブジェクトの集合であり，OID(Ii)の有限列で成立つ．

{I1, I2, ... , In}

set型オブジェクトに対しては，生成，削除のほか，要素の挿入，削除，検索が行える．

3．複合オブジェクトの2次記憶上での実現

複合オブジェクトの2次記憶上管理法の設計の際の問題点としては以下のような点があげられる．

- 複合オブジェクトの2次記憶への効率的な格納
- 可変長オブジェクトデータ構造の扱い
- オブジェクトの参照の効率化

このような問題に対し，今回の2次記憶管理システムは，次のように解決している．

- 複合オブジェクトを同じ型をもつオブジェクトごとにまとめることにより，複合オブジェクトを時間的空間的に効率よく管理する．
- オブジェクトのデータ管理にB^+-tree構造，ハッシュ構造などを取入れることにより，オブジェクトの性質に合ったデータ管理法を選ぶことを可能とする．
- オブジェクトの参照関係に基づき，永続的キャッシング[1][3]を行うことにより，2次記憶へのアクセス回数を減らす．

3．1　tuple型オブジェクトの管理

ある型に属するtuple型オブジェクトの管理は図1のように行なう．オブジェクトはB^+-tree形式でまとめられており，1次インデックス(primary index)によって順序付けられている．主インデックスにはクラスタリングを行なうキーを選ぶ．隣接するタプル同志は物理的にも隣接することになる．また，必要によって，2次インデックス(secondary index)を張ることもできる．

このように，tuple型オブジェクトが必ず固定長であることに着目し，従来の同一型レコードによって構成されるファイル管理法と同様な方法で，複合オブジェクト管理を行なえることになる．

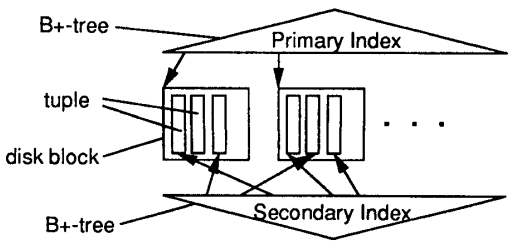

図1　tuple型オブジェクト管理法

Complex Object Management on the XERO Operating System
Atsunobu NARITA, Kazuhiko KATO, and Takashi MASUDA
Dept. of Information Science, Univ. of Tokyo.

3．2　set型オブジェクトの管理

set型オブジェクトの管理は図2のように行なう．tuple型の場合と同様，同じ型のオブジェクトを要素にもつもの(set型オブジェクト)をB^+-treeによりまとめて管理する．また，set型オブジェクト自身も，要素となるオブジェクトのOIDをB^+-treeによって管理している．

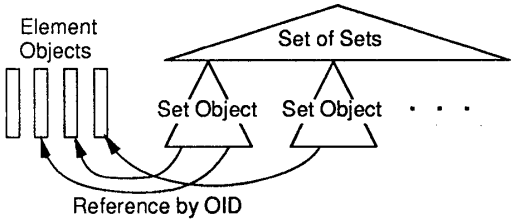

図2　set型オブジェクト管理

3．3　atom型オブジェクトの管理

atom型オブジェクトは，値そのものが固有のものであるため，値をそのままOIDとして扱うことにする．つまり，tuple型，set型オブジェクトの要素にatom型オブジェクトが与えられた場合は，atom型の値そのものをOIDのかわりに代入しておく．ユーザによってatom型オブジェクトの生成が要求された場合は，atom型オブジェクト1つからなるtuple型によって実現できる．

4．永続的キャッシング技術の利用

2つのオブジェクト間に参照関係がある場合，参照している側に，参照先のオブジェクトの値の一部のコピーをもち，参照先オブジェクトの書き換えが起きていない場合はそのコピーを参照することにより，2次記憶アクセス回数を減じる方法を提案している．この方法を永続的キャッシングと呼ぶ．また，更新の伝播を遅延させることで，更新の伝播に伴うオーバーヘッドを減じている．ここでは，この永続キャッシング技術を，複合オブジェクト管理に活用する方法について述べる．

4．1　OIDから物理アドレスへの変換への利用

永続的キャッシングを，OIDからオブジェクト値への変換だけでなく，OIDから物理アドレスへの変換に用いることができる．XEROでは，OIDは物理的な位置に対して独立であるために，オブジェクトの値の物理アドレスへの変換を行なわなければならない．物理アドレスをキャッシングしておくことにより，OIDから高速に実データを得ることができる．

4．2　tuple型オブジェクトでの利用

tuple型オブジェクトは，通常はOIDの列によって構成されているが，OID列だけではなく，各参照について，タプル内にキャッシュ格納領域を設け，参照先のオブジェクトの値のコピーをこの領域に保持する（図3）．

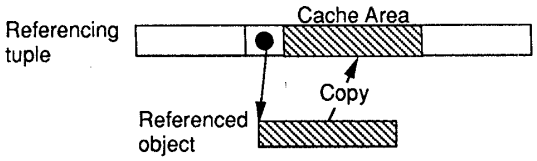

図3 tuple型オブジェクトにおける永続キャッシング

4．3　set型オブジェクトでの利用

set型オブジェクトの構成要素は，論理的にはOIDであるが，永続的キャッシングにより，要素の値のコピーを保持することができる（図4）．

永続的キャッシュの更新の伝播を遅延させず，実体とキャッシュの間の一貫性を常に保持する機構を採用すると，set型オブジェクトの要素に主インデックスを付け，クラスタリングを行なうことが可能となる．

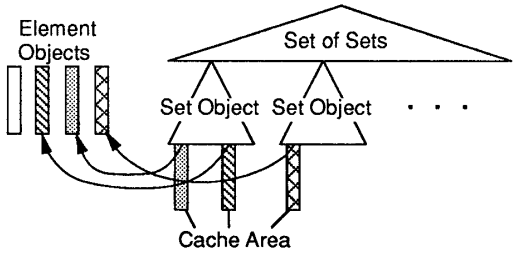

図4　set型オブジェクトにおける永続キャッシング

5．おわりに

現在，本稿で述べた永続オブジェクトの管理の実現を，Sony NEWS上で進めている．また，本稿で述べた方式を分散環境へ拡張するために，ネットワークにまたがるオブジェクト参照，オブジェクト転送のプロトコルなどの問題について検討を行っている．

参考文献

[1]Kato, K. and T. Masuda, "Persistent Caching: An Implementation Technique for Complex Objects with Object Identity," Univ of Tokyo, Dept. Information Science Technical Report 89-21, 1989.

[2]加藤，猪原，脇田，益田，"データベース処理を指向した分散オペレーティング・システムXEROの構想，" 電子情報通信学会コンピュータシステム研究会CPSY-89-29, 1989.

[3]登内，加藤，益田，"データベース指向OS XERO における永続的キャッシング技術，" 情報処理学会第40回全国大会講演論文集，1990.

[4]脇田，加藤，益田，"データベース処理を指向した分散OS XEROの永続オブジェクト管理，" 情報処理学会第39回全国大会講演論文集，1989.