永続データのためのB-木を用いた推論方式の提案

3U－5

西山 智　小野 智弘　小花 貞夫
株式会社 KDD研究所

1. はじめに

網管理等のエキスパートシステムは，網のトラフィック情報等大量のデータを用いて推論する必要がある．このようなエキスパートシステムの構築を容易とするために，データをデータベース上に永続的に格納し，そのデータ (永続データ) を主記憶上のデータと同様に推論可能とする推論方式が提案されている[1][2][3]．しかしながらこれらの方式には，(1) 主記憶用の推論アルゴリズムをほぼそのままデータベース上で実現した方式は，使用する推論用の索引 (α メモリ、β メモリ) へのアクセス回数が多く，データベース上では効率的ではない，(2) データベース用の索引と推論用の索引を重複して管理する必要がある，という問題があった．そこで，本稿では，永続データに対しては α メモリ等の専用索引を作成せず，データベースに既に存在する B-木による索引を推論に使用する推論方式を提案し，その評価を示す．

2. 永続データのための推論方式の問題点

2.1 従来の永続データのための推論方式

これまでに，データベース上の永続データを扱うための推論方式がいくつか提案されている (表1参照)．

- DBRete[1]は RETE アルゴリズムに必要な全てのデータ構造 (α メモリ、β メモリ) を RDB 上のレコードとして格納し，RETE に必要なメモリ操作を SQL を用いて行う．
- DATEX[2]は専用 DBMS 上に FI(Filtered Index) と呼ぶ索引を構築し，それを用いて最良優先探索に基づく推論を行う．
- 実時間推論シェル[3]は，専用 DBMS 上に α メモリを持ち，それを用いて最良優先探索に基づく推論を行う．

2.2 従来方式の問題点

(1) α メモリや β メモリは，主記憶上でのアクセスを想定して参照や更新が索引の1要素毎に行われるため，アクセスコストの高いデータベース上では効率的ではない．

(2) 一般にデータベースは，外部からの更新，検索用に B-木やハッシュ表等の索引を提供する．従来の推論方式はいずれも推論専用の索引を持つため，データベースの索引と重複して維持，更新する必要があり効率的ではない．

表 1: 従来方式の比較

	使用 DB	索引	マッチング方式
DBRete	RDBMS	α メモリ β メモリ	RETE
DATEX	専用 DB	FI	最良優先探索
実時間推論シェル	専用 DB	α メモリ	最良優先探索

上記の2点の問題点を解決するためには専用の索引を作成せずデータベースの既存の索引を利用する推論方式が必要となる．ここで，推論における検索条件には一致以外の条件も含まれる事からハッシュ表は適用できない．そこで次節では，データベースに既に作成されている B-木による索引 (正確には B+-木など，一致以外の演算も提供する B-木．以下では単に B-木と呼ぶ) を利用する推論方式を提案する．

3. B-木を用いた推論方式の提案

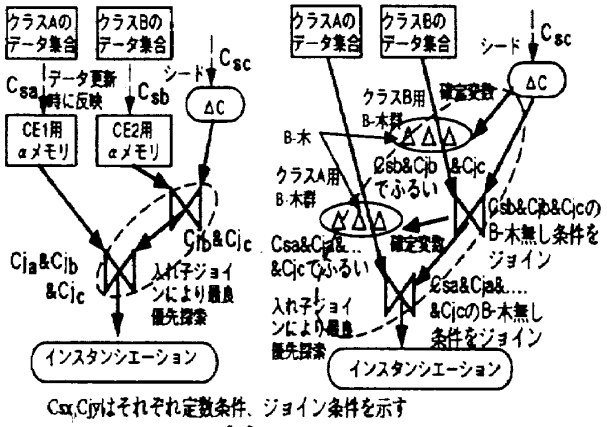
3.1 基本方式

提案方式は，実時間推論シェル[3]に実装した最良優先探索に基づく推論方式の拡張版である．拡張前の方式は，α メモリ間での入れ子ジョインによりマッチング処理を行う (図 1(a) 参照)．提案方式は，永続データについては α メモリを使用せず，代わりに永続データの個々の属性に関する B-木の索引を使用する (図 1(b) 参照)．図 2は図 1で例示したルールを示しており，図中の C_{sx}，C_{jy}はそれぞれ条件節での定数条件，ジョイン条件 (変数を含む条件) を示している．提案方式では，探索の起点となったデータ (シード，ここでは Δ C) から確定している変数値を用いてジョイン条件 C_{jb}&C_{jc}の確定部分を定数条件化する．この定数条件と，CE2 に対するクラス B への定数条件 C_{sb}を元に，適用可能な B-木 (図中の三角形) を用いてクラス B のデータをふるう．さらに残ったデータ集合に対して，実際のデータの内容を参照して B-木が設定されていなかった定数条件によりふるう．最後に残ったデータ集合を，最良優先探索を実現するために競合解消戦略 (通常は新しいもの順) に基づきソートする．その個々のデータに対してさらに内側の入れ子ジョイン (図 1(b) ではクラス A に対するジョイン) を行う．

"Matching Algorithm for Persistent Data using B-Tree" by Satoshi NISHIYAMA, Chihiro ONO and Sadao OBANA, KDD R&D Labs., Inc.

表 2: 従来の推論方式との性能比較

	Tourney				Waltz			
	DB ページ参照数	バッファヒット率 (%)	実行時間 (秒)	相対性能	DB ページ参照数	バッファヒット率 (%)	実行時間 (秒)	相対性能
従来方式 [3]	3.14×10^5	99.9	32.1	1.00	5.55×10^5	99.8	27.7	1.00
提案方式 (β' メモリなし)	4.06×10^4	79.5	39.7	0.70	5.40×10^4	98.8	13.0	2.13
提案方式	2.90×10^4	98.0	16.1	1.99	5.30×10^4	98.8	11.9	2.33



Csx,Cjyはそれぞれ定数条件、ジョイン条件を示す

(a) 従来の推論方式の例[3] (b) 提案方式 (基本方式)

図 1: 従来方式と提案方式の違い

```
rule join_ABC {
  &CE1  (A  Csa  Cja)
  &CE2  (B  Csb  Cjb)
  &CE3  (C  Csc  Cjc)
  --> ....  }
```

図 2: 図 1 で用いるルール

3.2 β' メモリ：検索の高速化

3.1節で述べた基本方式は, 入れ子ループにより同じ条件に基づく B-木検索を繰り返し行う. その繰り返し検索を避け高速化を図るために, 確定した定数条件とその結果のデータ集合をバッファに格納しておき, 同一の条件に対する検索要求が発生した場合, B-木を検索せずバッファから結果を返す. このバッファを β' メモリと呼ぶ. β' メモリと呼ぶのは, β メモリのある特定条件下でのサブセットと等価であるためである. β' メモリは, 対応するルールのいずれかの条件節に示されたクラスのデータが追加, 削除されたタイミングでその内容を削除する必要がある.

また大量のデータをアクセスすることを想定すると β' メモリの量も多くなる. β' メモリの量を一定に制限するために β' メモリを LRU アルゴリズムにより管理することとする.

4. 評価

既に開発済の実時間推論シェルに提案方式を実装し, 2つのサンプルプログラムを用いて従来の α メモリを用いた推論方式[3]と比較した. Tourney はブリッジの組合せ作成プログラムで 10〜15 のクラス間のジョインを必要とする高コストルールが含まれている. また Waltz は配線プログラムであり, 高コストルールはない. これらについて, 推論制御用ではないデータ (Tourney ではデータ量が多くなる対戦済参加者の組 (alreadyPlayed), Waltz は配線の元となるデータ (line, edge)) を永続データ化した. また, 提案方式については β' メモリを使用しない場合も測定した. 表 2に結果を示す. 表において, DB ページ参照数は DB のページ (2KB) のバッファ前のアクセス数を示す. DB に設けた 3MB のバッファにより次のカラムに示すヒット率が得られたため, 実際にディスクにアクセスにいった回数は参照数× (1-ヒット率) である. また実行時間は Sun SS1000 (Solaris2.5, CPU:50MHz) で測定した値, 相対性能は従来方式の性能を 1 とした値である.

表 2から, 提案方式は両プログラムにおいて従来方式のほぼ 2 倍の性能であった. これは, α メモリを参照しない事により DB ページの参照数が 1 桁以上減少した事が大きい. β' メモリを使用しない場合, Waltz ではほぼ同等の性能であったが Tourney では従来方式以上に性能が悪化した. これはバッファのヒット率が低下しディスクアクセスが多数発生した事が原因であり, 高コストルールのジョインで, ランダムにデータ内容を参照したためと考えられる. このことから, 高コストルールを含むプログラムでは β' メモリは有効であるといえる.

5. おわりに

本稿では, 永続データを推論する際に, 従来の α メモリ, β メモリや FI 等の専用の索引ではなくデータベース上の索引である B-木をそのまま推論に利用する推論方式を提案した. 提案方式は, 推論の高速化が図れる上にデータベース用と推論用の両方の索引を重複して管理する必要がない. 評価の結果, α メモリを使用する従来方式と比較してほぼ 2 倍の性能を得られた. 最後に日頃御指導頂く (株)KDD 研究所 村谷拓郎所長, 鈴木健二副所長に感謝します.

参考文献

[1] T. Sellis et. al.: "Coupling Production Systems and Database Systems: A Homogeneous Approach," IEEE Trans. on KDE, Vol.5, No.2, (1993).

[2] D. A. Brant and D. P. Miranker: "Index Support for Rule Activation," Proc. of the 1993 ACM SIGMOD, (1993).

[3] 西山 他: "永続データをサポートする実時間推論シェル," 情処研資 DBS 116-34, pp.55-62, (1998).