推論機能と関係データベースの融合

1M-6

―――ハッシュとソートによる検索―――

大森　匡　田中　克己　土井　晃一　吉田　敦　田中　英彦
東京大学工学部

1．はじめに

単一化可能パタンの検索は，要求パタン・対象パタン共に複数個存在する場合，組合せが増大し隘路となりやすい。本小論文では　上記の場合を単一化を含んだ結合演算とみなし，従来のハッシング技法に加えてソートを用いた単一化可能パタンの検索方法について述べる〔1〕。

2．検索方法

以下では，原子論理式を属性値にとった関係A，Bを考え，この間の単一化を伴った自然結合（単一化結合と呼ぶ）を例にとる（図1）。

但し，各関係はハッシィング等によって述語／アリティ（引数の数）の等しいタプル集合に予め分割するとし，A，B共　述語，アリティ一定のタプル集合と仮定する。

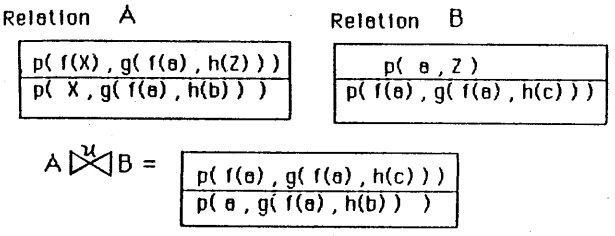

図1．単一化結合の例

2．1　ハッシング

述語のアリティに応じて木構造（ハッシュ木と呼ぶ）をあらかじめ設定し，原子論理式をこれに合わせて展開する（図2）。

そして関係A，Bの各タプルxについて，ハッシュ木上の各ノード　n0，n1．．．に対応したxのシンボルa0，a1．．．を取り，ハッシュベクトル

H = [h(a0) , h(a1) , . . .] を与えた後A，B各々をハッシュバケツ $\{H_A^i\}$，$\{H_B^j\}$ に分類する。

但し，$h(a) = \begin{cases} 0 & (a\text{が変数}) \\ \mathrm{int}(a) \bmod N + 1 & (\text{その他}) \end{cases}$

とし，　また，niに対応するシンボルがxに存在しないときはh（a）＝0　とする。　例えば，図2のハッシュ木上で元xが　p(f(a) , g(X , Y))　なら

H = [h(p) , h(f) , h(g) , h(a) , 0 , 0 , 0] となる。

従って，ハッシュバケツ H_A^i のタプルと H_B^j のタプルとが単一化可能であれば H_A^i と H_B^j はハッシュベクトルの内積 $H_A^i * H_B^j$ の非零ノードでハッシュ値が等しくなる。これをハッシュベクトルの単一化とみなすと，全体の単一化アルゴリズムは図3のようになる。

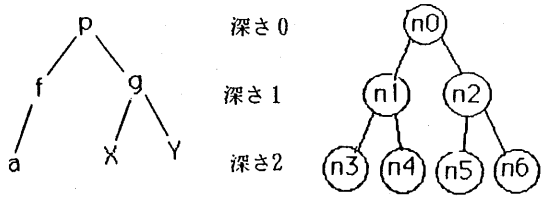

図2．アリティ＝2のハッシュ木にあわせた
p(f(a) , g(X , Y)) の展開例

```
Begin
    partition relation A & B into hash-buckets
    {H_A^i} & {H_B^j}

    for all H_A^i , H_B^j
        if H_A^i is unifiable with H_B^j
        then
    (*) ----  unify-try for each combination of
              members in H_A^i & H_B^j
        fi
    rof
End
```

図3．ハッシングによる単一化アルゴリズム

関係A　のハッシュバケツをα個，各バケツの濃度を C_A^1，C_A^2，．．．C_A^α，関係Bのそれを各々β個，C_B^1，．．C_B^β　とすると

（1）計算量＝　$K1 \cdot \alpha \cdot \beta + K2 \cdot \sum_{i,j} C_A^i \cdot C_B^j$

（但し，K1 とK2 は定数：i，j は H_A^i と H_B^j とが単一化可能なもの。）

単一化可能なハッシュバケツの探索はハードウエア化できる。また，ハッシュ木はタプルの分布に応じて分散の大きいノードのみ取ることが望ましい。

2．2　ソーティング

ハッシュ木と同様にしてソート木を設定し，各タプルxに対してソートベクトル　S = (s(n0) , s(n1) , . . .) を与える。但し，s（ni）＝0（niが変数）　1（その他　）とする。

ハッシュベクトルとソートベクトルを共用すると，ソートベクトルはハッシュバケツ毎に与えられる。そして，単一化可能なハッシュバケツ H_A^i，H_B^j に対して，内積 $Ht = H_A^i * H_B^j$ をもとめHt中0でないノード列をキーとしてH_A^i，H_B^j のタプルをマージソートする。

ソート列中のキー同値類は ソート木中での単一化可能なタプル集合である。故に図3での（＊）の部分は下のようにかわる。

```
Begin
        mergesort all members in
        buckets H_A^i & H_B^j
(*)-->
        unify-try for each combination
        of members in all KEY equi-
        valent classes in sorted stream
End
```

従って

$$\text{計算量} = K_1 \cdot \alpha \cdot \beta + K_2 \sum_{i,j} (C_A^i + C_B^j) \log (C_A^i + C_B^j)$$

（ 定数は（1）と同じ。）

例えば，

H = [[1,2,3,4,0,0,0], p(f(a), g(X, Y))]
H = [[1,2,3,0,0,2,5], p(f(X), g(f(a), h(Z)))]

とすると，H＝〔1，4，9，0，0，0，0〕となり，ソート木中のノード列〔n0，n1，n2〕をキーとしてソートすることになる。

3． 評価

ここではハッシュ木とソート木を共用し木の深さをかえて，ソート時にキーとなるノード数と単一化試行数における絞り込み効果との関係を調べる。

ソート用の想定木は図2上で 1）深さ1迄の3ノード
2）深さ2迄の7ノード
の2通りとし，サンプルは 関係A，B共，この木の各ノードについて n0は述語pで一定，n1～n6は変数1，関数子2，定数1の比率で無作為に組み合わせた原子論理式 100個とする。

このとき，関係A，Bのタプル集合がソート木の各ノードに対応して持つシンボル分布を考えると，上の1），2）は各々，深さ2迄7ノードのソート木を予め設定したとき

1） この分布が，設定したソート木のノード中深さ1では特定のシンボルに偏っていないが深さ2では偏っているため，深さ2のノードがソートの対象となっても絞り込み効果が期待でき無い場合。

2） 全ノード上でシンボル分布が特定の定数または変数に偏っていない場合。

を示している。

表1．ソート木が
1）3ノード 2）7ノード の時の
全組合せについて単一化を試行する際を基準とした
a）ソート後の単一化試行数の比率
b）単一化可能なハッシュバケツの組合せ数

	1) 3 nodes	2) 7 nodes
a) unification try (%)	17%	12%
b) unifiable hash-buckets combinations (pieces)	4^2	16^2

評価パラメタは

・1ノードあたりのハッシュ値のサイズN
・想定木で選ぶノード数
・単一化可能なハッシュバケツの組合せ数

の3つである。

今回はN＝1とし，ノード数を変えた時の単一化可能なバケツの組合せ数も調べた（表1）。

その結果，1）では図2のノードn1，n2共に非変数なるハッシュバケツにタプルが集中するため，絞り込み効果は2）と大差なくなっている。

故に，ソート木を設定する際は，次の3点が必要である。

① できるだけ深さが小さくて非変数シンボルになる確率が高く，且つ特定のシンボルに偏っていないノードが2～3個含む様にすべき。

② 変数になる確率の高いノードと，その子ノードまで含んでも絞り込み効果は小さい。

③ 一方，想定木のノード数nを増すときには単一化可能なハッシュバケツの組合せが $\mathcal{O}(2^n)$ となるため，ハードウエア化等による高速化が必要である。

4．おわりに

本小論文では，ハッシュとソートを用いた単一化可能パタンの検索方法について述べ，簡単な評価を行った。

今後は，タプルの分布に応じたハッシュ木の動的な最適設定，単一化可能なハッシュバケツ選択の高速化 等の検討が必要である。その際には，ソート用のキーをつくるオーバーヘッドとの兼ね合いでハッシュ値のサイズを大きくし1バケツあたりの濃度を下げてソートを省くことも考えるべきである。

〔参考文献〕

1． Relatonal Algebra Machine Based on Hash and Sort : GRACE
M.Kitsuregawa 東京大学学位論文 '82