スーパーインポーズドコードを用いた構造体の検索方式

森田　幸伯　　和田　光教*　伊藤　英則

（財）新世代コンピュータ技術開発機構　*（株）沖電気工業

1．はじめに

最近エキスパートシステムや知識ベースなどでは知識を表現するものとして、変数を持った構造体が多く使用されている。このように表現された知識の扱いには、パターンマッチ、特に単一化処理が重要な役割を演じている。今後、扱う知識の量が増大すると伴に、１次記憶や２次記憶におかれた知識の高速な検索技術がますます重要となってくる。

一方重ね合わせ符号(superimposed code) 法を用いた検索技術は、マルチキーファイルの構成やシグニチャファイルを用いたファイルアクセスの高速技法などに適用されている(1)。また、エディタなどのサーチコマンド等にも応用されてきている。

本稿では、重ね合わせ符号法を構造体の一種である項に対して適用させる方式について述べる。これは、ある項と単一化可能な項を含むオブジェクト(レコード、ページ等)に対するインデックス検索を可能にするものである。

2．定数に対するＳＣＷ方式(1)

重ね合わせ符号による検索方式(本稿ではＳＣＷ方式とよぶ)は、レコードに対してsuperimposed code word(ＳＣＷと記す)なる一種のインデックスを用いて、大容量のレコードを検索する時間を短縮しようとするものである。

R_iをファイルＦのレコードとする。R_iには幾つかのキーワードが付けられている。各キーワードはある関数（ハッシュ関数)によって２進符号語(ＢＣＷ)に写像される。あるレコードR_iに対してその全てのキーワードのＢＣＷのビットごとのＯＲをとったものをＳＣＷと呼び、S_iと表記する。ファイルＦの全てのレコードについて対応するＳＣＷを計算し１つのファイルとし、これをＳＣＷファイルＳと呼ぶ。

今、ファイルＦに対して幾つかの指定したキーワードをもつレコードを検索する問題を考える。この検索のために指定されたキーワードを上記ＳＣＷを生成したのと同じ関数を用いてＢＣＷを生成しＯＲをとってＳＣＷを生成する。これをキュアリマスクと呼びＱで表わす。ＱとあるＳＣＷ・S_iとが以下の条件(1)を満足しないならば、対応するレコードR_iは、はじめの検索条件(キュアリ)を満足しない。

$$Q \wedge S_i = Q \qquad (1)$$

ファイルＳから(1)を満足するS_iの集合を選び出し、対応するR_iに対してのみキュアリを満足するかどうか調べることにより探索空間を減らすことができる。勿論条件(1)を調べる処理は、オーバーヘッドとなる。しかし、この処理は比較的簡単な処理であるため、ＳＣＷをうまく設計すれば、このオーバーヘッドは、十分小さくできる。

3．ＳＣＷ方式を用いた項の検索方式

前節のキーワードを変数を含む一種の構造体である項に拡張する。それにより、キュアリはある項と単一化可能な項をキーワードとしてもつレコード(オブジェクト)を求めるものとなる。(本節では議論を簡単にするために各レコードのキーワードもキュアリのキーワードも１つとする。)

ＢＣＷを項に拡張するために、次の２点で拡張を行う。

A．項が構造体である(木構造とみなせる)ことより、木構造の各ノードに対してそのノードの値(関数記号または変数記号)から２進符号語の一部分への関数(特定ビットを０に固定する)を用意し、項のＢＣＷとしては、それらの値のＯＲをとったものとする。

B．項が変数を含むことにより、キーワードからＢＣＷを生成する関数を通常のデータに対する関数とキュアリに対する関数を別の関数とする。

結局、項tにたいするＢＣＷは、tを木構造とみなしたときのノードの集合をＮｏｄｅとし、ノードｎのデータ用関数をｈｎ$_n$、ノードにある値(関数記号or変数記号)をｎａｍｅ$_n$とすると

$$BCW_t = ht(t) = \vee_{n \in Node} hn_n(name_n)$$

となる。同様にキュアリに現われる項tに対しては、ノードｎのキュアリ用関数をｈｎ$_n'$とすると、

$$BCW_t = ht'(t) = \vee_{n \in Node} hn_n'(name_n)$$

となる。

各ノードのデータ用関数またはキュアリ用関数ｈｎに対し、全ての関数記号および変数記号ｘに対してある$\{a_i \in \{0,1\}\}$が存在して、$hn(x) = \Sigma_{i \in E} a_i 2^{i-1}$ となる最小の整数の集合ＥをＢＩＴ(ｈｎ)で表わす。

各のノードのデータ用関数およびキュアリ用関数に対し

Structure retrieval via the method of superimposed codes
Yukihiro MORITA[1], Mitsunori WADA[2], Hidenori ITOH[1]
[1] ICOT, [2] Oki Electric Industry Co., Ltd.

て以下の条件を考える。

［条件１］

親ノードｎを定めると、親ノードのデータ用およびキュアリ用の２進符号への関数ｈｎ，ｈｎ' に対し、ｍ個の子ノードのデータ用およびキュアリ用の２進符号への関数 hn_i，hn_i' （$1 \leq i \leq m$）がそれぞれ一意に定まり、

$$BIT(hn_i) \subseteq BIT(hn)$$
$$BIT(hn_i') \subseteq BIT(hn')$$

［条件２］

全てのノードの２進符号への関数は、以下を満足する。

任意の変数記号ｘに対して、Ｅ＝ＢＩＴ(ｈｎ)とすると、

$$hn(x) = \Sigma_{i \in E} 2^{i-1}$$
$$hn'(x) = 0$$

任意の関数記号ｘに対して、

$$hn(x) = hn'(x)$$

［補題］

条件１，２を満足する項ｔのデータ用ハッシュ関数ｈｔ(ｔ)，及びキュアリ用ハッシュ関数ｈｔ'(ｔ)は、

$$ht'(t\ \ ) \wedge ht(t\theta) = ht'(t\ \ )$$
$$ht'(t\theta) \wedge ht(t\ \ ) = ht'(t\theta)$$

をみたす。ただし、θは置換(substitution)。

［定理］

条件１，２を満足する項ｔのデータ用ハッシュ関数ｈｔ(ｔ)，及びキュアリ用ハッシュ関数ｈｔ'(ｔ)は、

t_1，t_2が単一化可能ならば

$$ht'(t_1) \wedge ht(t_2) = ht'(t_1) \quad (2)$$

をみたす。

［証明］

t_1，t_2が単一化可能なことよりある置換θが存在して $t_1\theta = t_2\theta$ かつこれらの項が変数を含まないようにすることができる。

補題より

$$ht'(t_1) \wedge ht(t_1\theta) = ht'(t_1\ \ )$$
$$ht'(t_1\theta) \wedge ht(t_1) = ht'(t_1\theta)$$

$t_1\theta$に変数を含まないことから、

$$ht(t_1\theta) = ht'(t_1\theta)$$

これらより

$$ht'(t_1) \wedge ht(t_1\theta) = ht'(t_1)$$
$$ht'(t_1) \wedge ht'(t_2\theta) = ht'(t_1)$$
$$ht'(t_1) \wedge ht'(t_2\theta) \wedge ht(t_2) = ht'(t_1)$$
$$ht'(t_1) \wedge ht(t_1\theta) \wedge ht(t_2) = ht'(t_1)$$
$$ht'(t_1) \wedge ht(t_2\ \ ) = ht'(t_1) \quad \square$$

本稿では定理２の(2)を満足するt_2をt_1の疑似単一化可能項と呼ぶ。

４．簡単な評価と他方式との比較

単一化処理をおこなう前にその可能性のある項の組を選び出す方式は、変数のバインディングを無視する方法[2]、項を文字列で表わし変数の前までを比較する方法[3]、構造を限定しハッシュ関数を用いて符号化して調べる方法[4]等幾つか提案されている。

表１は、幾つかのサンプルデータに対する疑似単一化項と単一化可能項の割合を示している。ＢＣＷは２６ビットとし、条件１についてはｎ引数関数に対してＢＣＷのビットをｎ＋１等分して各子ノードに割り当てた。データは１引数のものから４引数のものまで用意し、各引数は３種の定数または変数である。

これらの例はハッシュ関数に衝突の無い場合であり[2]や[4]の方式と同程度の割合であると思われる。[3]に対しては平均７割位低くなっている。衝突が多くなってくると[2],[4]よりも単一化可能でない疑似単一化項を選び出す確率が高くなると思われる。しかし、本方式は[2]よりも簡単な処理（ビット演算のみ）で疑似単一化可能項を選択でき、[4]のように構造に対して制限を必要としない。また、複数の項のどれかと単一化可能であるという検索や複数のキュアリに対しては有効であると思われる。

表１．疑似単一化可能項と単一化可能項の比率

引数の数	1	2	3	4
疑似単一化可能項の比率	1.000	1.094	1.317	1.667

５．おわりに

重ね合わせ符号法が項に対する単一化可能条件に応用できることを示した。３節ではレコード(オブジェクト)とキュアリに含まれる項の数を１つとしたが、通常のＳＣＷと同様に３節の項に対するＢＣＷを重ね合わせて使用することも出来る。また、本方式の有効な具体的な応用例およびそのときのハッシュ関数の選びかた、特に条件１の各ノードの関数の定めかた等も今後の課題である。

［参考文献］

[1] Roberts, C. S., "Partial-Match Retrieval via the Method of superimposed Codes", Proc. of IEEE. Vol.67, No.12, pp.1624-1642, Dec 1979.

[2] Sabbarel, G. B., "Unification for A Prolog Database Machine" 2nd Logic Programming Conference 1984.

[3] Morita, Y. et al."Retrieval-By-Unification Operation on a Relational Knowledge Base Model", Proc. of the 12th Int'l Conference on VLDB 1986.

[4] 大森　他「推論機能と関係データベースの融合ーハッシュとソートによる検索」第32回情処全大, 1N-6,1986.