ハッシュ・リストによる集合の表現と演算*

3C－1

松崎 友保 †雲下 雅道‡　中村 克彦¶
東京電機大学大学院理工学研究科§　東京電機大学理工学部‖

1 はじめに

集合演算を高速で行うためのデータ構造とアルゴリズムの実現は，広範囲の情報処理において必要とされるもっとも基本的な問題である。この問題に対してこれまでに，ビット・テーブルを用いた方法，および2分探索木などの辞書のための各種のデータ構造が知られている。しかし，これらの方法はいずれも部分集合，和集合，共通集合などの基本演算に対して，集合のサイズ以上のオーダーの計算時間を必要とする。

本報告では，ハッシュ・リストによって辞書のためのデータ構造であるtrieを表すことによって，高速の集合演算を可能にした方式を述べる。ハッシュ・リストは後藤英一氏 [1] によってH-Lispのために導入され，その後われわれのH-Prolog [2] で使用された。

2 ハッシュ・リスト

ハッシュ・リストは同じリストが，かならずハッシュ記憶領域内の同じ場所に格納されるようなLisp型リストのためのデータ構造である。そのため，ハッシュ・リストへのポインタは一意に決定される。例えば，図1の2つのリスト構造は両方ともS-式[[b,c],a,b,c]を表しているが，一般にLispやprologでは一つのS-式に対して各種のリスト構造が作られる。しかし，ハッシュ・リストはかならず(a)の形式となる。

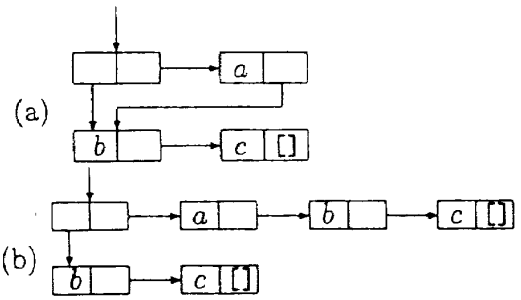

図 1: $[[b,c],a,b,c]$ のリスト構造

3 trie

trieは，次の条件を満足するラベル付2進木である．

1. 終端は名前(atom)のラベルをもつ。各名前にはアトムのビット列（実際はアトムのポインタである）が対応している。
2. 深さkの非終端節点における左の部分木はコードのk番目ビットが0である名前を含み，その右の部分木はk番目ビットが1である名前を含んでいる。

trieの例を図2に示す。ここで重要なのはtrieの木構造は名前の集合によって一意的に決定されることである。したがって，trieは特にハッシュ・リストを用いて表すのに適している。

4 H-Prolog上の実現

われわれはH-Prolog上でtrieを表現し，集合演算のプログラムを作成した。

次のプログラムで定義される述語 in/2 は，$\mathtt{in}(A, S^*)$ の形の目標が所属関係 $A \in S$ を意味する。ここで，S^* は集合 S を表すtrieである。

```
in(El,S):-
    code(El,Code),
```

---

*A Data-Structure for Sets Based on Hash List
†Tomoyasu Matuzaki
‡Masamiti Kumosita
§Tokyo Denki University Graduate School of Science and Engineering
¶Katsuhiko Nakamura
‖Tokyo Denki University The Factory of Science and Engineering

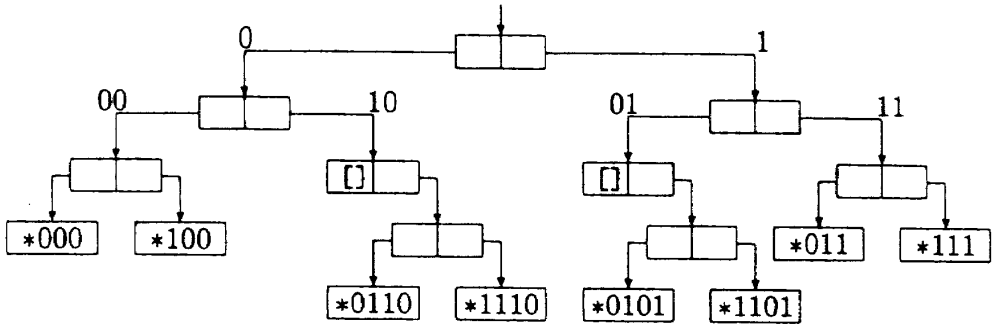

図 2: trie の例

```
    in1(0,S,El,Code).
in1(N,El,El,Code):- !.
in1(N,[L|_],El,Code):-
    bit(N,Code,0), !,
    N1 is N+1, in1(N1,L,El,Code).
in1(N,[_|R],El,Code):-
     N1 is N+1, in1(N1,R,El,Code).
```

述語 in1($N, S, El, Code$) は，Elが要素Sを含む深さNの部分木であることを意味している。また，$Code$はElのビット列である。このプログラムの計算時間は $O(\log|S|)$ である。

以下のプログラムは，目標 subset(S^*, T^*) が $S \subseteq T$ を意味するような述語 subset/2 を定義している。

```
subset(S,T):- subset1(0,S,T).
subset1(_,S,S):- !.
subset1(_,[],T):- !.
subset1(N,S,T):-  !,
   code(S,Code0), in1(N,T,S,Code0).
subset1(N,[A|B],[C|D]):- N1 is N+1,
   subset1(N1,A,C), subset1(N1,B,D).
```

$|T-S|=1$ の場合，計算時間は $O(log|T|)$ で与えられる。したがって，このプログラムの計算時間は

$$O(|T-S|\log|T|),$$

となる。すなわち，2つの入力集合が類似したものである（$|T-S|$ が小さい）場合に計算速度は速くなる。

一方，任意の集合 S に対して計算時間は $O(|S|)$ であるから，一般の計算時間は

$$O(\min(|S|, |S-T|\log|T|)).$$

によって与えられる。

和集合$S \cup T$および共通集合$S \cap T$の演算に対して，上記と同様に trie を用いることによって計算時間が

$$O(\min(|S \cup T|, (S \cup T - S \cap T)\log|S \cup T|)).$$

で与えられるプログラムを作成できる。

5 結び

この報告では，ハッシュ・リストと trie を用いた高速集合演算について述べた。今後の課題として，この方式を類似した集合を効率良く探索する方法に応用すること，および，これらの高速集合演算の応用などがある。

参考文献

[1] Eiichi Goto, Monocopy and associative algotrithms in extended LISP, *Technical Report of Information Sience Laboratory, the University of Tokyo*, 1974.

[2] Katuhiko Nakaura, Associative concurrent evaluation of Logic Programings, Jour.of logic programming, pp.285-295, 1984.