4B-7　連想メモリを利用した高速単一化アルゴリズム

大久保 雅且　高木 直史　安浦 寛人　矢島 脩三
(京都大学工学部)

1．はじめに

単一化(unification)は、与えられた2つの項のMGU(Most General Unifier)を求める操作で、Prolog等の論理型言語の基本操作として重要な演算の一つである。単一化は、項を項グラフによって表すことにより、項グラフ上の節点の同値分割と見なすことができ、UNIONとFINDという集合に対する2つの操作によって処理することができる[1]。これらの操作は、単一化の他、minimum spanning treeを求める問題や、決定性有限オートマトンの等価性判定、switching level simulation等、幅広い応用が考えられる[2][3]。しかし、UNION-FIND問題をソフトウェアによって効率良く処理するためには、複雑なデータ構造と、それに対する操作を必要とする。

本稿では、UNION-FIND問題を高速に処理するために、連想メモリ(CAM)を利用する方法を提案し、また、この方法を単一化に応用した際の処理時間について考察する。

2．UNION-FIND問題のCAMによる処理

集合に対する操作として、次の2つを考える。

FIND(i)…要素iが属する集合の検索

UNION(i,j)…要素i，要素jが属する集合の併合
但し、異なる2つの集合は共通の要素を含まないものとする。

CAMは、内容による一致検索、及び一致した語へのデータの書込み等を行えるメモリである。本稿では、[4]のCAMの利用を考える。CAMのアドレスに要素番号を対応させ、アドレスiの内容として要素iの属する集合名を記憶する。また、初期状態として、各要素iは集合iに属するものと考え、CAMのすべての語の内容を0としておき、以後のUNION操作によって新しい集合名に書き換えてゆく。CAMを利用したFIND，UNIONは、次の操作となる（図1参照）。

FIND(i)　… アドレスiの語の読出し
　もし0ならばiとする

UNION(i,j) … step1　R1 ← アドレスiの語
　step2　R2 ← アドレスjの語
R1≠0,R2≠0のとき　step3　検索レジスタ ← R2
　step4　一致検索
　step5　一致した語 ← R1
R1≠0,R2=0のとき　step3　アドレスjの語 ← R1
R1=0,R2≠0のとき　step3　検索レジスタ ← R2
　step4　一致検索

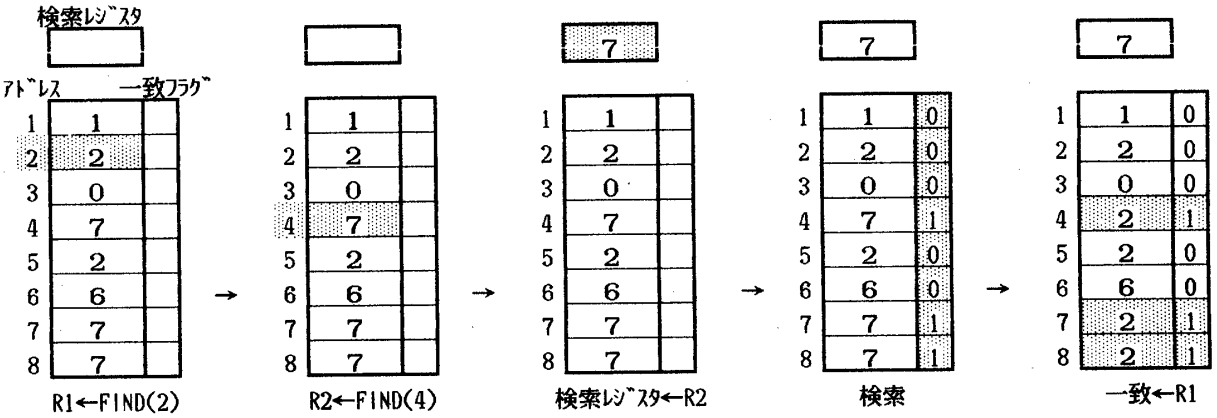

図1 UNION(2,4)のCAMの操作

A High Speed Unification Algorithm Using Content Addressable Memory
Masaaki OHKUBO, Naofumi TAKAGI, Hiroto YASUURA and Shuzo YAJIMA
KYOTO University

step5　一致した語 ← i
step6　アドレスiの語 ← i
R1=0,R2=0のとき　step3　アドレスiの語 ← i
step4　アドレスjの語 ← i

CAMでは、並列検索、並列書込みが可能なため、上記の各ステップは1クロックで行える。したがって、FINDは1～2クロックで、UNIONは4～7クロックで完了する。

3．単一化アルゴリズム[1]

項グラフの節点を前章の要素に対応させ、単一化をUNION-FIND問題に帰着する。アルゴリズムを図2に示す。

```
Algorithm UNIFY(t1,t2)
if BIND(t1)≠BIND(t2) then do
 if "BIND(t1) or BIND(t2)が変数節点" then MERGE(t1,t2)
 else do
   if BIND(t1).label≠BIND(t2).label then "fail";
   MERGE(t1,t2);
        /* BIND(t1)の出次数をnとする */
   for i=1 to n do
     if BIND(t1i)≠BIND(t2i) then UNIFY(t1i,t2i) od
 fi
fi
```

図2 単一化のアルゴリズム

ここでBIND,MERGEは、それぞれFIND,UNIONを利用した操作で、集合名として節点名を使う。また、MERGE後の集合名は、一方のBINDの結果が関数節点の場合にはその関数節点名に合わせる。図2のアルゴリズムを実現するハードウェアのブロック図を図3に示す。入力となる項グラフはRAMに格納し、項グラフの各節点を記憶している語のアドレスを節点番号とする。項グラフ処理部では関数節点対を入力し、その子節点対を出力する（アルゴリズムの再帰呼び出しに対応）。UNION-FIND処理部ではCAMを利用して単一化を行う。また、全体としてパイプライン処理を行い、処理時間の短縮化を図る。

4．評価

[1]では、UNION-FIND専用のハードウェアとして、UNION-FINDメモリ(UFM)を提案している。UFMは、CAMを2ポート読出し可能とし、また読出しと同時に検索を行えるもので、UNION-FIND問題を、CAMを利用する場合の約半分のクロック数で処理できる。しかし、CAM，UFMを利用した場合のそれぞれについて、シミュレーションにより単一化の処理時間を比較したところ、両者にはほとんど差が見られなかった。図3で、UNION-FIND処理部における処理時間よりも、項グラフ処理部におけるデータ（節点）の読出しに、より時間がかかる。このため、CAM(UFM)とRAMの並列アクセスを仮定しているこのハードウェアでは、CAMとUFMの処理速度の差は、項グラフ処理部の処理時間で吸収されたと考えられる。したがって、単一化に関しては、CAMでも十分な速度が得られ、高集積化が可能[5]なCAMを利用する本方法の有効性が確認できた。

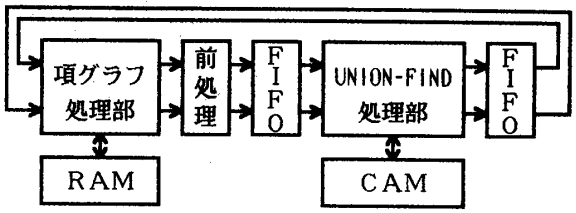

図3　単一化用ハードウェアの構成

5．おわりに

連想メモリを利用したUNION-FIND問題の処理法と、それを応用した単一化について述べた。本稿で提案した手法は、単純かつ高速であり、様々な問題に対して有効と考えられる。

[謝辞]
CAMに関する資料を頂いたNTT厚木電気通信研究所の小倉武氏に感謝致します。本研究は一部文部省科学研究費による。

[参考文献]

[1] 安浦,大久保,矢島：論理型言語における単一化操作のハードウェアアルゴリズム，信学技報，EC84-67(1985)
[2] J.E.ホップクロフト,J.D.ウルマン共著；野崎,野下共訳：アルゴリズムの設計と解析Ⅰ，ｻｲｴﾝｽ社,1977，第4章
[3] R.E.Bryant:A Switch-Level Model and Simulator for MOS Digital Systems, IEEE Trans.on Comp. vol.c-33, No.2, Feb., 1984
[4] 小倉,山田,丹野,石川：4Kb CMOS 連想メモリLSI，信学技報，SSD83-78(1985)
[5] 小倉,山田慎,山田順：20Kb CMOS 連想メモリLSI，昭61信学総全大,477