

7D-9

プロダクションシステムにおける
高速条件照合方法

小池 博 富田 浩史

吉村 精純

((株)日立製作所システム開発研究所) (日立西部ソフトウェア(株))

1. はじめに

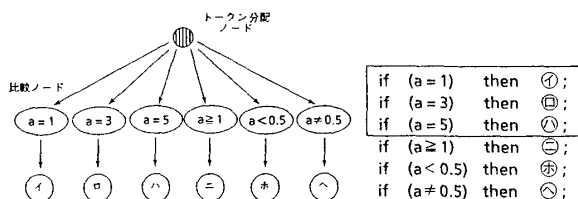
近年、プロダクションシステム(P S)の高速化を目的として、推論処理の大半を占める条件照合処理の効率化の研究が盛んに行われている。

本稿では、プログラミング言語のcase文の効率的実現手法を応用したP Sの高速条件照合方法と、この方法が適用可能である条件の範囲を拡大するための相互排他条件変換法を提案する。

2. 排他性に着目した高速条件照合方法

2.1 排他リンク方式

P Sの照合処理の効率的な実現手法として、Reteネットワーク(Reteネット)を用いたReteアルゴリズム[1]がある。Reteネットの最適化として、田野[4]は失敗確率の高い条件判定(定値との条件判定)ノードをネットワーク上部に配置する手法を提案した(図1参照)。この部分の処理は手続き型プログラムで表現すると図2となり、条件文を逐次実行する逐次照合方式と解釈できる。



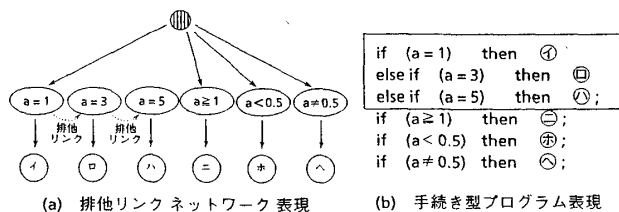
```

if (a=1) then ①;
if (a=3) then ②;
if (a=5) then ③;
if (a≥1) then ④;
if (a<0.5) then ⑤;
if (a≠0.5) then ⑥;

```

図1 Reteネット上部 図2 手続き型プログラム表現 (逐次照合方式)

荒屋は互いに排他的な等号比較条件のノード間を排他リンクで接続し、その間では条件が"真"となったときに比較を打ち切る高速化方式(排他リンク照合方式[3])を提案している。図1を排他リンク方式の下で置き換えたのが図3(a)で、処理としては図3(b)である。



```

if (a=1) then ①;
else if (a=3) then ②;
else if (a=5) then ③;
if (a≥1) then ④;
if (a<0.5) then ⑤;
if (a≠0.5) then ⑥;

```

(a) 排他リンク ネットワーク 表現 (b) 手続き型プログラム表現

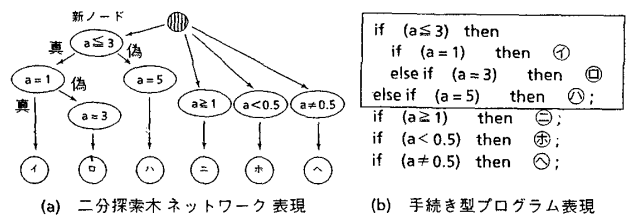
図3 排他リンク照合方式

2.2 case文実現手法を応用した高速条件照合方法

排他的な等号比較条件の効率化として、プログラミング言語のcase文の各種実現手法[2]が応用できる。本節では、case文実現手法のReteネット上への応用として、二分探索木照合方式、テーブル二分探索照合方式、ジャンプテーブル照合方式を提案する。

(1) 二分探索木照合方式

二分探索木を用いる方式は、条件内の比較値を分類し、これを基に二分探索木状に新たな条件文を配置して検索を行う。二分探索木を用いる方式のReteネット上への適用は、条件判定の"真", "偽"によって分岐先が異なる新ノードを導入し、このノードを用いて二分探索木を構成することで行える(図4(a)参照)。処理としては、図4(b)である。

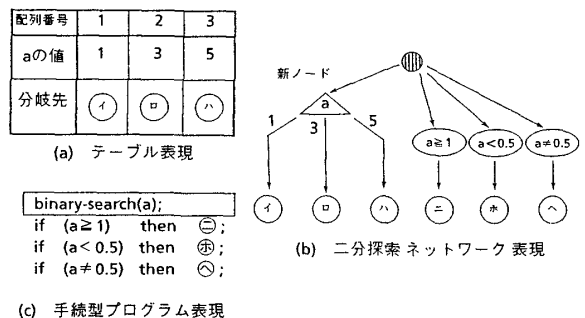


(a) 二分探索木 ネットワーク 表現 (b) 手続き型プログラム表現

図4 二分探索木照合方式

(2) テーブル二分探索照合方式

テーブルを用いた二分探索法を用いる方式は、分類した条件内の定値と分岐先をテーブルに格納し(図5(a)参照)、このテーブルを二分探索して分岐先を求めるもので、Reteネット上への適用は、内部に持たせたテーブルを二分探索し、値が存在すればその分岐先へ分岐し、無ければ次の条件判定へ分岐する新ノードを導入することで実現できる(図5(b)参照)。処理としては、図5(c)である。



(a) テーブル表現

```

binary-search(a);
if (a≥1) then ④;
if (a<0.5) then ⑤;
if (a≠0.5) then ⑥;

```

(c) 手続型プログラム表現

図5 テーブル二分探索照合方式

(3) ジャンプテーブル照合方式

条件内の変数が小さな自然数だけを取る場合、ジャンプテーブル方式の適用が可能である。ジャンプテーブル

とは、配列番号と検索する変数の値を同じとみなした表であり。この表を用いた場合、変数の値で直接配列要素をアクセスするだけで、分岐先を高速に得ることができる。表の引き方を除けば、テーブル二分探索照合方式と同じ手法でReteネットに適用できる。

2.3 方式比較

上記の各方式の平均比較回数を以下に示す(Nは等号比較の個数, Mは他の比較の個数)。

方式名	平均比較回数
逐次	$N + M$
排他リンク	$N / 2 + M$
テーブル二分探索	$\log_2 N + M$
二分探索木	$\log_2 N + M$
ジャンプテーブル	M

メモリ使用量は、逐次方式、排他リンク方式の方が、他の方式より少ない。

3. 相互排他条件変換法

2.2節の提案方式が有効に働くには、各条件が互いに排他的である性質が必要である。そこで、提案方式の適用範囲を拡大するために、照合対象である条件集合を互いに排他的な新条件集合へ変換する方法を提案する。この方法により、不等号、否定比較を含む条件照合においても、効率的な上記提案方式が適用可能となる。

3.1 相互排他条件への変換

ある変数aに対し、その変数aと定値との比較からなる条件文集合(R(a)とする)では、変数aの値に対して、R(a)内の各条件文の“真”“偽”は一意に決まる。また、R(a)内の各条件文の比較対象定値(その集合をC(a)とする)を基準に、変数aの値域を複数の領域に分割すると、1つの分割領域内では、変数aがどの値を取っても、R(a)内の各条件文の“真”“偽”は一定である。この性質を用いると、R(a)は互いに排他的な新条件文の集合に変換できる。

変換アルゴリズム(図6, 図7参照)

1. C(a)内の全定値を基準に、変数aの値域を重なりのない複数の領域(その集合をRE(a)とする)に分割する
2. RE(a)内の各領域reとR(a)内の各条件文rに対し、
 - 2.1 変数aが領域re内にある場合の条件文rの“真”“偽”を求める
 - 2.2 もし結果が“真”ならば、領域reに条件文rが“真”の場合の分岐先を対応付ける。
3. RE(a)内の各領域reに対し、対応する分岐先をTHEN部に持ち、領域reを条件とする新条件文を作成する

種類	条件	各条件を満たす変数aの値域
等号	① $a = 1$	0.5 --- 1
	② $a = 3$	1 --- 3
	③ $a = 5$	3 --- 5
不等号	④ $a \geq 1$	1 --- ∞
	⑤ $a < 0.5$	0.5 --- ∞
否定	⑥ $a \neq 0.5$	0.5 --- ∞

分割領域	0.5	1	3	5	...
分岐先	⑤ ⑥	① ④ ⑥	② ④ ⑥	③ ④ ⑥	④ ⑥

図6 相互排他な条件への変換

```

if (a < 0.5) then ⑤⑥;
if (a = 0.5) then ;
if (0.5 < a < 1) then ②;
if (a = 1) then ①③④;
if (1 < a < 3) then ③④;
if (a = 3) then ②③④;
if (3 < a < 5) then ③④;
if (a = 5) then ①③④;
if (a > 5) then ③④;
  
```

図7 変換した相互排他条件集合

3.2 排他変換した新条件への照合方式の適用

2.2節の照合方式を排他変換した新条件に適用する場合、ジャンプテーブル照合方式はそのまま適用できる。二分探索法を用いる各照合方式の適用では、条件比較の対象が定値でなく領域である点が問題となる。しかし、変数の値がどの領域に含まれるかの判定は、領域を区切る各定値を対象とした二分探索で行えるので、照合方式の適用に問題はない。

Reteネットへの適用の際、分岐先が新条件に対して複数存在するという問題があるが、トークン分配ノードを図8のように用いれば解決する。

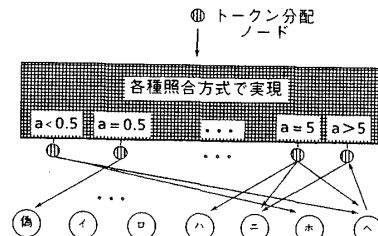


図8 相互排他変換した条件のネットワーク表現

3.3 変換を行った場合の比較回数

排他条件変換法を施した後の平均比較回数を以下に示す(Nは等号比較の個数, Mは他の比較の個数)。

方式名	平均比較回数
テーブル二分探索	$\log_2 (N + M) + 1$
二分探索木	$\log_2 (N + M) + 1$
ジャンプテーブル	0

排他条件変換法を適用することで、2.2節提案方式の平均比較回数は減少する。また、変換前の各条件間と同じ定値が存在する場合、変換後の新条件の数が減るため、上記の平均比較回数は更に減少する。

4. おわりに

プロダクションシステムの照合処理の効率化として、二分探索法、ジャンプテーブル法をReteネットワークの処理に応用した。更に、相互排他条件変換法の提案により、適用可能な条件の範囲を拡大した。

参考文献

- [1] Forgy, C.L., "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", Artificial Intelligence, 1982-19
- [2] S.Arthur, "The Implementation of Case Statements in Pascal", SOFTWARE-PRACTICE AND EXPERIENCE, 1981-11
- [3] 荒屋, 他 "プロダクションシステムのための高速パターン照合アルゴリズム", 情報処理学会論文誌, 1987-28
- [4] 田野, 他 "知識ベースシステム構築用ツールEUREKAにおける高速処理方式", 情報処理学会論文誌, 1987-28