人　工　知　能　104－5
（1996．5．14）

プロダクションシステムのためのジョイン演算の適切な順序に関する一考察

南保英孝　木村春彦　広林茂樹
金沢大学 工学部 電気・情報工学科 人工知能研究室
〒920 石川県金沢市小立野 2-40-20

エキスパートシステムは専門家の代行を務めるシステムであり，人間の知識をルール化したものとデータとを照合させることにより，推論を行うシステムである．このエキスパートシステム構築用のツールとしてよく用いられるものにプロダクションシステムがある．しかし，プロダクションシステムはルールとデータの照合に時間がかかるため，条件照合の高速化が課題となっている．

本稿では，ルールの条件部の条件要素の照合の順序を変えると，条件照合にかかる時間が変化することに着目し，照合回数を確率的に予測し最適化する手法を提案する．また，その手法が実際にどの程度有効かを実験によって明らかにする．

A Consideration on Ordering of Join Operations for Production Systems.

Hidetaka Nanbo, Haruhiko Kimura, and Shigeki Hirobayashi
Faculity of Technology, Kanazawa University.
2-40-20, Kodatsuno, Kanazawa, Ishikawa, 920, Japan

Production systems are an established method for encoding knowledge in an expert systems. But the matching speed of the production systems is a problem.

In this report, we propose an optimization method for production systems. The method predicts matching times by using static informations that acquired by analyzing productioin and detects the best match order which matching times is the smallest. It will be able to make production systems more faster, because CPU time are proportional to matching times.

1 はじめに

専門家の代行を勤めるシステムにエキスパートシステムがある．このエキスパートシステム構築用のツールとして，プロダクションシステムが広く用いられている．プロダクションシステムは，知識を “IF~THEN~” で表されるルールで表現し，データとルールの照合によって推論を行なっていくルールベースシステムとなっている．このように，プロダクションシステムは人間の知識を表現するのに馴染みが良いのだが，条件照合速度が遅いという問題がある．

これまでも，条件照合の高速化に関するさまざま研究がなされており，多くの高速化の手法が提案されている．

本稿で提案する高速化手法は，ルールの条件部の条件要素の順序を替えることで，条件照合にかかる時間が変化する点に着目し，条件照合の高速化を図るものである．具体的には，照合時に用いられるネットワークを，照合のコストがもっとも小さくなるようなネットワークに最適化する．また，提案手法ではルールを実行せずとも得ることの可能な静的情報のみからネットワークの最適化を行なう．実行時に得られる動的情報を用いなくても良いため，より速やかに最適化を行なうことが可能である．また，最適化のプロセスはユーザーが意識せずとも自動的に行なうことが可能であり，一種のフィルタとして用いることができる．よって，知識ベース記述に関して経験のないユーザーが効率の悪いルールを記述した場合であっても，実行時間の増加を未然に防ぐことができるという利点も考えられる．

また，実際に知識処理を行なう知識ベースに対して提案手法を適用し，有効性を明らかにする．

2 Rete ネットワーク

2.1 Rete アルゴリズム

本稿で対象とするプロダクションシステムは OPS5 である．OPS5 は有名な前向き推論方式のプロダクションシステムである．OPS5 では，条件照合のアルゴリズムに Rete アルゴリズムが用いられている．Rete アルゴリズムでは，ルールの条件部を Rete ネットワークと呼ばれるデータフローグラフに変換し，データをトークンとしてネットワークに入力することで条件照合を実現している．

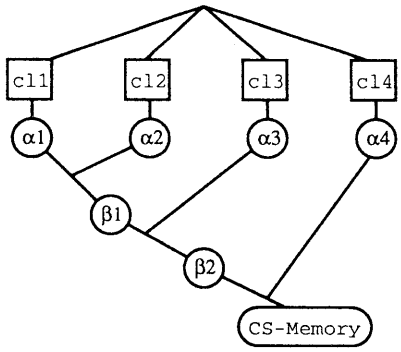

図 1: ルール rule1 の Rete ネットワーク

```
(p rule1
  (c1 ^a11 <A> ^a12 <C> ^a13 <C>)
  (c2 ^a21 <B> ^a22 <D>)
  (c3 ^a31 V1  ^a32 <B> ^a33 V2 ^a34 <A>)
  (c4 ^a41 V3  ^a42 <A>)
-->
   (actions))
```

これは，OPS5 のルールの例である．このルール rule1 は，Rete ネットワークに変換されると図 1 の様になる．図中の α_1 から α_4 までは α メモリと呼ばれるノードである．これらは，それぞれが条件要素に対応しており，条件要素を満足したトークンが記録される．また，β_1,β_2 は β メモリと呼ばれるもので，ジョイン演算を満足したトークンが記録される．なお，ジョイン演算とは，複数の条件要素に現れる同じ変数の値が一致するかについてのチェックである．最終的には CS メモリにルールの条件を全て同時に満足するトークンが記録されることになる．

Rete ネットワークには，照合の途中結果が記録される．そのため，同じルールに関する照合を再度行う時には，前回の照合から比べてデータが変化した部分についての照合だけですむ．つまり，無駄な照合を行う必要がなく，効率が良くなる．しかし，データの変化が大きかったり，ネットワークに入力されるトークンの量が多い場合には，依然として照合に時間がかかる．

2.2 条件照合の順序

以下のようなルール rule2 を考える．

```
(p rule2
  (c1 ^a11 <A> ^a12 <C> ^a13 <B>)
  (c3 ^a31 V1  ^a32 <B> ^a33 V2 ^a34 <A>)
```

```
  (c4 ^a41 V3  ^a42 <A>)
  (c2 ^a21 <B> ^a22 <D>)
-->
  (actions))
```

rule1 と rule2 は，条件部に記述された条件要素は同じであるが，その順序が異なっている．しかし，rule1 と rule2 の実行時間を調べたところ，ある条件下では約 3 倍の差がつくことが分かった．

本稿で提案する高速化手法は，この点に着目し，ルールの条件部の条件要素の順序を最も照合時間が短くなるような順序に並び換えることで，条件要素の最適化を行ない高速化を図るものである．また，照合時間を予測するための照合回数の期待値の計算方法を提案する．この計算は，知識ベースの静的情報のみを用いて行われ，知識ベース実行時に得られるような動的な情報は用いない．つまり，静的情報のみを用いることで，速やかな最適化が可能となる．

3 条件照合回数の予測

以下では，Rete ネットワークにおける条件照合回数の予測方法について述べる．

3.1 静的情報の取得

本研究の特徴の一つとして，動的情報を用いずに静的情報のみから条件照合の最適化を行なうという点が挙げられる．ここで，静的情報とは，知識ベースの記述のみから得られる情報のことを指す．

以下では，Rete ネットワークにおける条件照合回数の予測に必要な静的情報の取得方法について述べる．予測の際に必要となる静的情報には，データである WME の分布，属性値の種類の 2 つがある．

3.1.1 WME のクラスの分布

知識ベースを実行した場合に生成，削除される WME のクラスがどのような分布となるかについての情報を WME のクラスの分布と呼ぶことにする．

本研究では，WME のクラスの分布を以下の 3 種類として仮定した．

- 一様分布

  全てのクラスの WME が等確率で生成，削除されるとしたもの

- 頻度

  知識ベース上で，あるクラスの WME が生成，削除される回数を WME のクラスの分布としたもの

- zipf 分布

  頻度から zipf 関数を用いて zipf 分布を求め，それを WME のクラスの分布としたもの

OPS5 では，WME の生成は make 命令，削除は remove 命令，更新は modify 命令で行なわれる．頻度分布や zipf 分布を利用する場合には，以上の 3 命令を知識ベースの中から検索し，各命令の出現頻度から WME の生成，削除の頻度を決定する．さらに，zipf 分布を用いる場合には，以下の計算が必要となる．

zipf 分布は zipf の法則から導き出される．zipf の法則とは，計量言語学の分野の語彙分布の法則の 1 つであり，語の頻度 f と順位 r の積が

$$f \cdot r = C \text{（一定）} \qquad (1)$$

の関係を満たすいう法則である．これを利用する．

3.1.2 属性値の種類

属性値の種類とは，あるクラスのある属性の値がどのような値を取り，その値は全部で何種類になるか，ということを意味している．この属性値の種類は，イントラコンディションテストの成功率や，ジョイン演算の成功率の計算に必要となる．

属性値の種類の取得には，WME の分布を得る時と同じように，知識ベースの各ルールの行動部の命令と条件部の条件要素の属性値を参照する．具体的には，条件要素で現れる属性値と行動部の make 命令，modify 命令を調べることになる．これらの箇所を調べる事で，属性の取り得る属性値とそれらの種類が分かる．

なお，本研究では，属性の取り得る属性の種類は，その取り得る値を一様に取るものと仮定している．

3.2 Rete ネットワークにおける照合回数の予測

WME の分布，属性値の種類の 2 つの静的情報が分かれば，Rete ネットワークにおける条件照合

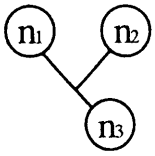

図 2: 基本的なネットワーク

の回数を確率的に求めることができる．計算の手順は以下のようになっている．

(1) イントラコンディションテストが成功する確率を求める．

(2) ジョイン演算が成功する確率を求める．

(3) 各ノードに記録されているトークンの量を計算する．

(4) ジョイン演算の回数とジョイン演算を通過するトークン数を求める．

図2の基本的なネットワークを例として計算方法を示す．ここで，図中のノードn_1はReteネットワークでのαまたはβメモリに対応する．また，ノードn_2はαメモリに対応する．ノードn_3はβメモリに対応する．一般のReteネットワークは，図2のネットワークの組合せからなっている．

以下では，ノードn_1, n_2はどちらもαメモリに対応しているものとする．また，ノードn_1, n_2に入力されるトークン数は，それぞれデータの追加のトークンの場合はm_1, m_2，削除のトークンの場合はd_1, d_2であるとする．これらの値は，知識ベースを調べて得られたWMEのクラスの分布より決まる値である．

3.2.1 イントラコンディションテストの成功率

イントラコンディションテストとは，Reteネットワークに入力されたトークンが，対応する条件要素を満足するかどうかのチェックのことである．

いま，以下のような条件要素を考える．

```
(class ^attribute '属性値の条件')
```

“属性値の条件”にはWMEの属性値が満たすべき条件が記述される．条件は，属性値(定数)，変数，述語を伴う表現，選言，連言等の場合がある．それぞれの場合について，入力されたトークンがイントラコンディションテストを満足する確率の計算方法が異なる．ここでは，属性値の条件が定数，変数の場合について説明する．なお，知識ベースを調べた結果，クラス“class”の属性“attribute”の取り得る属性値の種類はn種類となったものとする．

(i) 属性値の場合

属性値の条件として属性値が定義されている場合には，その属性値と，入力されたトークンの同じ属性の属性値が一致しなくてはならない．よって，イントラコンディションテストが成功のはn種類中1種類なので，確率は$1/n$となる．

(ii) 変数の場合

属性値の条件として変数が定義されている場合には，任意の値と一致するので，イントラコンディションテストが成功する確率は1となる．

3.2.2 変数の取り得る値の種類

ジョイン演算の成功率を計算する前に，ジョイン演算で無矛盾性チェックの対象となる変数が，どのような値を取るかを知る必要が有る．これを変数の取り得る値の種類と呼ぶ．一般に，変数の取り得る値の種類は，その変数が現れる条件要素の属性の属性値の種類と種類と等しくなる．これは，変数に代入される属性値は，イントラコンディションテストを通過したトークンの属性値だけであるためである．

また，照合が進むにつれて，未確定変数の値が確定していくため，その点も考慮して変数の取り得る値の種類を決定する必要が有る．

また，本研究では，変数がどの値を取るかについては，取り得る値を等確率で取ると仮定している．

3.2.3 ジョイン演算が成功する確率

ジョイン演算で無矛盾チェックの対象となる変数が同じ値を取って一致し，ジョイン演算をパスする確率のことである．図2を用いて説明する．今，n_1, n_2間のジョイン演算で無矛盾チェックの対象となる変数を

$$V_1, V_2, \cdots, V_n$$

とする．また，それぞれの変数V_nがノードn_iで取り得る値の種類をそれぞれ

$$W_{n_i1}, W_{n_i2}, \cdots, W_{n_in} \quad (2)$$

とする．ここで，ノードn_iにおける変数V_1の値の種類の内容は，

$$T[1], T[2], \cdots, T[W_{n_i 1}] \tag{3}$$

となっているものとする．また，属性の属性値が取り得る値の分布は一様分布である仮定しているため，変数V_1が$T[1]$となる確率は$1/W_{n_i 1}$となる．よって，n_1, n_2に記録されているトークンの変数V_1が属性値$T[1]$を取って一致する確率は，

$$\frac{1}{W_{n_1 1}} \times \frac{1}{W_{n_2 1}} = \frac{1}{W_{n_1 1} \cdot W_{n_2 1}} \tag{4}$$

となる．また，変数V_1はn_1, n_2でそれぞれ$W_{n_1 1}$, $W_{n_2 1}$種類の値を取り得る．変数が一致するのは，n_1とn_2でV_1が同じ値を取る場合である．そこで，n_1, n_2での変数V_1の値のうち，n_1, n_2で共通に現れる属性値を調べ，変数が一致する組合せ数を得る．共通の属性値がC_{V1}種類となったとき，変数V_1がジョイン演算において一致する確率は，

$$\frac{1}{W_{n_1 1} \cdot W_{n_2 1}} \times C_{V1} = \frac{C_{V1}}{W_{n_1 1} \cdot W_{n_2 1}} \tag{5}$$

となる．他の変数$V_2, \cdots, V_n$についても同様に考えると，変数V_m $(2 \leq m \leq n)$の一致する確率は，

$$\frac{1}{W_{n_1 m} \cdot W_{n_2 m}} \times C_{Vm} = \frac{C_{Vm}}{W_{n_1 m} \cdot W_{n_2 m}} \tag{6}$$

となる．よって，変数$V_1, V_2, \cdots, V_n$が全て同時に一致する確率，つまり，ジョイン演算が成功する確率は，

$$\frac{C_{V1}}{W_{n_1 1} W_{n_2 1}} \times \frac{C_{V2}}{W_{n_1 2} W_{n_2 2}} \times \cdots \times \frac{C_{Vn}}{W_{n_1 n} W_{n_2 n}} = \prod_{k=1}^{n} \frac{C_{Vk}}{W_{n_1 k} W_{n_2 k}} \tag{7}$$

となる．

3.2.4 各ノードに記録されているトークンの量

イントラコンディションテストが成功する確率と，変数が一致してジョイン演算が成功する確率が求められたならば，次に各ノードに記録されるトークンの量が計算できる．

再度，図2を用いて説明する．今，ノードn_iにおけるイントラコンディションテストの成功率がそれぞれI_i $(i = 1, 2)$，ジョイン演算の成功率がrとなったとする．

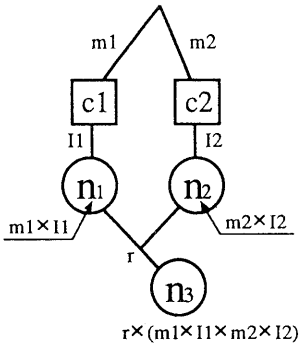

図 3: 各ノードに記録されるトークン数

ノードn_iに入力されるトークン数をm_iとする．ノードn_iはαメモリであるとしているので，入力されたトークンに対してイントラコンディションテストが行なわれる．イントラコンディションテストを満足し，ノードn_iに記録されるトークンの数M_iは，

$$M_i = m_i I_i \tag{8}$$

となる．

このとき，ノードn_3に記録されるトークンの数は，n_1, n_2に記録されているトークンの組み合わせの中で，ジョイン演算を通過する組の数となる．つまり，ノードn_1, n_2のトークンの組み合わせ数$M_1 \times M_2$のうちで，変数が一致する組となる．よって，

$$M_1 \times M_2 \times r = r M_1 M_2 \tag{9}$$

がノードn_3に記録されるトークン数となる (図3)．

3.2.5 ジョイン演算で必要となる照合回数

次に，Reteネットワークを用いて照合作業を行うときに，ジョイン演算時に必要となる照合回数を計算する．この計算は，WMEが追加された時と削除された時に分けて考える必要がある．

(1) WMEが追加される場合

WMEが追加されたときには，そのWMEを表すトークンがReteネットワークに追加される．追加されたトークンは，対応する条件要素との間でイントラコンディションテストが行われ，それぞれのαメモリに記録される．そして，そのトークンが次のノード（βメモリ）に記録されるかどうかを調べるため，ジョイン演算が行なわれる．

今，ノードn_1に対応するトークンが1つ入力された場合を考える．このトークンがイントラコン

ディションテストを満足し，n_1 に記録される確率は I_1 である．そして，ノード n_2 との間のジョイン演算が行なわれる．その時の照合回数は，n_1 に記録されているトークンの変化分 $(1 \times I_1)$ と n_2 に記録されている全てのトークンとの間の照合回数となる．よって，

$$1 \times I_1 \times M_2 = I_1 M_2 \qquad (10)$$

となる．同様に，n_2 に対応するトークンが 1 つ追加された時の照合回数は，

$$1 \times I_2 \times M_1 = I_2 M_1 \qquad (11)$$

となる．

ノード n_1, n_2 に入力されるトークンの割合は，それぞれ M_1, M_2 であるため，追加時に図 2のネットワークで必要となる照合回数 MT_a は，

$$MT_a = M_1 \times I_1 M_2 + M_2 \times I_2 M_1 \qquad (12)$$

となる．

また，ジョイン演算をパスして次のノードに流れるトークンの組の数は，ジョイン演算で行なった照合のうち，変数が一致したトークンの組の数となる．よって，

$$r \times MT_a \qquad (13)$$

となる．

(2) WME が削除される場合

WME が削除されるときには，マイナストークンが Rete ネットワークに入力される．マイナストークンが入力された場合には，それぞれのノードで削除の対象となる WME を探すための照合が行なわれる．その照合は，ノード内の全てのトークンを調べる作業であり，照合回数はノード内のトークン数となる．

n_i にマイナストークンが入力される割合を d_i とする $(i = 1, 2)$．n_1 に入力された場合には，n_1 と n_3 で削除の照合が行なわれ，n_2 に入力された場合には n_2 と n_3 で照合が行なわれる．よって，WME の削除時に必要となる照合回数 MT_r は，式 8，式 9より，

$$MT_r = d_1(M_1 + rM_1M_2) + d_2(M_2 + rM_1M_2) \qquad (14)$$

となる．

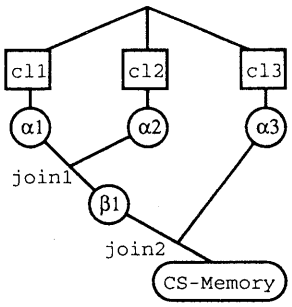

図 4: ルール rule3 の Rete ネットワーク

よって，最終的に図 2で必要となる照合回数 MT は，

$$MT = MT_a + MT_r \qquad (15)$$

で表される．

3.3 具体例

具体的に以下のルール rule3 を用いて，上記の計算方法を説明する．なお，このルールを Rete ネットワークに変換したものを図 3 に示す．簡単のため，全ての属性の属性値の取り得る値の種類は 1 ～ 10 までの整数 10 種類とする．また，WME の追加に関するクラスの分布は cl1 : cl2 : cl3 = 10 : 20 : 30 とし，削除はないものとする．

```
(p rule3
  (cl1 ^at11 <A> ^at12 { > 5})
  (cl2 ^at21  8  ^at22 { <A> < 7})
  (cl3 ^at31 <B> ^at32 { <A> << 1 2 >>})
-->
  (actions))
```

(1) イントラコンディションテストの成功率

まず，イントラコンディションテストの成功率を計算する．クラス cl1 の条件要素では，属性 at11 の属性値の条件は変数であり，必ず条件が満たされる．また，属性 at12 では述語 “>” が使用されており，この条件を満たす属性値は 6,7,8,9,10 の 5 種類となる．よって，クラス cl1 の条件要素についてのイントラコンディションテストが成功する確率は 5/10 となる．クラス cl2 の条件要素では，属性 at21 の条件を満たす確率が 1/10，属性 at22 の条件を満たす確率は 6/10 であり，イントラコンディションテストの成功率は 6/100 となる．クラ

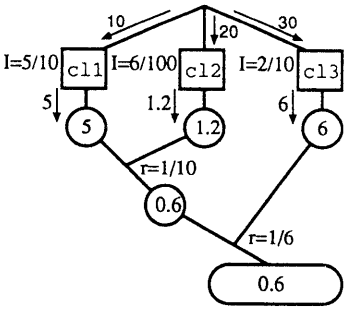

図 5: トークン数とジョイン演算の成功率

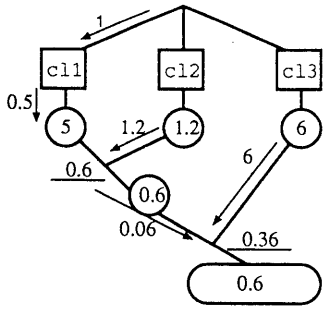

図 6: 照合回数の予測 (cl1 に追加の場合)

スcl3では，属性at31は任意，属性at32は1,2のいずれかの時に条件を満足する．よって，2/10となる．

(2) ジョイン演算の成功率

次にジョイン演算の成功率を計算する．join1では，変数<A>についての無矛盾チェックが行われる．cl1の条件要素では，変数<A>は10種類の値を取る可能性があり，cl2では1 ～ 6の6種類となっている．また，共通の属性値は1 ～ 6の6種類であるため，成功率は式7より1/10となる．そして，変数<A>の値の種類は1~6の6種類に制限される．同様にして，join2でも変数<A>のチェックが行われ，属性値の種類はcl2までででは1 ～ 6の6種類，cl3では1,2の2種類である．共通の属性値は1,2であるので，ジョイン演算の成功率は1/6となる．

(3) 各メモリ内のトークン数の予測

次に各メモリ内のトークン数を予測する．イントラコンディションテストの成功率と式8より，$\alpha_1 : \alpha_2 : \alpha_3 = 5 : 1.2 : 6$となる．また，$\beta_1$のトークン数はジョイン演算の成功率と式9から，0.6となり，CSメモリのトークン数は0.6となる(図5)．

(4) 照合回数の予測

最後にReteネットワークにおける照合回数を計算する．式10より，クラスcl1のWMEが追加されたときは，join1で0.6回の照合が必要となる．また，join1をパスしjoin2へ至るトークン数は，式13より0.06となる．よって，join2では0.36回の照合が必要となる(図6)．同様にして，cl2に追加されたときはjoin1で0.3回，join2で0.18回の照合が，cl3に追加されたときはjoin2で0.12回の照合が必要となる．

各αメモリに入力されるトークン数の分布を考慮して，最終的な照合回数は，

$$10 \times 0.96 + 20 \times 0.48 + 30 \times 0.12 \quad = \quad 22.8$$

となる．

3.4 最適照合順序の決定

予測照合回数が計算できれば，もっとも照合回数の少なくなるような，最適な照合順序が決定できるようになる．

条件部の条件要素の順序を変えながら予測照合回数を求め，その中でもっとも照合回数が少なくなった条件要素の順序を最適照合順序とする．その際に，条件要素のすべての組み合わせを考えていたのでは，計算に時間がかかる．また，OPS5の競合解決戦略による制約もある．そこで，条件要素の一部だけを並べ換える．具体的には，2番目からN番目にあてはまる条件要素を決定することにする．但し，負の条件要素は考慮に入れていないので，並べ換える対象となる条件要素は2番目から最初に現れる負の条件要素の1つ前の条件要素，となる．よって，並べ換える組合せの数は，条件要素数がn，並べ換える数がNのとき，

$$\frac{P_n}{P_{n-1-N}} \tag{16}$$

となる．これより，Nにより組合せ数が変化することが分る．ここで，Nを最適化レベルと呼ぶことにする．

4 実験

4.1 実験方法

実際の知識ベースに提案手法を適用し，実行時間がどの程度改善されるかを測定した．OPS5の知識ベースに対して最適化処理を行った後にコンパイルしたものと，最適化処理を行わずにコンパイルしたものの実行時間を測定した．なお，最適化レベルは4とした．

表 1: 実験結果

	ライフゲーム	オイラー路	乱数
最適化なし	263.5	14.9	4.1
最適化あり（一様分布）	251.4	53.71	0.14
最適化あり（頻度分布）	196.1	5.14	0.14
最適化あり（zipf 分布）	189.4	5.3	0.24
最適化にかかる時間	1.59	1.96	1.27

(単位 秒)

用いた知識ベースは以下の 3 種類である.

・ライフゲーム

16×16 のセル上でライフゲームを行う知識ベースである.

・オイラー路検出

オイラー路, いわゆる, 一筆書きの答えを見つける知識ベースである.

・乱数で作成されたルール

5 種類の WME のクラスを定義しておき, 条件部のクラス, 属性値の条件を乱数を用いて決定し, 4 つの条件要素からなるルールを 1 つ作成する. また, 追加する WME も乱数を用いて作成する.

4.2 実験結果

測定は SUN Sparc Classic 上で行い, 最適化にかかった時間と照合時間を測定し, その user 時間を調べた. 最適化にかかった時間は time 命令, 照合時間は OPS5 のコンパイラの出力を用いて測定した. それぞれ 20 回測定し平均を求めた. 結果を表 1に示す. 一様分布を用いるのは, 実際の知識ベースには適していないことが分かる. つまり, 実際の知識ベースでは, 生成される WME のクラスは一様ではないためであると思われる. また, 頻度分布や zipf 分布を用いた場合には, 実行時間はかなり改善される. ライフゲームでは約 30%, オイラー路では約 80% の高速化がなされた. また, 実際の知識ベースの場合, zipf 分布を用いた方が少しではあるが効率が良くなった. これは, zipf 分布を用いると頻度の差が縮まり, 頻度が少ないクラスの影響が現れるためであると考えられる.

5 まとめ

確率に基づいて照合回数を予測する手法を用い, 最適な条件要素の並び順を求める方法を提案した. また, 知識ベースに提案方法を適用し, 実際に高速化が可能であることを実験により実証した.

参考文献

[1] Wong,E. et al."Decomposition – a strategy for query processing", ACM Trans. D atabase Systems, Vol.1(No.3), pp.223–241(1976).

[2] Forgy, C.L.:"RETE: A fast algorithm for the many pattern / many object pattern match problem", Artif.Intell., Vol.19, No.1, pp.17–37(1982).

[3] Brownston,L., Farrelll,R., Kant,E. and Martin,N.:"Programming Expert System in OPS5: An Introduction to Rule–Based Programming", Addison–Wesley(1985).

[4] 石田亨, 桑原和宏：“プロダクションシステムの高速化技術”, 情報処理, Vol.29, No.5 , pp.467–477(1988).

[5] 石田亨, 横尾真：“プロダクションシステムのトポロジカル・オプティマイザ”, 情報処理学会, 知識工学と人工知能研究会, 56–7(1988).

[6] Toru Ishida:"An Optimization Algotirhm for Production Systems",IEEE Transactio ns on Knowledge and Data Engineering, Vol.6, No.4, pp.549–558 August1994