知識工学と人口知能　68－5
（１９９０．１．１８）

データ変更の履歴にもとづく照合ネットワークの動的変形

水上孝一　　植田栄治

広島大学大学院工学研究科

プロダクションシステムの高速化手法として、ルールのコンパイルによる照合効率化が提案されている。しかし、推論以前にルールをコンパイルするために推論中ルールの評価順序は推論対象とするデータによらず一定である。そのため、無駄なマッチングを繰り返している。

本論文では推論中に生じる「データ相互の構造的類似性に関する知識」を抽出し、従来のネットワーク構造を動的に再構成することによってパターンマッチングを行なうネットワークを強調化、限定するアルゴリズム RINGSを提案し、ネットワークの再構成による効率化をシミュレーション実験により確認する。

DYNAMICAL REORGANIZATION OF DISCRIMINATION NETWORKS WITH HISTORIC INFORMATION OF MODIFIED DATA PATTERN.

Koichi MIZUKAMI　Eiji UEDA

Graduate School of Engineering Hiroshima university
1-1-89, Higashisenda-machi, Naka-ku, Hiroshima 730, Japan

As a method of accelarating Production system, it has been suggested that the number of matching process is reduced by compilation with rule pattern. Under the compilation, matching of rule pattern elements in the fixed order causes to repeat the unnecessary pattern matching.

In this paper , we propose RINGS algorithm which reorganizes both the compiled network structure and matching order by using data relationship knowlege. RINGS is enabled to emphasis and restrict the searching network space. Then we make an experiment on RINGS by simulation and obtain a stisfactory result.

1.はじめに

プロダクションシステムはエキスパートシステムの構築手法として最もよく利用される基本推論アーキテクチャの一つである。プロダクションシステムでは知識を

IF ～ THEN ～

形のルールにより断片的に取り扱うことができるためルールベースの変更に対して柔軟に対応できる長所を持つ。しかし、ルール数の増大にともない知識の構造的冗長性に起因するパターンマッチング量の爆発的な増加により推論速度が低下する欠点がある。ルールを追加することにより成長する性格の強いプロダクションシステムでは、推論の高速化はプロダクションシステムの広範囲の分野への適用の大きな課題の一つとなっている[4]。

プロダクションシステムの高速化手法としてこれまで多くのアプローチがなされてきたが、その中の一つとしてルールのコンパイルによる高速化手法が提案されている。これはルールを構造化することによってルールの条件部の構造的な冗長性を削除し、全推論コストの約 90～98%を占めるパターンマッチングを効率化することで推論速度を向上させる手法である[1][2]。

ところがルールをコンパイルすることにより無駄な照合がおこっている。ルールをコンパイルすることによりルールパターン要素とデータパターン要素とのマッチング順序は線形に固定するため、必要のないマッチングを行なう必要がある。ルール数の増大に従ってこのような無駄なマッチングは増加する。

しかし、このような無駄なマッチングはマッチングを行なうルールパターンを限定することによって削除できる。推論実行以前にルールの排他性を抽出しコンパイル時に利用することに加えて[3]、推論実行時に生じるデータ間の類似性を抽出し前回までのマッチングの結果を利用することで後のマッチングにおいて必要なルールパターンをあらかじめ指定できる。

本研究では、データパターン間の類似性を推論実行中に効率よく抽出し、マッチングに利用するマッチングアルゴリズムRINGS(Relative Inserted Nodes Gate system)を提案する。

RINGS ではマッチングの経験をもとに推論実行中に判別ネットワークを動的に変形することによってデータパターンとルールパターンさらに他のデータパターンとの関係をネットワークの構造として明示し、判別に必要なルールパターンのマッチングを優先して行なう。 RINGSを適用することにより、判別ネットワークは現在の推論対象となっているデータにふさわしい構成に変形される。また RINGSは照合過程をのぞいてReteアルゴリズムに忠実に従っているので、Reteアルゴリズムを基本とする他の効率化法を組み合わせることでいっそうの効果が期待できる。

2.対象とするプロダクションシステム

本研究において対象とするプロダクションシステムは知識表現、パターンデータのコンパイルについてはReteアルゴリズムに従いマッチングインタープリタにおいて提案するアイデアが付加的にインプリメントされている。

推論は次の「認知－実行サイクル」によって行なわれる。

1.照　合　：パターンマッチングを行い現在のワーキングメモリの内容により、満足されるルールを競合集合に入れる。
2.競合解決：ある戦略に従い競合集合の中から実行するルールを決定する。
3.実　行　：選ばれたルールの実行部をワーキングメモリに作用させる。

Reteアルゴリズムにもとづく照合過程において、パターンマッチングはネットワークにコンパイルされたパターンデータの探索である。さらにネットワークは推論１サイクル単位の状態を保持するので、パターンマッチングはワーキングメモリの変化分のみであるが、記憶された状態を変更するために、変更以前のデータパターンと変更後のデータパターンのマッチングを行なう必要がある。本研究で対象とするプロダクションシステムは各

サイクルでのワーキングメモリの変化量の少なく、ワーキングメモリの状態を変化させるデータパターンの差分が少ないことを仮定している。

3. 照合効率化の考え方

本研究の照合効率化の考え方は

(A)条件判別ネットワークフィルタ

(B)条件部の排他的関係

を利用した従来の照合効率化に

(C)マッチングの成否の結果

(D)データ相互の類似性に関する知識

を用いたヒューリスティック則を付加したものである。

(A)に対応する従来の効率化は推論以前にルールを判別ネットワークにコンパイルすることによって行なわれる。マッチングはネットワークの線形縦型探索順に行なわれるため、推論の対象となるデータと独立であり、原理的に全探索を行なうので無駄なマッチングが多く存在する。(B)を利用することによってある程度無駄なマッチングを回避できるが線形探索に起因する無駄な照合が依然存在する。

提案マッチングアルゴリズムでは(A),(B)で得られたネットワークに加え(C)にもとづきネットワークを変形し、残された無駄な照合を極力回避する。そしてさらに(D)を利用することによって「データ間の同一要素」(以下簡単に「同分」とする)の重複したマッチングの削除を行いさらに効率化を図る。

4. 従来方法

4-1. 条件判別ネットワークフィルタ

Reteアルゴリズムにもとづく照合過程はルール間の構造的類似性により、コンパイルされた推論1サイクル単位の記憶をもつネットワークを条件判別フィルタとして利用している[2]。ネットワークを条件判別フィルタとして用いることによって照合過程はデータパターンによるネットワーク探索により行なわれることになる。

たとえば、次のようなルールが存在したとする、それぞれ条件が満たされれば &で示される条件パターンを 結論部のパターンへ変更することを表わしている。R1の場合このルールの発火よりパターン (C2 ^A21 ^A22 V22)は(C2 ^A21V21 W22)に変更される。

```
(R1 &1(C1 ^A11 V11 ^A12 V12)
    &2(C2 ^A21 V21 ^A22 V22)
      ->
    MOD(&2 C2 ^A21 V21 ^A22 V22))
(R2 &3(C3 ^A31 V31 ^A32 V32)
    &4(C2 ^A21 V21 ^A22 W22)
      ->
    MOD(&3 C3 ^A31 V31 ^A32 V32))
```

これらのルールが図1のような判別ネットワークにコンパイルされるとする。

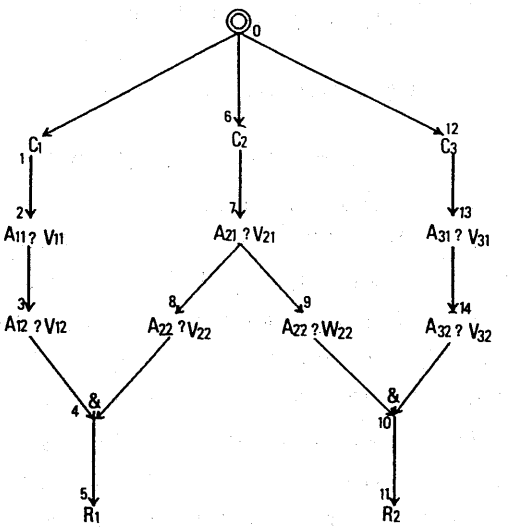

図1. 条件判別ネットワーク

全ての条件に対してデータの照合を行なう基本的な方法は、固定されたネットワークをパターンエレメントのマッチングが全て失敗するまで線形縦型探索を行なうことであるが、このことはデータパターンは探索されたノードシーケンスの一部であることを示している。

いまR1が実行されたとする。R1が実行可能となるためにはデータパターン

<C1 ^A11 V11 ^A12 V12>

<C2 ^A21 V21 ^A22 V22>

がすでにネットワーク内に保持されていることである。よってこのデータパターンをネットワーク内のメモリから削除するために、同一のパターン

<C2 ^A21 V21 ^A22 V22>

によりネットワークを再探索する必要がある。そして新たなデータパターン

<C2 ^A21 V21 ^A22 W22>

によるネットワーク探索によって、パターンマッチングが行なわれる。この変更に要するルールパターン要素シーケンスは図２のように表わされる。

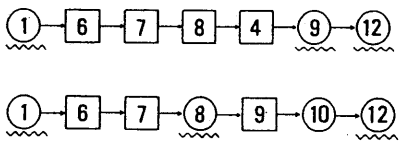

図２．無駄な要素を持つマッチングパターン

□はマッチングの成功、○は失敗を表わし、数字はノード番号に対応する。

ここで多くの無駄なマッチング失敗が含まれている。照合過程に必要な条件は、ある状態において全ての実行可能なルールを判別することであり、現在の推論サイクルのワーキングメモリの内容により、実行可能となるインスタンシエーション（具体例）を求めることができることである。よってルートからターミナルへのパスに含まれていないノード（図中波線部）で現されるルールパターン要素はデータパターンに含まれず、そのマッチングの失敗はルールの具体化には必要がない。

4-2. 排他的知識の利用

ここでルールパターンの排他的な知識をコンパイル時に考慮し、ネットワークにその関係を明示することによって、ネットワーク探索空間を動的に削減することができる（図３参照）[3]。排他的な関係にあるパターン要素において、一つの要素のマッチングが成功することによって他の要素のマッチングの失敗はマッチング行なう以前に分かる。

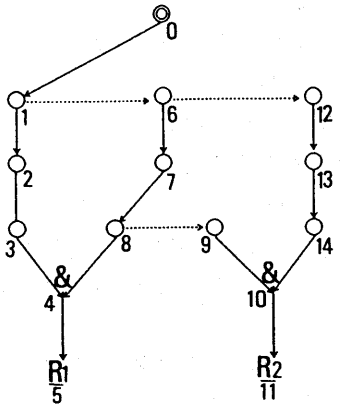

図３．排他的関係を明示したネットワーク

また排他関係にあるノードをネットワークの上流部に配置し、できるだけ早い時点で照合を行なうヒューリスティックが有効である（図４）。

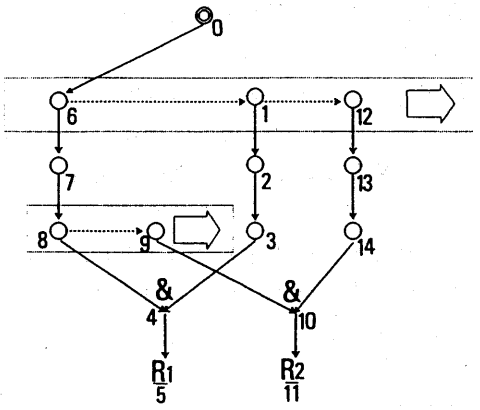

図４．ネットワークの成長方向

しかし、ルール数が増加し、ルールパターンが複雑化することによってパターン間の構造的類似性や排他性が満足されにくくなり、結果としてネットワークは横方向に成長し、ネットワークの横方向探索により生じる無駄なマッチングの失敗が増加する　（図４）。

またパターン間の排他性に関する知識はマッチングの結果には関係がなく、推論以前のコンパイ

ルではマッチングの早期成功を期待しているに過ぎず、ネットワーク上流においてマッチングが成功しない場合ネットワークの横方向探索に起因する無駄なマッチングは回避されない。

5.提案方法

5-1.方　針

さて本論文では推論中に生じる
(C)マッチングの結果に関する知識
(D)データパターンの構造的な類似性
を抽出し、保存することによって、残された無駄なマッチングを極力回避し、そしてさらに不必要なマッチングを回避する手法について述べる。本手法はReteアルゴリズムの適用が有効であるような推論1サイクルにより、データの変更がわずかしか行なわれないシステムを仮定している。このようなプロダクションシステムではデータベースの状態が比較的ゆっくりと変化するので、ルールの行動部により、発生するデータパターンは前後に発生するデータパターンが非常に類似した状況がしばしば起こる。そこで (C),(D)を利用したヒューリスティックが照合の効率化に有効である。その方針は (C),(D)各々について

(C)前回のパターンマッチングでマッチングに成功したルールパターン要素を優先してマッチングを行なう。

マッチング結果により、ネットワークを変形し、マッチング順序を線形縦型探索のまま現在推論の対象となっているデータに依存させて変更する。

(D)データパターンの相互の構造的類似性をマッチングに利用する。

以前行なったパターンマッチングにおいてマッチングを行なったルールパターン要素を保存し、ネットワークにより処理する前にデータパターン相互のマッチングを行なう。

これらのアイデアは推論中に生じるデータパターン要素とデータパターン間の知識を用いて探索すべきネットワークの範囲を動的に限定することにより、照合を効率化する事を目的としている。推論実行中に発生したデータに関する処理を行なうので、マッチング回数の減少分を越えないような、オーバーヘッドが少ない具体的方法が要求される。次章ではこれらの具体的方法である RINGS (Relative Inserted Nodes Gate System)を提案する。 RINGSは動的にネットワークを変形することにより、探索空間を強調し、限定するアルゴリズムである。

RINGSはネットワークに対する手法Ⅰ（強調化）と手法Ⅱ（限定化）からなる。手法Ⅰで判例を利用したマッチングをルールパターンネットワークにより行なう方法を提案し、手法Ⅱにおいてさらに手法Ⅰを効率化する。

5-2.具体的手法

5-2-1.手法Ⅰ：ネットワークの強調化

Reteアルゴリズムでは推論1サイクル単位の状態を保存することができるが1サイクル未満の部分的なマッチングに関する情報は保存されない。このことは照合処理では、データパターンはそれぞれ独立してパターンマッチングが行なわれることを意味する。しかし、類似したデータパターンが発生しネットワークに流れるとき、後続のデータパターンは先行するデータパターンの流れたパスとほぼ同様のパスを再度たどるために起こる無駄なマッチングが起こっている。そこで先行するデータフローの経験を後続のデータフローに利用できればこのような無駄なマッチングを回避することができる。

データフローの経験を保存するためにネットワーク構造をデータフローにより強調化する。そのためにに複数のノードからなる機能付きノードを定義し、それを利用することによりマッチング順序を変更し、無駄なマッチングを回避する。

[RN]リングノード

並列ノードをマッチングする過程においておいて最も最近マッチングに成功したノードにポインタをつなぎかえることにより、過去のデータパターンのたどったパスを保存する。

STEP 1 :ネットワークにおいて並列なノードをリング状に接続しリングと定義する。ポインタにより、他のリング、またはノードと接続する（図６）。

STEP 2 :ポインタを持つノードを始点としてリングを一巡してマッチングを行う。

STEP 3 :最も最近マッチングに成功したノードにポインタをつけ替える。

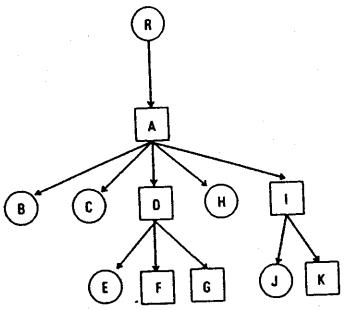

図５．ネットワークによるマッチング結果

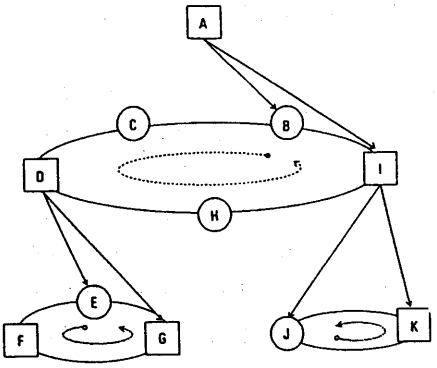

図６．リングノードを持つネットワーク

ポインタの変更により、データパターン１フローのパスが保存され、従来の縦型探索を適用することによって１フロー前のパスを優先的にたどることができる。またリングに排他的関係が明示されている場合リング内のマッチングはベストケースで１回である（図７）。

類似したデータパターンが流れてくる場合同分のマッチングは最小回数ですむ。

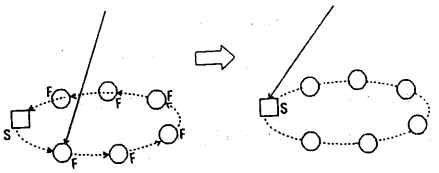

図７．排他的なリングノードのマッチング

次にリングノードの改良版であるゲートノードを示す。

[GN]ゲートノード

並列ノードをマッチングする過程においておいて最も最近マッチングに成功したノードを先頭に再配置することにより、過去のデータパターンマッチングに関する情報を保存する。

STEP 1 :ネットワーク内の並列要素をゲートと定義する、ポインタにより、他のゲートまたはノードと接続する（図８）。

STEP 2 :ゲート内の要素を線形順序でにマッチングを行なう。

STEP 3 :マッチングに成功した要素をゲートの先頭に再配置する。

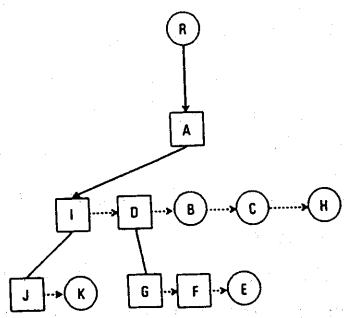

図８．ゲートノードを持つネットワーク

ゲートノードの利用することによって照合の履歴によりネットワークの構造を強調することができる。マッチングに使用されたルールパターン要素がネットワークの上流に移動するために、ルールによる判別と判例による判別が同一のパラダイム行なうことが可能となっている。

さらに、[RN]に比べ複数のデータパターンフローの結果を保存できるので、ルールによる判別に

先だって判例による判別が行なわれるためにパターンマッチングのワーストケースの発生頻度が低くなる。また、ゲートにおいて排他的関係が明示されているならば、[RN]以上に、効率化が可能である。

5-2-2. 手法Ⅱ：ネットワークの限定

手法Ⅰにより、無駄なマッチングを極力回避する方法を提案した。手法Ⅰでネットワークの上流においてデータパターン間の同分のマッチングを行なっているが、同分のマッチングはマッチングの結果が分かっているために、ネットワークを探索によって結果がすでに判明しているマッチングが繰り返される。手法Ⅱではデータパターン間の類似性に関する知識を利用することによって、ネットワークの探索空間の必要部分を限定する。

パターンP1<A D G ..>が先行して発生した場合、[GN] によりネットワークは図９のように変形される。

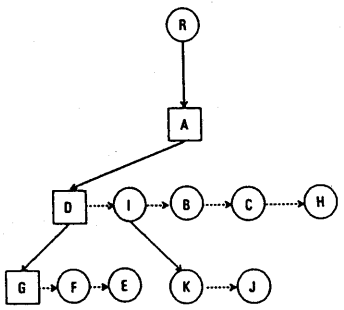

図９．ゲートノードによるマッチング

パターンP2<A D E ..>が流れる時探索されるときルールパターン要素は図１０で示される。P1とP2の同分に対応するパターンははP2が流れる前にP1のパターンマッチングにより、結果が分かっている。

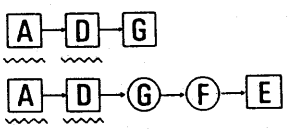

図１０．同分のマッチングパターン

そこで、データパターンにマッチングを行なったルールパターンに対応する情報を保存し、ネットワークによるパターンマッチングを行なう前に、先行して発生したデータパターンとのマッチングを行ない、ネットワークを短絡させるためのルートノードを動的に変化させる。

データパターンの各々の要素に対してマッチングメモリを付加し、ネットワーク探索を行なう。マッチングが成功した時点で対応するルールパターン要素ナンバーを代入する。図１１に手法Ⅱの考え方の概略を示す。一般にテンポラリルートナンバは探索されるパスの数に応じて複数存在することになる。

Pk(i) ：パターン要素
Mk(i) ：マッチングメモリ
TRk　：テンポラリルートナンバ

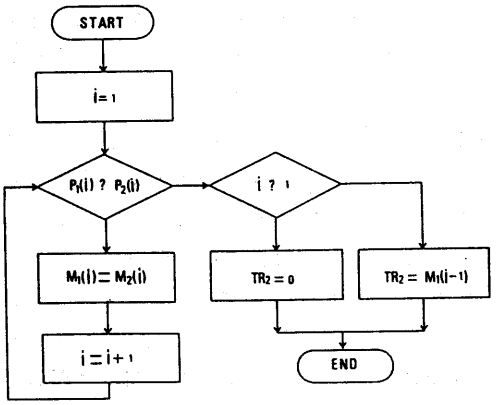

図１１．手法Ⅱフローチャート

マッチングメモリのデフォルト値は初期ルートノードを指す 0が入っている。よって比較するデータパターンがまったく異なる場合、ルートノードは変化しない。

TRk+1から Pk+1を流すことによって、結果の決定されていない要素のマッチングのみが行なわれる。このことにより探索すべきネットワークを限定することができる。よって縦方向探索による同分のマッチングの重複を削減することができる。

以上の手法Ⅰ[GN]に手法Ⅱを加えたアルゴリズムをRINGSとする。

6. RINGSの評価

6-1. Reteアルゴリズムとの比較

Reteアルゴリズムと RINGSの大きな違いはマッチング順序を動的に変更していることである。

RINGSの有効性はマッチング順序の変更によるマッチング回数の削減と、ネットワーク変形によるオーバーヘッドの抑制にある。

よって RINGSの適用は全マッチング数の減少に加え、

・ルール発火に不必要なマッチング数の減少
・全マッチング時間の減少

が特徴である。また、Reteアルゴリズムではネットワーク内にメモリノードが散在するのに対して、RINGS ではマッチングの結果、ネットワークは対象とするデータに関係の深いルールパターン要素がネットワーク上流に極在する構造に変化する。

また RINGSはReteアルゴリズムを拡張した方法であるので、 RINGSの最悪ケースがReteアルゴリズムとなっている。 RINGSはデータ変更が比較的ゆっくり起こるシステムに適用することを仮定としておいているので、発生するデータパターンがランダムであるような、データが推論の1サイクルで大きく変化するようなシステムでは提案したヒューリスティックが効果的でない。しかしそのような場合はReteアルゴリズムの適用に問題がある。よってReteアルゴリズムの適用が有効である場合 RINGSの適用は有効である。

6-2. 計算機によるシミュレーション

次に計算機によるシミュレーションの結果を表1に示す。対象としてモンキーとバナナを選んだ[5]。ルール数は 19ルール、ネットへのコンパイルはReteアルゴリズムに従い、さらに、ゲートノードが定義されている。全ネットワークノード数は約50である。このネットワークに3種類のマッチングアルゴリズム

(1) Rete :Reteアルゴリズム[2]、
(2) XRete :Reteに排他関係を明示[3]、
(3) RINGS :提案マッチングアルゴリズム

をSONY WORK STATION NEWS にC言語によりインプリメントした。シミュレーションの結果、全マッチング回数の減少そしてマッチングの失敗の減少が認められた。Reteアルゴリズムに対して RINGSではマッチング回数で約 75%マッチング実行時間で約 63%の効率化が達成され、マッチング回数の効率化とともに時間的な効率化もなされている。これは RINGSに用いた手法がシンプルなためオーバーヘッドが問題にならなかったことを示している。

	(1)RETE	(2)XRETE	(3)RINGS
SUCC	193	193	153
FAIL	207	131	109
SUCC	400	324	262
TIME	100	83(%)	75(%)

SUCC:マッチングの成功回数
FAIL:マッチングの失敗回数
TIME:RETEと比較した実行時間（比率）

7. むすび

RINGSは Reteアルゴリズムで利用されていないデータ間の構造的特徴を抽出することにより照合を効率化している。ネットワークへのコンパイルはReteアルゴリズムに従い、提案した手法はプロダクションシステムの特徴を利用したきわめてシンプルなものである。コンパイルの方法について多くの提案がなされているが、推論前のコンパイルは基本的にマッチングの成功を期待したもので、複雑なネットワークを構築するためにはコンパイル時のオーバーヘッドがより多くかかるかる。特にルールが追加的に増大するプロダクションシステムにおいてはひんぱんにルールを再コンパイルする事により、システム運用に関する効率が悪化

が懸念される。その点 RINGSの持つシンプル性はシステム運用の効率化をもたらし、新たなアイデアを付加しやすので、システムを効率的に発展させることができると予想される。

今後の課題として他の多くの問題へ適用し、より厳密な RINGSの評価があげられる。またマッチング照合効率化として

・変数間のマッチングの効率化

が残されている。変数間のマッチングに RINGSをそのまま適用できないが RINGS適用によるネットワークの変形の効果を利用することを現在検討中である。

参考文献

[1] McDermott, J., Newell, A. and Moore, J.: The Efficiency of Certain Production Implementation, in Watermann, D. A. and Hay-es. Roth . F. (eds.), Pattern Directed Interface Systems, pp. 155-176, Academic Press, New York(1978).

[2] Forgy, C. L:Rete:A Fast Algorithm for Many Pattern/Many Object Pattern Matching Problem , Artif. Intell., Vol. 19, pp. 17-37 (1982).

[3]荒屋ほか: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 Vol. 28, No. 7, pp768-775, July, (1987).

[4]石田、桑原 ：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, pp467-477(1988)

[5]田中、下井 ：エキスパートシステムの構築の方法 OPS83人工知能プログラミング入門、パーソナルメディア(1987).

[6]C. L. ﾌｫｰｷﾞｰ：人工知能用言語OPS83、パーソナルメディア(1986).

[7]Hayes, Waterman, Lenat:Building Expert Systems, Addison-Wesley Publishing Company, pp. 301-307, (1983).

[8]北川敏男 ：学習制御および、学習制御機械、共立出版, pp. 741-747(1966).

[9]荒屋ほか ：プロダクションシステムにおける効率的パタン照合のための連想ネットワーク表現、情報処理学会論文誌、vol. 29 No. 8, pp. 741-747, (1988).

[10]田野ほか：推論高速化のための弁別ネットワークの動的変形法、情報処理学会第33回（昭和61年後期）全国大会論文集、pp. 1417-1418 (1986).

[11]藤田ほか：前向きプロダクションシステムのための要求駆動型マッチングアルゴリズム、情報処理学会、計算機アーキテクチャ研究会資料(19-89. 3. 23).

[12]水上、植田：Reteネットワーク探索時の無駄な照合削減、電気関係学会中国支部第39回連合大会講演論文集、pp. 157(1988. 10. 23).

[13]水上、植田：プロダクションシステムにおけるデータ変更時の照合の問題点、電気関係学会中国支部第40回連合大会講演論文集、pp. 182(1989. 10. 29).