計算機アーキテクチャ　97－7
（１９９２．１２．４）

筆順に基づく漢字構造知識を用いる超並列向き手書き漢字認識方式

村上仁利　高橋義造
徳島大学工学部知能情報工学科
E-mail:{zintan,taka}@n30.is.tokushima-u.ac.jp

漢字構造を筆順に注目して，ストローク間の関係位置として確度を含んだ述語で表現し，この知識ベースを辞書として用いる．また漢字のセグメントデータから構造を示す確度付き述語を抽出し，字種ごとに並列にルールと比較を行う．個別の認識結果より総合評価をし，類似文字に対して繰り返し認識処理を行わせてシステム全体の認識結果とする．

Massively parallel hand-printed Kanji recognition using knowledgebase of character structure due to writing sequence.

Hitotoshi MURAKAMI　Yoshizo TAKAHASHI
Department of Information Science and Intelligent System,
Faculty of Engineering, Tokushima University
2-1 Minami-josanjima-cho, Tokushima 770, Japan
E-mail:{zintan,taka}@n30.is.tokushima-u.ac.jp

We propose a massively parallel hand-printed Kanji recognition system using knwoledgebase of character structure due to writing sequence. A set of predicates, which describe relative positions among strokes with certainty factors, are generated from segment analysis of an input Kanji character, and are broadcast to all processors. Each one of the processors is assigned knowledgebase of individual Kanji structure, and evaluates the certainty factor of matching to it. These values are collected at a master processor which makes final judgement.

1 はじめに

我々は，近い将来実現されるであろう超並列計算機を想定した応用プログラムの開発目標として，高精度の手書き漢字認識方式の研究を行っている [1] [2]．従来の漢字認識システムでは辞書として，入力漢字パターンと同じ大きさの多値パターンを用いることが多かった．しかしこのようなシステムでは入力パターンの変形，雑音などに弱く，十分な認識結果を得られない．そこで漢字を画像パターンとして扱わずに，ある種の構造を持った構造パターンと考え，漢字構造を知識ベースに用いて認識を行うことにする．漢字構造の表現方法として確度を持つ述語を用いた．本稿では，はじめに超並列について述べ，提案する認識システム，漢字構造の表現方式，構造特徴の抽出，個別の認識とシステム全体の認識，個別の漢字認識実験と説明する．最後に今後の課題を示す．

2 超並列計算機向きの認識方式

我々はプロセッサーやメモリー等の資源が無数にある並列計算機を超並列計算機とし，超並列向きアルゴリズムを高精度の計算結果を得るためには計算機の資源や処理時間を無制限に使っても良い方式と考える．つまり従来の並列処理とは考え方が異なっており，並列処理による処理時間の短縮を目的にするのではない．計算機の処理能力や資源などの制限によりとうてい実現できなかった高精度の処理を行う方式を研究する事が目的である．

そこで我々が提案する超並列向き手書き漢字認識システムを図1に示す．まずイメージスキャナーやタブレット等の入力システムよりパターンの２値データを得る．これから我々がセグメント呼ぶ，直線部分からなる漢字を構成する最小部品の抽出を行なう．抽出された結果はセグメントの２つの端点情報である．次にセグメントの端点情報を構造解析にかけてストロークの相対関係を抽出する．そしてその結果を用い，ルールによる個別認識を漢字を担当するプロセッサーで個々におこなう．個別の認識結果をマスタープロセッサーに送り，総合評価を行い最終の認識結果とする．

このシステムでは，各々の漢字を担当するプロセッサーごとに漢字の構造を表現するルールを持たせる．

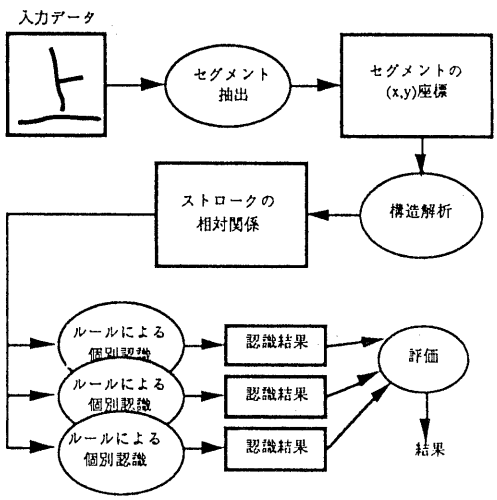


図 1: システム構成

この漢字ごとにルールとの比較を行なう処理は計算量が非常に多く，従来の並列計算機などでは計算が困難である．つまり，これは我々が目標とする超並列処理向きのシステムであるといえる．

3 漢字構造の表現方法

3.1 確度を用いた述語

物体などの構造知識を表現する方法として，意味ネットワークや述語を用いる方法などがある．[3] [4] [5] [6] 我々は漢字の構造表現にこれらの手法を応用し，ストロークの相対関係を述語を用いて表現する方法を提案する．この方法で用いる述語を表 1に示す．ここで用いられている確度とは，述語がどの程度正しいかを示す尺度であり，0〜100 までの整数値を取る．相対関係を表す述語には大きく分けて，ストロークに関する述語とノード(端点) に関する述語の 2 種類がある．さらにストロークに関する述語にはストローク自身の性質を表す述語と相対関係を表す述語の 2 種類がある．ノードに関する述語には相対関係を表す述語だけがある．

表 1: 構造表現のための確度付き述語

述語表現	意味 S:ストローク N:ノード
vertical(S,P)	S は垂直である
horizontal(S,P)	S は水平である
leftup(S,P)	S は左上がりである
rightup(S,P)	S は右上がりである
sleft(S1,S2,P)	S1 は S2 の左にある
sright(S1,S2,P)	S1 は S2 の右にある
supper(S1,S2,P)	S1 は S2 の上にある
slower(S1,S2,P)	S1 は S2 の下にある
smiddle(S1,S2,P)	S1 は S2 の間にある
strokes(N,P)	ストローク数は N である
node(N,S,P)	N は S の点である
nleft(N1,N2,P)	N1 は N2 の左にある
nright(N1,N2,P)	N1 は N2 の右にある
nupper(N1,N2,P)	N1 は N2 の上にある
nlower(N1,N2,P)	N1 は N2 の下にある

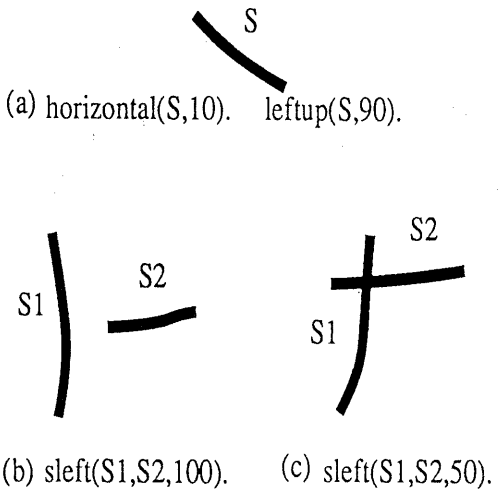


図 2: 確度付き述語の例

このように曖昧な知識を取り扱ったエキスパートシステムに MYCIN[7][8] [9] がある．MYCIN でも同様の信頼性係数 (Certainty factor) を用いる．この信頼生係数はルールに付加されるが，我々の提案する方式では CSA で抽出される述語に確度が付加される．またルールの信頼性係数は MYCIN では AND 結合のルールではルールを構成する述語の信頼性係数の最小値であり，我々の方式では平均値である．

図 2はストロークの性質と相対関係を付加した述語とストロークの概念図である．図 2(a) のストロークは同一のストロークの性質を 2 つの異なる述語で表現している．この場合，水平 (horizontal) の性質は 10%の確度を持ち，左上がり (leftup) の性質は 90%の確度を持つことを示している．また図 2(b)(c) のストローク対は，相対関係の sleft についての図である．ストローク対が重なっていない場合は 100%の確度であり，図 2(c) のように互いに重なっている場合には 50%の確度になっている．

次にこれら述語の確度の算出方法について述べる．ストロークの性質に関する述語の内，horizontal, vertical, leftup, rightup の確度は図 3の変換関数により求められる．ストロークの相対関係を示す述語は，図

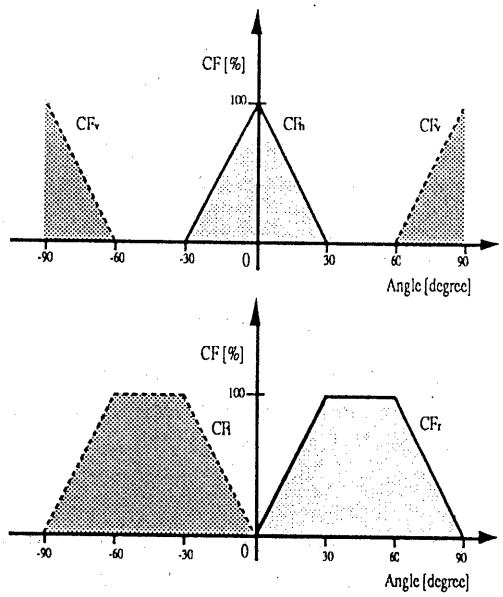


図 3: 確度の変換関数

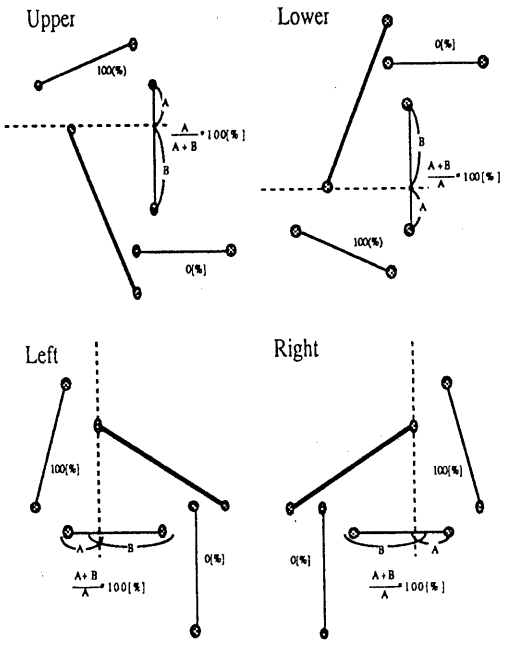


図 4: 相対関係を示す述語の確度の算出方法

4で示すとおり，述語が示す相対位置にストロークの一部分が存在する割合で定義される．今後は，述語に付加される確度の求め方について更に研究を進める予定である．

3.2 筆順による構造表現

漢字の構造を表 1の述語を用いて表現するのであるが，ストロークすべての相対関係を用いて漢字の表現をするのは効率が悪いばかりでなく，ルールの柔軟性を著しく害する．また辞書となるルールを作成するのが困難になる．しかし人間が漢字を書く時には，まだ書かれていないストロークとの相対関係すべてを考慮していると考えるよりはむしろ，新たなストロークを追加していく際にすでに書かれているストロークとの相対関係だけを考慮していると考えるのが自然である．つまりストロークの相対関係で漢字を表現する際に，ストロークを追加する特定の順にそれらの相対関係を追加すれば良いことになる．この特定の順序として筆順を用いることにする．筆順に従ってルールを作成すると無駄なルールを省ける，ルールの作成が容易になり間違いが少なくなる，等の利点がある．

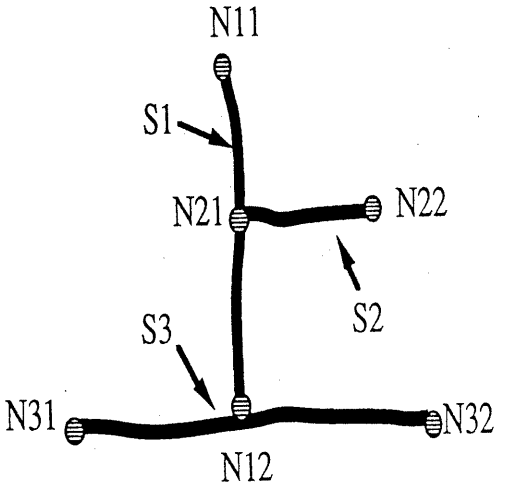
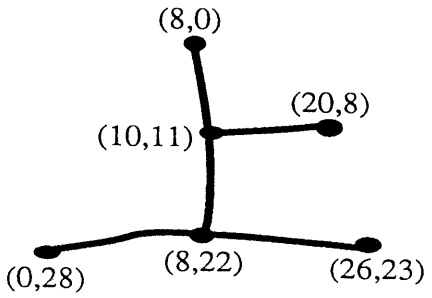


図 5: '上' のストローク

上 : −

$$
\begin{aligned}
&strokes(3, P1), vertical(S1, P2).\\
&horizontal(S2, P3), sleft(S1, S2, P4).\\
&smiddle(S2, S1, P5), horizontal(S3, P6).\\
&supper(S1, S3, P7), smiddle(S1, S3, P8).\\
&supper(S2, S3, P9),\\
&P\ is\ (P1 + P2 + \cdots + P9)/9.\\
&write('P =''), write(P).
\end{aligned}
\tag{1}
$$

図5は，'上' をストロークに分解してラベル付けした例であり，式1は表 1の述語を用いて表現したルールである．これを用いて '上' を構成するルールの説明を行なう．最初に '上' を構成するストロークの数を strokes(Num,P) で定義する．これは Num 個のストロークが '上' を構成する事を示す述語である．全てのルールは最初にストロークの個数を定義する．'上' を書くとき最初に書き始めるストロークは中心線にあたる縦棒であり，図 5では S1 である．このストロークは '垂直' という性質を持っている．この時点では相対関係を考慮するストロークが存在しないの



(8,0) -(8,22)
(10,11)-(20,8)
(0,28) -(26,23)

図 6: 入力パターンとセグメント情報

で，ストロークの性質の述語のみである．次にストローク S1 の横側にやや短めのストロークを水平に書く．このストロークを S2 とすると，S2 は水平の性質を持つ．また S1 との相対関係を示す述語には，S1 が S2 の左側にあるので sleft(S1,S2,P) のルールがあり，S2 が S1 の間にあるので smiddle(S2,S1,P) のルールがある．同様の手続きをストローク S3 にも行い，式 1のようなルールが完成する．このルールでは，最後に確度 P の計算を単純な平均で行なっている．確度の算出方法をどのようにするかは今後の課題である．

4 構造特徴抽出処理系 CSA

4.1 処理系としての CSA

図 1のシステム構成で，入力パターンのセグメント情報からストロークの相対関係などの構造特徴を抽出する処理がある．これを CSA(Character Structure Analyser) と呼ぶ構造解析処理で実現した．この CSA は入力として図 6のようなセグメントの端点情報を用いる．

4.2 CSA の実行例

図 6のセグメント情報は実際には表 2のようなデータ形式で CSA に入力する．この入力に対して CSA は表 3を出力する．例えば，stroke3 は本来 horizontal の性質だけを持つ．しかしこの CSA ではそれ以外にも異なる性質を出力する．先ほどの stroke3 では

表 2: CSA の入力情報

8	0	8	22
10	11	20	8
0	28	26	23

leftup の性質も出力されている．またストロークが持つ全ての端点やストローク間の相対関係も出力されている．たった 3個と少ないセグメント情報からこのように多くの述語が生成される．しかしながら全ての述語が認識に用いられるのではなく個別のルールで必要なものだけが参照される．

ストロークの相対関係や更に端点間の相対関係を示す述語は，ストローク数の増加にともない 2 乗のオーダーで増える．つまり非常に多数の述語が生成されるわけで，これをルールと比較して認識する処理は膨大な計算量といえる．生成される述語からも，これを用いる本方式は超並列でなければ実現できない方式であると考えられる．

5 個別ルールの競合による認識

各漢字を認識する方式として我々はプロセッサーごとに漢字の構造知識を持たせ，CSA の出力と比較をさせる方法を提案する．この方式の利点は，ある 1つの漢字を担当するプロセッサーがあり，それらは入力された漢字が自分が担当する漢字であると信じて認識をすることである．この特定の漢字を担当して行う認識処理を個別認識という．個別認識は 1つのスレーブプロセスに割り当てられる．個別認識を統括してシステム全体の認識結果を求める処理をマスタープロセスに割り当てる．この様子を図 7に示す．

個別認識を行うスレーブプロセスには担当する漢字のルールがある．このルールはすでに述べた確度付きの述語で表現された漢字の構造知識である．担当するプロセッサーでは入力されたパターンから CSA を用いて述語表現された事実を得る．ルールは確度を集計する事ができる推論型処理系で利用され，ルールが必要とする事実が入力されたパターンから生成された事実に存在するかどうかが検証される．全てのルールが満たされた場合，担当する漢字の確度が

表 3: '上' に対する CSA の出力

```
horizontal(stroke2,76).
leftup(stroke1,100).
leftup(stroke3,92).
nleft(node1,node3,100).
nleft(node1,node6,100).
nleft(node2,node4,100).
nleft(node3,node4,100).
nleft(node4,node6,100).
nleft(node5,node2,100).
nleft(node5,node4,100).
nlower(node2,node1,100).
nlower(node2,node4,100).
nlower(node2,node6,100).
nlower(node3,node4,100).
nlower(node5,node1,100).
nlower(node5,node3,100).
nlower(node5,node6,100).
nlower(node6,node3,100).
node(node1,stroke1,100).
node(node3,stroke2,100).
node(node5,stroke3,100).
nright(node1,node5,100).
nright(node2,node5,100).
nright(node3,node2,100).
nright(node4,node1,100).
nright(node4,node3,100).
nright(node6,node1,100).
nright(node6,node3,100).
nright(node6,node5,100).
nupper(node1,node3,100).
nupper(node1,node5,100).
nupper(node2,node5,100).
nupper(node3,node5,100).
nupper(node4,node2,100).
nupper(node4,node5,100).
nupper(node5,node2,100).
nupper(node6,node5,100).
shmiddle(stroke1,stroke3,100).
shmiddle(stroke2,stroke3,100).
shmiddle(stroke3,stroke2,34).
svmiddle(stroke1,stroke2,10).
svmiddle(stroke1,stroke3,15).
svmiddle(stroke2,stroke1,100).
supper(stroke1,stroke2,38).
supper(stroke1,stroke3,84).
supper(stroke2,stroke3,100).
horizontal(stroke3,92).
leftup(stroke2,76).
nleft(node1,node2,100).
nleft(node1,node4,100).
nleft(node2,node3,100).
nleft(node2,node6,100).
nleft(node3,node6,100).
nleft(node5,node1,100).
nleft(node5,node3,100).
nleft(node5,node6,100).
nlower(node2,node3,100).
nlower(node2,node5,100).
nlower(node3,node1,100).
nlower(node4,node1,100).
nlower(node5,node2,100).
nlower(node5,node4,100).
nlower(node6,node1,100).
nlower(node6,node4,100).
node(node2,stroke1,100).
node(node4,stroke2,100).
node(node6,stroke3,100).
nright(node2,node1,100).
nright(node3,node1,100).
nright(node3,node5,100).
nright(node4,node2,100).
nright(node4,node5,100).
nright(node6,node2,100).
nright(node6,node4,100).
nupper(node1,node2,100).
nupper(node1,node4,100).
nupper(node1,node6,100).
nupper(node3,node2,100).
nupper(node3,node6,100).
nupper(node4,node3,100).
nupper(node4,node6,100).
nupper(node6,node2,100).
sleft(stroke1,stroke2,100).
sleft(stroke3,stroke1,47).
sleft(stroke3,stroke2,50).
slower(stroke1,stroke2,51).
slower(stroke3,stroke2,100).
sright(stroke2,stroke1,100).
sright(stroke3,stroke1,52).
sright(stroke3,stroke2,15).
strokes(3,100).
vertical(stroke1,100).
```

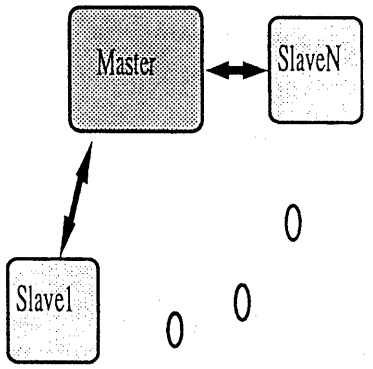


図 7: 競合による認識システム

表 4: マスターのステータステーブル

番号	状態	引数	確度
1	動作	-	-
2	競合	15	92 %
3	終了	0	21 %

アルゴリズムに従い求められる.

マスタープロセスでは, 各スレーブプロセスの動作状況をイベントドリブン形式で管理する. スレーブプロセスの動作状況を管理するテーブルとして表4のようなステータステーブルを用いる. ステータステーブルにはルール番号, 求められた確度, 現在の状態, 状態の引き数がある.

状態には動作, 競合, 終了の3種類がある. '動作' では個別の認識処理を行い, '競合' では2つのスレイブ間での競合が行われる. '終了' 状態は認識処理が完了している事を示す. 引数はそれぞれの状態をより詳しく説明する情報で, '競合' では競合しているルールの番号, '終了' では結果の有効 (1)/無効 (0) を示す. '動作' では引数は不定である.

マスタープロセスにはステータステーブル以外に, ルールの競合を管理する競合テーブルを持つ. このテーブルはスタック形式であり, スレーブプロセスからの認識結果で確度がある一定のしきい値を越え

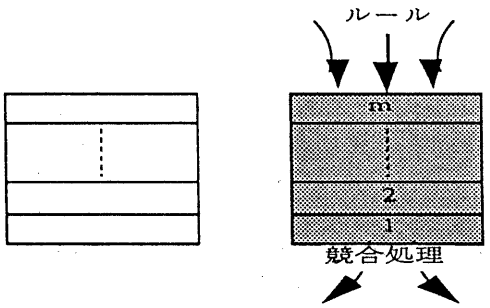


図 8: 競合テーブル

ていればスタックされる．競合テーブルにスタックされているルールが2つになった時，スタックから2つのルールが選択されて競合が行われる．この様子が図8に示されている．同時にステータステーブルの状態を競合に変更する．またマスターからスレーブに対してはどのスレーブと競合するのかを送信する．競合状態になったスレイブでは競合する相手のルールを取得し，再び認識処理を行う．この結果をマスターに送信する．この処理を繰り返し行う．マスターは全ての処理が終了した事をステータステーブルで確認した後に最終の認識結果を出力する．

これらの競合方式を計算機上にインプリメントしその有効性を検証する予定である．また競合処理を行う方法についても考察中である．

6 推論型処理系による認識実験

6.1 実験方法

現時点ではワークステーション上でセグメントの端点情報を CSA によりストロークの相対関係と端点の相対関係の述語に変換し，Prolog のルールで構造表現された漢字のルールとマッチングを行い認識する実験を行った．実験方法の詳細を以下に示す．

実験は SUN SPARC Station 2 上で行い，X-Window を使用して文字パターンの作成をした．作成したパターンから端点情報を手作業で取得し，セグメントを直線で近似した漢字データを作成した．このデータをセグメントの端点情報という．本来ならば手書きの漢字をイメージスキャナなどで取り込み，画像データから2値化したパターンからセグメントの抽出をして，端点情報を取得すべきである．しかし今回は本方式が超並列計算機向きの方式であることを調査するためのシミュレーションであるため，セグメントの端点情報はすでに得られていると仮定した．

今回の実験の目的は，ストロークの相対関係で漢字の構造を表現したルールを用いた認識方式の有効性と，超並列処理で行うことによる認識率の向上の検証が目的である．このシミュレーションでは以下のような取り決めと制約を設けているので，認識システム全体をシミュレーションしているわけではない．

- ストロークの性質を示す述語 (vertical, horizontal, leftup, rightup) を求める際の確度を絶対座標で固定した．つまり文字パターンの回転を考慮していない．
- ストロークや端点の相対関係を示す述語も入力パターンの絶対座標で算出している．
- 端点の相対関係を示す述語は確度が 100 % で固定されているので，漢字を表現するルールの確度を求めるのには利用しない．また可能な限りルールにも用いない．
- 認識実験に用いる推論型処理系として Prolog を用いる．述語の確度を求めるための処理を漢字構造を表現するルールに付加した．
- 辞書である漢字構造を表現するルールとして類似文字を数種作成した．
- ルールの確度を算出するアルゴリズムは前記の通り述語の確度の平均値とする．
- 最終結果として，CSA の出力する述語を事実とした場合のルールの確度が得られる．
- 最終結果で確度の最も高いルールを認識結果とし，この文字の正誤で認識率を求めた．

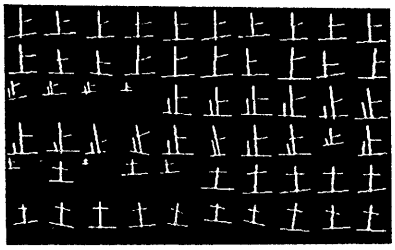


図 9: 端点情報から復元した入力パターン

表 5: パターン'上'の認識結果

パターン	CF(上) [%]	CF(止) [%]	CF(土) [%]	結果
1	92.5	-	-	O
2	93.8	-	-	O
3	92.8	-	-	O
4	94.8	-	95.7	X
5	96.4	-	-	O
6	92.5	-	-	O
7	93.8	-	-	O
8	92.8	-	-	O
9	94.8	-	95.7	X
10	96.4	-	-	O
11	95.7	-	93.1	O
12	91.6	-	95.1	X
13	94.7	-	-	O
14	94.4	-	-	O
15	95.2	-	-	O
16	93.9	-	-	O
17	97.1	-	-	O
18	95.5	-	-	O
19	91.9	-	93.2	X
20	95.8	-	94.4	O

表 6: 認識結果

ルール	認識率
上	80.0 %
止	100.0 %
土	100.0 %
平均	93.3 %

6.2 実験結果

以上の条件で数種のルールを用いた認識実験を行った. 図9にセグメントの端点情報から復元した入力パターンを示す. それぞれのセグメントは直線で近似されている. また上段の左側手より順に'上-1','上-2'と番号付けされている. 表5は, 20個の'上'のパターンをそれぞれのルールと比較した場合の確度と認識結果を示している. 表6に個別の認識結果を示す. 認識率はルール毎に競合をさせて認識を行なっていないのでやや低い認識率であるが, 前記の通りにルールを互いに競合させて類似文字の候補を絞り込めば非常に高い認識率が得られると考える.

7 おわりに

本研究では, 従来の並列計算機などでは実現できない方式でもプロセッサー台数の多い超並列計算機ならば実現できる方式として個別にルールを用いた本方式を提案した. この方式は漢字の構造を述語を用いてを表現する方式であり, 漢字の構造を簡単に表現できるといえる. またシステムの一部分を実験したのにすぎないが本方式が有望であることを示せたと考える. 今後は互いに競合してシステム全体の認識を行う方式のインプリメント, 確度を求めるための方式の検討, 漢字を表現するルールの自動作成などの課題がある.

参考文献

[1] 須原 康次他:2進木計算機による並列パターン認識システムについて, 信学技報, CPSY89-23, Vol.89, N0.166, 1989

[2] 佐々木一陽他:超並列手書き漢字認識の研究, 第43回情報処理学会全国体会, 論文集(文冊2), 2D-6, 1991

[3] 長尾 真:パターン情報処理, コロナ社,pp.160-161,1983

[4] 長尾 真:知識と推論, 岩波書店, pp.222-232, 1988.

[5] 安西 祐一朗:認識と学習, 岩波書店, pp.10-12, 44-45, 1989

[6] 太原 育夫:人工知能の基礎知識, 近代科学社, pp.73-103, 1988

[7] 白井良明 他:岩波講座 情報科学-22 人工知能, 岩波書店, pp.170-173, 1982

[8] AIUEO訳:エキスパート・システム, 産業図書, pp.104-109, 1985

[9] 白井良明:人工知能の理論, コロナ社, pp.141-143, 1992