進化的手法に基づく再構成可能な推論ハードウェア

安永守利[†] 高橋雅聡[††] 吉原郁夫[†††]

FPGA（Field Programmable Gate Array）等の再構成可能な集積回路を利用した推論ハードウェアの設計手法を提案する．本手法では，事例データベースを基に真理値表を作成し，この真理値表が未知事例データ（質問）も正しく包含するように変換する．この変換のためのオペレータの決定に遺伝的アルゴリズムを使用し，そのための遺伝子と染色体構造，および遺伝的操作を提案する．オペレーションを施した真理値表（汎化能力を持つように進化した真理値表）から展開されたANDゲート群に，推論精度を向上させるためのカウンタ，最大値検出回路を付加して推論ハードウェア（専用集積回路チップ）を構成する．本提案手法を英単語の発音記号推論チップの設計に適用し，その性能を評価した．その結果，回路規模266,388ゲート，1音素あたりの推論時間（推論速度）500 nsで推論精度81.9%を得た．この推論精度は従来技術である記憶ベース推論とほぼ同等であるが，推論速度は高速で，かつ，ハードウェア量もはるかに小規模である．本提案手法により，推論問題の並列性を直接的にハードウェアに埋め込み，かつ，問題の状況に応じて適応的に再構成できる推論チップが可能となる．

Reconfigurable Reasoning Hardware by Using Evolutional Algorithm

MORITOSHI YASUNAGA,[†] MASATOSHI TAKAHASHI[††]
and IKUO YOSHIHARA[†††]

We propose a new design methodology for reasoning hardware using reconfigurable VLSIs such as FPGAs (Field Programmable Gate Arrays). In the methodology, case database is transformed to the truth table. By using the genetic algorithm, sets of operators (genes), which generalize the truth table to include unknown case data correctly, are evolved with newly proposed chromosomes. The reasoning hardware is basically constructed by transforming the evolved truth table to the AND gate circuits. Counters and maximum-value-detector circuits are added to the AND gate circuits to increase the reasoning accuracy. English pronunciation reasoning (EPR) chip, reasoning accuracy and reasoning time of which are 81.9% and 500 ns/phoneme, respectively, is designed by this methodology containing 266,388 gates. The EPR chip shows higher reasoning speed, smaller hardware size and the same reasoning accuracy comparing with the traditional memory-based reasoning approach. The reasoning chips that involve intrinsic parallelism in each task and are reconfigured corresponding to varying tasks can be designed by the proposed design approach.

1. はじめに

規則を完全に把握することが困難な推論問題に対して，従来より事例ベース推論や記憶ベース推論，あるいはニューラルネットワーク等の手法が用いられてきた[1)~7)]．これらの手法は，規則を明示的に与える必要がない反面，事例データの統計処理や学習に多大な計算時間を必要とする．このため，大規模な推論問題に対しては，超並列計算機やベクトル計算機といった大規模な高性能計算機が必要であった．このような背景から，大規模な推論問題を小規模なハードウェア（1チップ，あるいはパーソナルコンピュータの拡張ボードの規模）で高速に処理することが望まれている．

1チップ，あるいは1ボードレベルの小規模なハードウェア量で大規模推論問題に対する高速な推論ハードウェアを実現するには，「各推論問題の事例データセットの並列性（事例データ数と同じ並列度）をハードウェア上に直接反映する」必要がある．このために，事例データセットから直接回路を生成する新たな回路設計手法を提案することが望まれる．しかし，このようなデータを基本とする設計（Data-based Design）コンセプトは，従来より広く使用されているゲートア

† 筑波大学電子・情報工学系
Institute of Information Sciences and Electronics, University of Tsukuba
†† 株式会社日立超LSIシステムズ
Hitachi ULSI Systems, Co., Ltd.
††† 宮崎大学地域共同研究センター
Cooperative Research Center, Miyazaki University

レイやスタンダードセル等の集積回路とは馴染まなかった．なぜなら，これらはフォトマスクを用いて汎用あるは準汎用な集積回路を大量に生産する方式であり，したがって，問題ごとに推論ハードウェアを個別設計し，かつ，製造後も問題の変化（データの追加等）に対応して迅速に修正を加える設計手法には適さないためである．

一方，ここ 2〜3 年の間に FPGA（Field Programmable Gate Array）や CPLD（Complex Programmable Logic Device）等の再構成可能な集積回路の集積度は飛躍的に向上しており，50 万ゲートレベルのものが市販されるに至っている[8),9)]．これらの集積回路は，もともと Rapid Prototyping 等を目的とした試作や開発のためのツールとして利用されていた．しかし，この再構成可能な超高集積回路を利用することにより，上述した新たな回路設計手法が現実的なアプローチとして見えてきた．

本論文の目的は，FPGA/CPLD 等の集積度の増加を背景に，以下の事項を満たす推論ハードウェアの新たな設計手法を提案することである．

(1) 従来推論方法と同程度の推論精度を有する．
(2) 従来必要とされていたハードウェア規模（超並列計算機やベクトルプロセッサ）よりはるかに小型で高速である．

本論文で提案する設計手法を LoDETT（Logic Design with Evolved Truth Table）と呼ぶ．LoDETT の基本的な考え方は，以下である．

(1) 事例データをゲート回路展開が可能なように真理値表で表現する（2 値化表現する）．
(2) この真理値表を，未知事例データも正しく包含する真理値表（未知事例データを正しいカテゴリに類別できる汎化能力を持った真理値表）に変換する．ここで，変換のためのオペレータの決定に遺伝的アルゴリズム（GA: Genetic Algorithm）[10)]を用いる．
(3) オペレータを実行した真理値表を回路化し，さらにこの回路に推論精度を向上するための多数決回路（カウンタと最大値検出回路）を付加することで，データレベルの並列度を持つ推論回路を実現する．

本提案手法では，事例データを真理値表にマッピングし，この真理値表に汎化能力を持たせた後に回路化を行う．ここで，汎化のためのオペレータ探索は膨大な探索空間が対象になり，ランダム探索では解を得ることが困難である．このため，スキーマタを見つけて効率良く解を探索する GA を用いる．

なお，近年研究が進んでいる‘進化するハードウェア’の中で，ゲートレベルの進化を対象とするアプローチで様々な開発事例が報告されている[11),12)]．本手法がこれらの手法と異なる点は，本手法ではハードウェアを進化させるのではなく，データを進化操作の対象ととらえ，データを進化させた後にこれをハードウェア化する点である．しかし，再構成可能な集積回路と遺伝的アルゴリズムを利用している点で，本手法も広い意味での進化するハードウェアの一部と考えることができる．

本論文では，はじめに LoDETT に関してその詳細を述べる．次に，LoDETT を用いて試作した「英単語の発音記号推論チップ」の設計とその性能評価結果を示す．

2. LoDETT

本章では，LoDETT 全体の基本的な処理フローについて述べる（図 1）．LoDETT の中核である真理値表を基にした推論方法とそのための進化手法については，次章でその詳細を述べる．

事例データベース（Case Database）は，過去の事

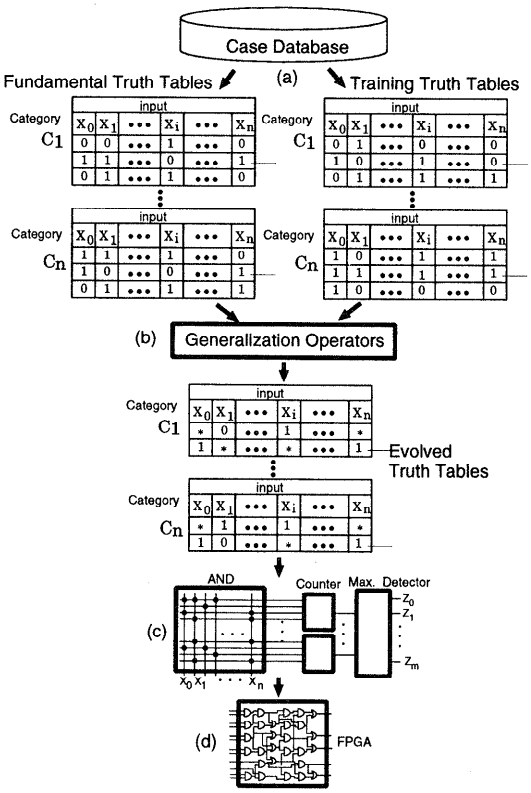


図 1 LoDETT の基本処理フロー
Fig. 1 Fundamental flow of LoDETT.

例データとそのカテゴリから成る．たとえば，事例データ{熱＝1 & せき＝1 & 頭痛＝1 & 歯痛＝0}はカテゴリ'風邪'に分類されている．LoDETTでは，はじめに事例データベースを2つに分割し，分割した事例データベースから2種類の真理値表を生成する（図1(a)）．1つは基本真理値表（Fundamental Truth Tables）であり，もう1つは学習用真理値表（Training Truth Tables）である．基本真理値表は回路生成のベースになる真理値表である．この基本真理値表が学習用真理値表も包含するようにするためのオペレータをGAで決定する．このGA操作の中で，オペレータは未知の事例データも正しく包含する能力を獲得する（汎化オペレータとなる）．この汎化オペレータを実行した後に回路生成を行う．

なお，後述するように，これら2つの真理値表は用途は異なるが構造はまったく同一である．基本真理値表と学習用真理値表は，すべてのカテゴリC_1～C_nに対して個別に生成される．真理値表の1つの行は，2値化された1つの事例データに相当する．簡単な例として，カテゴリ'風邪'とカテゴリ'虫歯'の事例データの真理値表は，**表1**のように構成することができる．

ここで，基本真理値表に対する汎化オペレータ（Generalization Operators）を導入する（図1(b)）．汎化オペレータは，基本真理値表の一部を'don't care'に変換することによって，基本真理値表が同じカテゴリの事例データだけをより多く包含するように変換する．この汎化オペレータを遺伝的アルゴリズムによって決定する．具体的には，基本真理値表が

(1) 同じカテゴリの学習用真理値表のより多くのデータ（行）を包含する

ように，また，

(2) 異なるカテゴリの学習用真理値表のより多くのデータ（行）を包含しない

ように遺伝的アルゴリズムによる学習操作によってオペレータを決定する．基本真理値表が学習用真理値表を正しく包含するように学習する過程の中で，未知事例データをより正しく包含する汎化オペレータが獲得できる．なお，本論文ではオペレータを実行した基本真理値表を'進化した真理値表（Evolved Truth Table）'と呼ぶ．本論文では，真理値表の構造に適合し，さらに追加事例データにも対応できる遺伝子（オペレータ）と染色体構造を提案した．これらに関しては，次章でその詳細を述べる．

次に，進化した真理値表を基に回路を作成（合成）する（図1(c)）．真理値表の各行はANDゲートによって展開することが可能である（積項による真理値の標準展開）．各ANDゲートに質問（未知データ）を入力することにより質問と各行（各事例データ）との一致検出を行うことができる．この一致検出結果を利用して，以下のように推論を行う．各カテゴリC_1～C_nの中で活性化されたANDゲートの数（進化した真理値表の中で，未知データと一致した行の数）を各カテゴリごとにカウンタ（Counter）でカウントする．次に，最大値検出回路（Max. detector）で最大値を示すカウンタ（Counter）とそのカテゴリを検出し，最大値を示すカテゴリを推論結果とする（一致したANDゲートの数が最も多いカテゴリを推論結果とする）．なお，このようなカウンタと最大値検出回路による推論は，例外事例データによって推論精度が低下するという問題を解決するために本論文で提案した手法であり，次章でその詳細を述べる．

ここで，前述したように，実装にFPGA等の再構成可能な集積回路を用いることによって，新たに事例データが追加されたり修正されたりした場合，上記プロセスを再び実行することによってつねに最新の事例データに適合する推論チップを再構成することができる（図1(d)）．

表1 事例データの真理値表の例
Table 1 Examples of truth table constructed from the case database.

カテゴリ'風邪'

入力			
熱	せき	頭痛	歯痛
1	1	1	0
1	0	1	0
1	1	0	0
.....			
.....			

カテゴリ'虫歯'

入力			
熱	せき	頭痛	歯痛
0	0	0	1
1	0	0	1
.....			
.....			

3. 真理値表を基にした推論方法とそのための進化的手法

本章では，真理値表が適応変化して汎化能力を得るとはどのようなことかを説明する．続いて，汎化能力を得る（進化した真理値表を作る）ための汎化オペレータのGAを用いた決定方法について述べる．

真理値表の各行の適応変化の考え方を図**2**に示す．真理値表の各行は，1つのANDゲートに変換される（図2(a)）．この例の場合，ANDゲートの

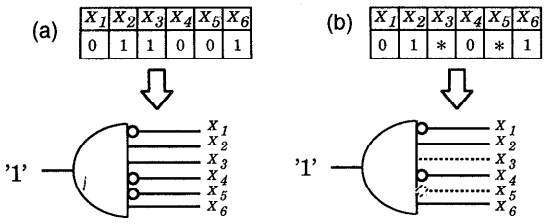


図2 真理値表の各行の適応変化
Fig. 2 Adaptation of a row in the truth table.

入力ピン数は 6 であり，X_2，X_3，X_6 が正入力であり，その他が負入力である．この AND ゲートと一致する入力（この AND ゲートを活性化する入力）は，このままでは，$\{X_1, X_2, X_3, X_4, X_5, X_6\} = \{0, 1, 1, 0, 0, 1\}$ のみである．一方，この行のフィールドの一部を図 2 (b) のように don't care で置換した場合，$\{X_1, X_2, X_3, X_4, X_5, X_6\} = \{0, 1, *, 0, *, 1\}$ で表されるすべての入力と一致する．これは，この AND ゲートが $\{X_1, X_2, X_3, X_4, X_5, X_6\} = \{0, 1, *, 0, *, 1\}$ で表されるすべての入力を包含し，同一カテゴリと類別するように変化したことを意味する．この類別が正しければ，この AND ゲートは汎化能力を得たことになる．LoDETT では，基本真理値表に汎化能力を持たせるための汎化オペレータを遺伝的アルゴリズムによって決定する．

以下にその詳細を述べるが，各個体の染色体は基本真理値表に対応した構造を持ち，各遺伝子座は真理値表の 1 つのフィールドに対応する．染色体が汎化オペレータであり，染色体は基本真理値表の全フィールドと同数の遺伝子座から構成される．各遺伝子座は対応するフィールドの値を保存するかあるいは 'don't care' に変更するかのどちらかのオペレーションを実行する．各フィールドに対して，どちらのオペレーションを実行するかを GA によって決定する．

3.1 汎化オペレータ決定のための GA

(1) 遺伝子コーディング

カテゴリ $C_1, C_2, \ldots, C_n$ に対し，それぞれ N 個体からなる集団を用いて，カテゴリごとに独立して進化を行う．各個体の染色体（Chromosome）の構造を**図 3** に示す．染色体は，基本真理値表（Fundamental Truth Table）の全フィールド数（行数 × 列数）と同数の遺伝子座から構成され，遺伝子座は基本真理値表のフィールドと 1 対 1 対応させる．図 3 では分かりやすくするために，染色体を 1 次元配列（ストリング）で表し，基本真理値表との対応を示している．このような真理値表のサイズに合わせた可変長染色体を用いることによって，集積回路作成後に新たな事例データが追加されても，その分ストリングを長くすることで容易に追進化と回路修正が可能となる．

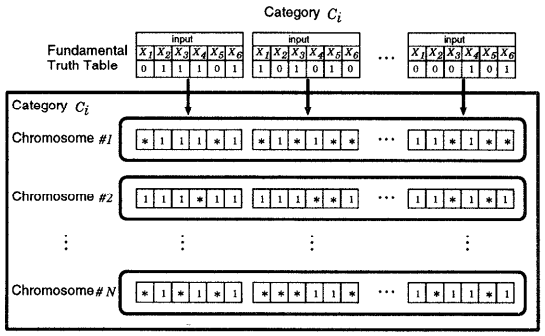


図3 染色体の構造
Fig. 3 Structure of chromosome.

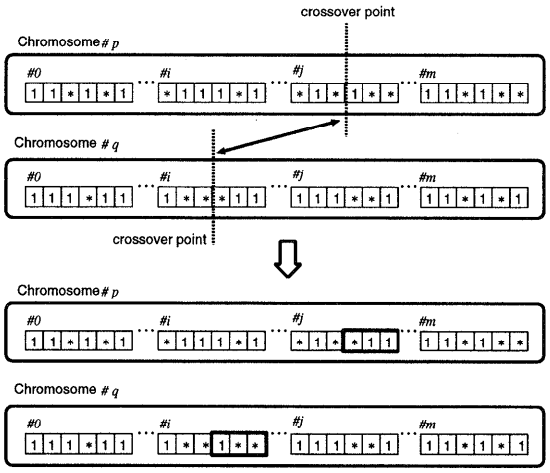


図4 染色体の交叉
Fig. 4 Crossover of chromosomes.

各遺伝子座は，2 つの記号 '1' または '*' のどちらかが占める．'1' は対応するフィールドの値を保存する操作を行い，'*' は対応するフィールドの値を don't care に変更する操作を表す．初期個体の染色体は，'1' または '*' をランダムに発生させて作る．

(2) 交叉

交叉を**図 4** に示す．はじめに，2 つの親個体 p と q をランダムに選択する．次に，親 p の j 番目の行と親 q の i 番目の行をランダムに選択する．最後に，選択された行の中で交叉点（crossover point）をランダムに選び 1 点交叉させる．

(3) 突然変異

突然変異操作は，全個体のいくつかの遺伝子座をランダムに選択し，オペレータをもう 1 つのオペレータと変更する（'1' なら '*' に，'*' なら '1' に変更する）．

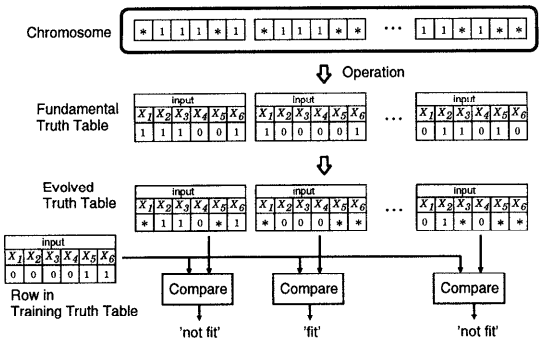


図5 染色体の評価

Fig. 5 Evaluation of chromosome.

(4) 適応度評価

個体の適応度の評価は，各カテゴリ C_i ごとに独立して以下のように行う（図 5）．基本真理値表（Fundamental Truth Table）に対して個体の染色体が意味するオペレーション（Operation）を実行し，進化した真理値表（Evolved Truth Table）を作る．すべての学習用真理値表（カテゴリ C_1〜C_n）から各行（Row in Training Table）を1つずつ取り出し，これが進化した真理値表（Evolved Truth Table）の各行に一致するか否かを検査（Compare）する．検査の結果，進化した真理値表の中に1つでも一致する行（図中の出力 'fit'）がある場合，この学習用真理値表の行は，進化した真理値表のカテゴリに包含されている．一方，検査の結果，進化した真理値表のすべての行と一致しなかった場合（すべての出力が 'not fit'），この学習用真理値表の行は，進化した真理値表のカテゴリに包含されていない．

検査に用いた学習用真理値表の行が，進化した真理値表と同じカテゴリであり，かつ，進化した真理値表に包含された場合，正しく推論が行われたことになる．また，検査に用いた学習用真理値表の行が，進化した真理値表と異なるカテゴリであり，かつ，進化した真理値表に包含されなかった場合，正しく推論が行われたことになる．

以上の結果から，各カテゴリ i の染色体（個体）の適応度を評価する．適応度の評価関数 f_i として，たとえば，

$$f(i) = \frac{1}{2}\frac{n_i}{N_i} + \frac{1}{2}\frac{n_i m_i}{N_i M_i} = \frac{n_i}{N_i} \times \left(1 + \frac{m_i}{M_i}\right) \times \frac{1}{2} \quad (1)$$

があげられる．ここで，N_i はカテゴリ i に属する学習用真理値表の総行数であり，n_i は，そのうち i に属すると正しく推論できた行数である．また，M_i はカテゴリ i に属さない学習用真理値表の総行数であり，m_i は，そのうち i に属さないと正しく推論できた行数である．評価関数の第1項は，カテゴリ i をどれだけ正しく推論できたかを評価しており，第2項はさらにその評価値に i でないカテゴリを正しく推論できたかを重み付けしている．それぞれの項は同じ割合（1/2ずつ）で評価値に寄与する．後述する試作では上記評価関数を用い，個体選択にはランキング選択を用いた．

4. 推論精度の向上（例外事例データを含めた推論）

上述した遺伝的操作によって決定した最優良個体を使って，各カテゴリごとに最終的な進化後の真理値を作成する．小規模な推論問題に対する推論回路は，進化した真理値表から AND ゲート群を作成し，**図 6** に示すように各カテゴリごとの AND ゲート群を OR ゲートでまとめることで設計することができる．AND ゲート群は，未知事例である質問（Query）に対する一致検出を完全並列に実行する．OR ゲートは各カテゴリ中の活性化した（質問と一致した）AND ゲートの有無を検出する．OR ゲートの出力が '1' であるカテゴリをもって推論結果とすることができる．

しかしながら，図 6 に示す回路では，以下の理由により事例データの中に例外事例データがあると正しい推論が困難となる．簡単な例として，4変数の事例データ空間 $\{X_1, X_2, X_3, X_4\}$ の中で $X_1 = 1$ であることが特徴であるカテゴリ C_i に対して，1つの AND ゲート $\{X_1, X_2, X_3, X_4\} = \{1, 1, 1, *\}$ が形成されたとする．これは，事例データ数 $2^4 = 16$ 個の空間の中で $X_1 = 1$ となる8個の事例データのうち，2個の事例データ $\{1, 1, 1, 0\}$ と $\{1, 1, 1, 1\}$ を検出する．

一方，カテゴリ C_i の例外事例データとして $\{0, 0, 1, 1\}$ があった場合，この例外データも包含するように進化した個体によって $\{X_1, X_2, X_3, X_4\} = \{*, *, 1, *\}$ のような回路が形成されることがある．この AND ゲートは，16個の全事例データのうち，異なったカテゴリのデータも含めて8個のデータによって活性化されてしまい，そのうちの約半分が誤活性である．図 6 の回路では，1つでもこのような誤活性 AND ゲートが発生すると回路全体（すべてのカテゴリ）で複数個の OR ゲートの出力が '1' となる．このため，どのカテゴリかが判定できず推論不可能となる．実応用レベルの大規模推論問題では，通常，各カテゴリに例外事例データが含まれるので，図 6 に示す回路では推論不可能なケースが頻出し，実用的な推論が

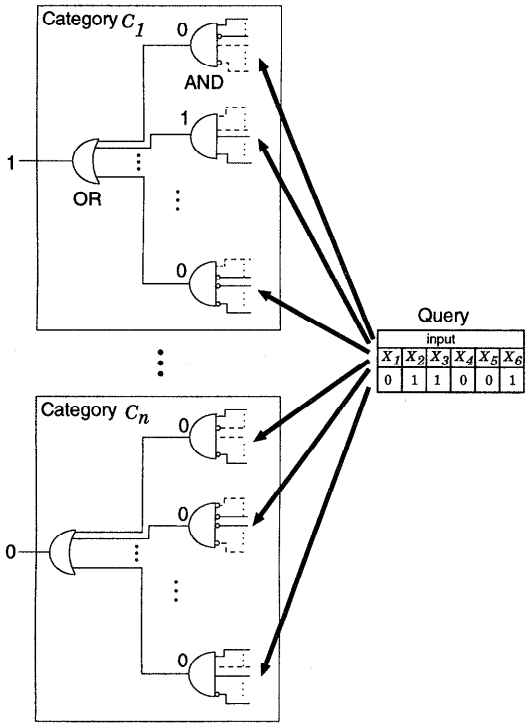


図6 ORゲートを用いた並列推論
Fig. 6 Parallel reasoning with OR-gates.

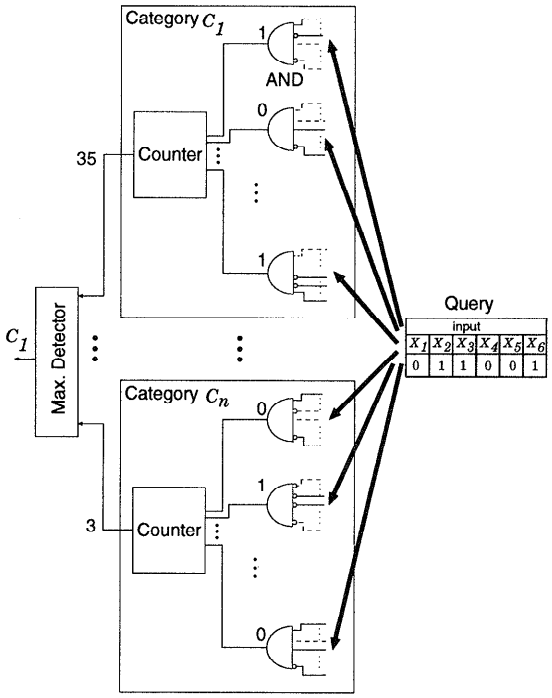


図7 カウンタと最大値検出回路による並列推論
Fig. 7 Parallel reasoning with counter and Max. detector circuits.

困難となる．

この問題を解決するために，**図7**に示す方法を提案した．入力された質問（Query）に対して，各カテゴリごとに設けたカウンタ（Counter）によって活性化されたANDゲートの数をカウントする．次に，これらカウンタ値の中の最大値を最大値検出回路（Max. detector）によって検出し，最大値を出力しているカウンタのカテゴリを質問に対する推論結果とする．この方法では，活性化されているANDゲート数の最も多いカテゴリを最も確からしいとする．活性化されたANDゲート数の比率から推論結果を導くため，例外事例データの影響を大きく受けたANDゲートがあっても，推論結果はこれらの影響を受けにくい．

たとえば，先の例でカテゴリ C_i に対する回路が $\{1,1,1,*\}$，$\{1,*,*,1\}$，$\{*,*,1,*\}$ の3つのANDゲートで生成されたとする（3番目のANDゲート $\{*,*,1,*\}$ は，先に述べた例外事例データによる）．一方，$X_1=0$ が特徴であるカテゴリ C_j に対して2つのANDゲート $\{0,1,1,*\}$，$\{0,*,*,1\}$ が生成されたとする．ここで，未知事例データ $\{0,1,1,1\}$ が入力された場合，C_i の $\{*,*,1,*\}$ がこの未知事例データと一致する．このため，ORゲートを用いた場合，カテゴリ C_i と C_j の2つのカテゴリともに出力が'1'となり推論不能となる．一方，カウンタを用いた場合，この未知入力に一致した C_i のANDゲートは1つ（$\{*,*,1,*\}$）であるのに対し，C_j は2つのANDゲートが一致する．これより，未知入力はカテゴリ C_j に属すると推論できる．この方法により，全体ANDゲート数が非常に多くなる実応用推論問題（たとえば，10,000個以上の事例データ）では，大幅な性能向上が期待できる．

この手法は，パターン認識において高い認識精度と理論的な枠組みが明らかにされているNearest Neighbor法（k-Nearest Neighbor法における $k=1$ のケース[13]）を2値符合化された推論問題に拡張したものと見なすことができ，今後この観点から解析できるものと考えられる．

以上，本論文で提案する設計手法（LoDETT）について述べた．次節では，LoDETTを実際に適用して開発した'英単語の発音記号推論チップ'について述べる．

5. 英単語の発音記号推論チップ

5.1 英単語の発音記号推論問題（EPR）とチップの回路構成

実用的な大規模推論問題に対するLoDETTの有

効性を評価するために，英単語の発音記号推論問題（EPR: English Pronunciation Reasoning）を取り上げた（英単語は，完全な表音文字でないため，同じ文字であっても単語によって発音が異なる）．発音に関する規則を抽出する（発見する）ことが難しいため，従来のルールベースの人工知能アプローチでは処理が困難な問題である．さらに，実用には会話速度での推論が要求されるため，高速処理（リアルタイム推論）が不可欠である．

Stanfill 等は超並列計算機を利用した並列統計処理から推論を行う「記憶ベース推論」を用いて EPR 用のシステムを開発した[1]．このシステムは MBRTalk と呼ばれる．記憶ベース推論は，ヒューリスティックな手法を用いずに事例データのフラットなデータ構造だけから，データ間の重み付き距離の統計処理によって推論を行う方法であり，ルールの発見が困難な推論問題に効果を発揮する．

MBRTalk では，学習や進化といった前処理が不要であるが，1つ1つの未知単語入力（質問）ごとに未知単語と辞書（事例データベース）中の単語との間で重み付き距離の統計計算を行う必要があり，膨大な計算量が発生する．このため，超並列計算機 Connection Machine 2 を用いている[14]．MBRTalk は，超並列計算機といった大規模ハードウェアが必要であるものの，従来のルールベースの人工知能では困難であった高い推論精度を実現している．これより，後述する試作ハードウェアの性能は MBRTalk と比較する．

LoDETT を用いて設計，試作する EPR 用チップの全体構成を**図 8** に示す．質問（Query）は，パーソナルコンピュータ（PC）からインタフェース回路（I/F）を介してチップ内の各カテゴリブロック（reasoning block for each category）に並列入力（ブロードキャスト）され，AND ゲート群によって同時に一致検出が行われる．カテゴリブロックの数は 52 である．推論チップは，前節で述べたように，各カテゴリ（発音記号）ごとの AND ゲート群とカウンタ（counter），および，最大値検出回路（Max. detector）から構成される．

5.2 真理値表の構造と進化操作

英単語の発音記号推論チップの設計にあたり，T.J. Sejnowski より提供されている辞書（EPR 事例データベース）を用いた．これは，MBRTalk の開発に使用された辞書（事例データベース）であり，Webster English Pocket Dictionary をベースに 20,008 語の英単語とその発音記号が納められている（なお，文字と発音記号が 1 対 1 対応するように，通常の発音記号に

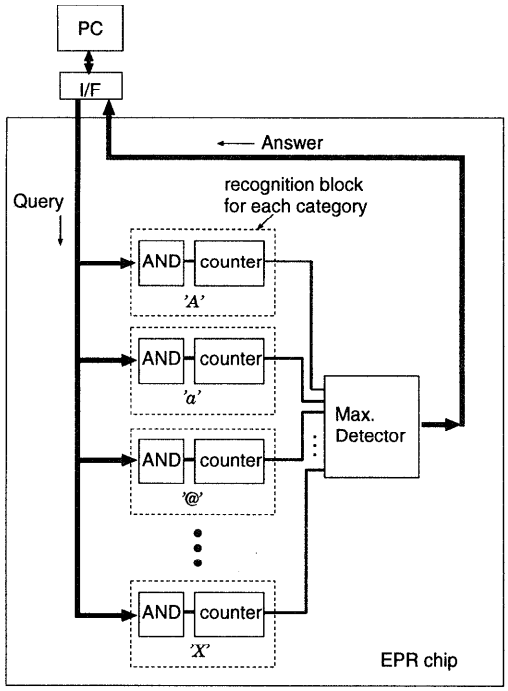


図 8 英単語発音記号推論チップの構成
Fig. 8 Block diagram of EPR chip.

拡張がなされている）．**表 2** に発音記号（Phoneme）と語例（Ex. word）を示す．発音記号の数は無音 '-' を含めて，52 種である．

MBRTalk では，辞書のデータ（単語と発音記号列の対）を基に機械的なシフト操作によって，記憶ベース推論のための事例データを作成している．これは，対象単語から 7 つの文字列をシフト操作によって取り出し，その中心文字（4 番目の文字）に対する発音記号とのセットを作成する操作である．**表 3** に，例として，'alphabet'（発音記号は，'@$lf-xbET$'）から作成される 8 個の事例データを示す．

LoDETT でも，これとまったく同様の事例データ作成し，真理値表にコーディングした．たとえば，'$- \ - \ -alph$' は 2 値化され，カテゴリ '@' の真理値表の 1 行となる．**図 9** に真理値表の構造を示す．各カテゴリとも 189 列（7 文字 × 27（26 英文字+スペース））で構成される．行数は，各カテゴリの事例データ数に等しい．

辞書から，基本真理値表用，学習用真理値表用の単語をそれぞれ最大 2,000 語までランダムに，かつ，重複しないように選択した（後述するように，最大 2,000 語まで，単語数を増やしながら推論精度と回路数を評価した）．2,000 語のときの基本真理値表用の事例データ総数は 14,796 個で，学習用真理値表の事例データ

表2 EPR辞書（事例データベース）の発音記号と語例
Table 2 Phonemes and word examples in EPR corpus.

Phoneme	Ex. word	Phoneme	Ex. word
a	odd	*D*	mother
b	bad	*E*	many
c	caught	*G*	long
d	add	*I*	busy
e	angel	*J*	jam
f	farm	*K*	anxious
g	gap	*L*	evil
h	hot	*M*	chasm
i	bee	*N*	shorten
k	keep	*O*	oil
l	lad	*Q*	quilt
m	man	*R*	after
n	and	*S*	wish
o	only	*T*	bath
p	apt	*U*	wood
r	rap	*W*	out
s	ask	*X*	mixture
t	tab	*Y*	use
u	you	*Z*	vision
v	vat	@	cab
w	we	!	pizza
x	welcome	#	exist
y	yes	*	what
z	zoo	%	up
A	eye	+	abattoir
C	chart	−	

表3 単語と発音記号列の対から生成される事例データの例
Table 3 Case data examples generated from a pair of word and its phonemes.

character sequence							phoneme
-	-	-	a	l	p	h	@
-	-	a	l	p	h	a	*l*
-	a	l	p	h	a	b	*f*
a	l	p	h	a	b	e	−
l	p	h	a	b	e	t	*x*
p	h	a	b	e	t	-	*b*
h	a	b	e	t	-	-	*E*
a	b	e	t	-	-	-	*T*

Category 'C_i'

input																													
1st character								2nd character									7th character												
a	b	c	···	x	y	z	-	a	b	c	···	x	y	z	-		a	b	c	···	x	y	z	-				
0	0	1		0	0	0	0	0	0	0		0	1	0	0		1	0	0		0	0	0	0				
0	0	0		0	1	0	0	0	0	1		0	0	0	0		0	0	0		0	0	0	1				
0	0	0		0	0	0	1	0	1	0		0	0	0	0		1	0	0		0	0	0	0				
0	0	0		0	0	0	1	0	0	0		0	0	0	1		0	0	0		0	0	1	0				
			···								···					···				···									
0	0	0		1	0	0	0	0	0	1		0	0	0	0		0	0	0		0	0	0	1				
0	0	0		0	0	1	0	0	0	0		0	0	0	0		0	0	0		0	0	1	0				
0	1	0		0	0	0	0	0	1	0		0	0	0	0		0	0	0		0	0	0	0				

図9 英単語発音記号推論チップの開発に用いた真理値表の構造
Fig. 9 Structure of the truth table used for EPR chip.

総数は14,721個であった．さらに，テスト用（推論精度評価用）の2,000単語をこれらと重複しないようにランダムに選択した．したがって，1つのカテゴリあたりの事例データ数は，約285（14,796/52）である．真理値表の1つの行は189列（フィールド）から構成

表4 GAパラメータ
Table 4 GA parameters.

個体数	20（各カテゴリごと）
交叉率	毎世代ごとに1個体ペアを選択
突然変異率	毎世代ごとに各個体の0.1%
最終世代数	10,000世代

されるので，1つのカテゴリの染色体の平均長（遺伝子座数の平均）は約53,865となる．

各カテゴリ（発音記号）ごとに初期個体群を作成し（各カテゴリごとの個体数は $N = 20$ とした），各染色体の遺伝子座の1%をランダムにオペレータ '*' とし，残りをオペレータ '1' とした．交叉操作と突然変異操作は前節で述べたとおりである．交叉は，毎世代で1対の個体をランダムに選出し，さらに交叉位置もランダムに決定した．突然変異は各世代ごとに全個体を対象とし，突然変異率は各個体の0.1%とした．設計に用いたパラメータを**表4**にまとめる．なお，個体数が20程度の集団でも世代数を増加させることにより，十分良質な個体が得られることが予備実験により示されており，これより $N = 20$ とした．

以上の遺伝的操作を各カテゴリごとに10,000世代行い，最終的に各カテゴリの20個体のうちから最優良個体を採用した．なお，各カテゴリとも10,000世代で適応度は十分飽和していた．この遺伝的アルゴリズムの処理は，パーソナルコンピュータ（Pentium 166 MHz）を6台用いて行い，すべてのカテゴリの進化終了（2,000単語）に約10時間かかった．

5.3 推論精度と回路規模の評価

進化操作終了後，FPGAへの実装前に推論精度をテスト用の単語を用いて測定した．結果を**図10**に示す．単語数は100語から2,000語まで変化させた．1,800語程度から飽和傾向が見られ，2,000単語で推論精度81.9%となった．この推論精度は，次章の性能比較で述べるようにMBRTalkの推論精度とほぼ同等である．なお，単語数の増加とともに推論精度が飽和する傾向はMBRTalkでも観測される．この共通した推論精度の飽和現象に関しても，次章の性能比較で考察する．

なお，図8のカウンタ（counter）と最大値検出回路（Max. detector）を図6に示すORゲートに変更したところ，2,000単語における推論精度は，33.4%まで低下した．この推論精度の低下は，前述した複数カテゴリのORゲートが同時に活性化されることによる推論不可能が原因である．これより，カウンタと最大値検出回路が推論精度の向上に大きく寄与していることが分かる．

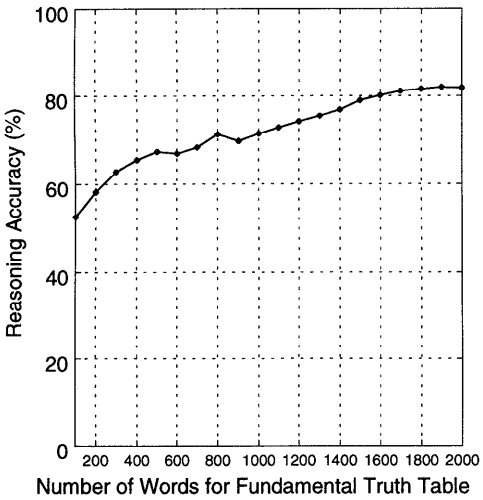


図 10 推論精度の測定結果
Fig. 10 Experimental results for reasoning accuracy.

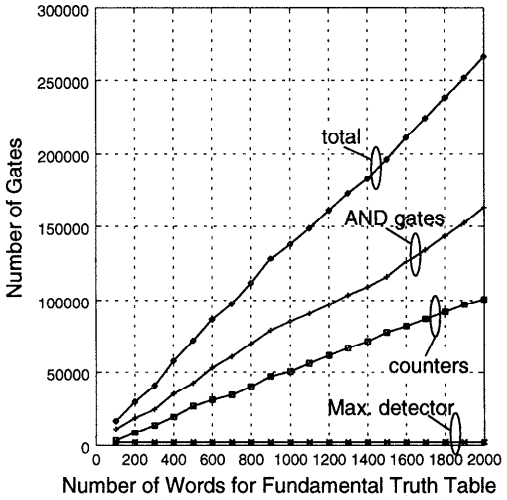


図 11 英単語発音記号推論チップの回路規模
Fig. 11 Circuit size of EPR chip.

推論精度の測定後，進化した真理値表を実際の回路に展開した．進化した真理値表の総行数は 14,796 と大きく，また，各行に含まれる don't care の数は，各行ごとに異なる．このため，実際には AND ゲートへの展開と展開した回路の圧縮を行うための CAD（論理合成 CAD）が必要となる．本試作では，(株) 図研の論理合成 CAD'Vps'☆を用いた．図 11 に単語数と論理合成した回路数の関係を示す．ここで回路数（Number of Gates）とは，2 入力 NAND ゲートの数である．図中には，進化した全真理値表の論理合成結果による AND ゲート群（AND gates）の回路数のほかに，すべてのカウンタ（counters）の総回路数と最大値検出回路（Max. detector）の回路数も示す．これらの回路の総数（total）は，図 8 の EPR チップ（EPR Chip）の回路数に相当する．AND ゲート群とカウンタの総回路数は，単語数の増加にほぼ比例して増加している．一方，最大値検出回路の規模は単語数の増加に関係なく，当然ながらほぼ一定の回路数となっていることが分かる．回路総数は 2,000 単語のときに 266,388 ゲートである．このゲート数は，実装率（最大ゲート数に対して，実際に実装できたゲート数の割合）を考慮して，現在すでに市販されている 50 万ゲートクラスの FPGA が 1 個あれば実装可能な規模である．

5.4 FPGA 上への実装結果

実装評価に使用したハードウェアの全体構成を図 12 に示す．パーソナルコンピュータ（PC）から質問（query）を入力し，推論結果（カテゴリ）をパソコンに返す．評価ボード上には，Xilinx 社の FPGA チップ XC4010-6PQ208C（1 万ゲート相当）☆☆が 12 個と，ボード上の接続配線とその再構成を行うための FPGA である FPIC（Field Programmable Interconnection Chip）が 3 個搭載されている．これより評価ボード全体は，12 万ゲート相当の 1 つの FPGA チップと同等なものとして使用できる．

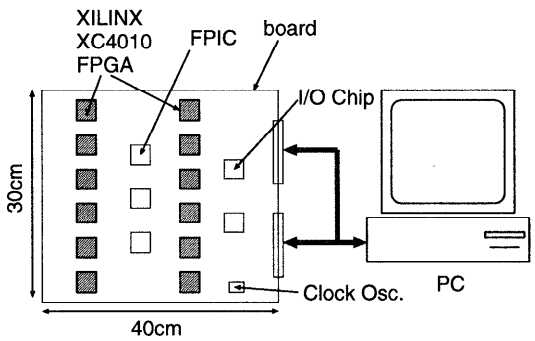


図 12 評価システムの構成
Fig. 12 Hardware configuration for EPR chip prototype.

論理合成 CAD（Vps）によって作成された論理ネットファイルを，Xilinx 社のマッピング CAD（テクノロジマッピング用のソフトウェア）である XACT に入力して配置配線用のコンフィギュレーションデータを作成し，FPGA にダウンロードした．ここで，実装評価に用いたハードウェアが上述のように 12 万ゲート相当であるため，2,000 単語で生成した回路（266,388 ゲート）を一度にすべて実装することはできない．そ

☆ http://www.zuken.co.jp/

☆☆ http://www.xilinx.com/

こで，図 8 をサブブロックに分け，サブブロックごとに実装することで 2,000 単語に対する全回路の動作確認を行った．このため，サブブロックごとにカウンタと最大値検出回路の出力をパーソナルコンピュータ（PC）にダウンロードし，パーソナルコンピュータ上で最終的な最大値検出とカテゴリ決定を行った．

質問（query）の入力から最大値検出回路までの動作速度は，評価ボード上のクロック（Clock Osc.）によって決まる．本実装評価では 2.0 MHz のクロックによって全体を動作させ，推論精度の評価を行った．この結果，図 10 と同様に，2,000 語で 81.9%の推論精度を得た．これより，クロック周波数 2.0 MHz において，実装した回路は正しく動作していることが分かる．クロック周波数 2.0 MHz は，現在のパーソナルコンピュータ等クロック速度に対してはるかに低速である．しかしながら本システムでは，事例データ数（基本真理値表の行数）と同じ数の AND ゲート群によって並列一致検出ができるので高速に推論結果を得ることができる．2,000 単語の場合，14,796 データであるので，並列度 14,796 の一致検出が同時に行われる．

なお，クロック速度 2.0 MHz は，ボード上の配線遅延と複数個の FPGA チップ内の回路遅延を含めた結果である．前述したように，現在の高集積 FPGA（50 万ゲート程度）によって図 8 全体が 1 チップ化できれば，さらなる高速推論が可能であると考えられる．

6. 性能比較

試作したハードウェアと MBRTalk の性能を比較評価した．比較では，(株) 日立超 LSI システムズ社製の超並列計算機 'MY-NEUPOWER' に MBRTalk を実装した．MY-NEUPOWER は，512 個のプロセッサ（クロック 25.0 MHz）が共有バスによって接続された SIMD（Single Instruction Multiple Data Stream）型のアーキテクチャであり，64 個の専用 LSI を中心にシステムが構成されている[15]．

前述した辞書（EPR 事例データベース）を 512 分割し，MY-NEUPOWER の各プロセッサのローカルメモリに事前にロードすることで MBRTalk を実行した．推論精度の測定結果を**図 13** に示す．推論精度の単語数依存性を測定するため，最大 5,000 語までの測定を行った．

遺伝的アルゴリズムを用いた本手法（図 10）と並列人工知能の手法を用いた MBRTalk のいずれにおいても，2,000 単語程度の事例数で推論精度が 81～83%に飽和している．これは，以下の点を示していると考えられる．

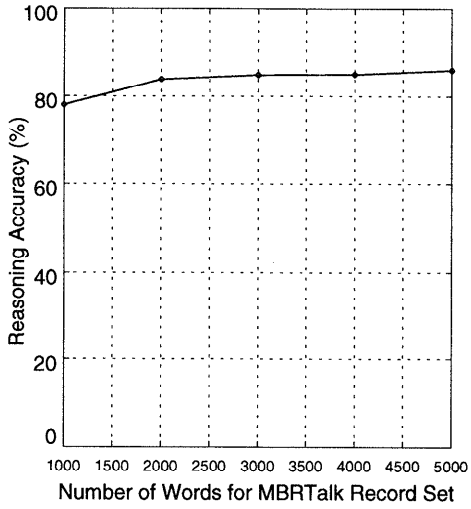


図 13 MBRTalk の推論精度の測定結果
Fig. 13 Experimental results for reasoning accuracy using MBRTalk.

(1) 単語からその発音記号を推論する規則は，2,000 単語程度の事例の中にすべて含まれており，それ以上事例を増しても新たな規則は得られない．
(2) 全単語の 19～17%は例外事例であり，規則からは推論できない（例外事例について我々人間は，推論ではなく，これらを記憶することで対処していると考えられる）．

これより，本手法と MBRTalk のいずれの場合も 2,000 語程度で推論を行うことが計算コスト，ハードウェアコスト的に最も効率が良いと考えられる．MBRTalk では，2,000 単語で推論精度 83.3%となった．

Stanfill ら[1]は MBRTalk の推論精度の評価のためにネイティブスピーカによる聞き取り試験を行っており，約 80%の推論精度であれば，違和感のない実用レベルの発音となることを示している．本手法（LoDETT）の推論精度の方がわずかに（1.4%）低いものの，ほぼ，MBRTalk なみの推論精度を実現している．これより，本ハードウェアもほぼ実用レベルの推論精度に達していると考えられる．

LoDETT による推論チップと MBRTalk の性能比較を**表 5** に示す．推論チップ（LoDETT）の推論時間 500 ns は，前述した評価ボード上のクロック周波数（2 MHz）による値である．MBRTalk の推論時間は，MY-NEUPOWER の各プロセッサ上のローカルメモリに事例データを転送後に測定した値である（データの転送速度を含んでいない MY-NEUPOWER 自体の実行時間である）．会話速度での発音記号推論を行うのには，50 ms/音素（発音記号）程度の推論速度が必要

表5 LoDETT（英単語発音記号推論チップ）とMBRTalkの比較（単語数2,000）
Table 5 Performance comparison between LoDETT and MBRTalk (2,000 words).

	LoDETT	MBRTalk
推論精度	81.9%	83.3%
推論時間（1発音記号あたり）	500 ns（評価ボードによる）	50.3 ms
ハードウェア規模	1チップ（50万ゲート）評価は12チップ（12万ゲート）の評価ボードに分割実装した.	超並列計算機 512プロセッサ
動作周波数	2.0 MHz（評価ボードによる）	25.0 MHz

であり，LoDETTとMBRTalkの両方とも要求を満たしている．なお，パーソナルコンピュータ（Pentium 200 MHz，32 MB）上でMBRTalkを実行した場合，2,000単語における推論時間は，1.2秒/音素（発音記号）であった．

以上，LoDETTを実用的な問題である英単語の発音記号推論問題に適用し，従来技術であるMBRTalkと性能を比較した．その結果，LoDETTによって実用レベルの高速推論チップを製作できる見通しを得た．

7. おわりに

FPGA等の再構成可能な集積回路を利用し，個々の事例データベースから専用推論ハードウェア（集積回路）を設計する手法（LoDETT）を提案した．本手法では，事例データベースから真理値表を作成し，この真理値表に汎化能力を持たせるためのオペレータを遺伝的アルゴリズムで決定する．遺伝的操作には，真理値表の各値と1対1対応した遺伝子座から成る可変長染色体を用いる．また，遺伝子（オペレータ）は，対応する真理値表の値を保存する遺伝子と'don't care'に変更する遺伝子の2種類を用いる．遺伝的操作によって得られたオペレータを実行した真理値表（進化した真理値表）をANDゲート群に展開することで推論のための基本回路が生成できる．しかし，このままでは例外事例データの多い実応用問題に対しては，高い推論精度が得られない．この問題点を解決するために，活性化されたANDゲート群の割合から推論を行う回路（カウンタと最大値検出回路）を用いた．

提案手法の有効性を評価するために，英単語の発音記号推論チップのプロトタイプを試作した．その結果，1チップ化が可能な約27万ゲートの回路規模のハードウェアで，推論精度81.9%，1音素あたりの推論時間500 nsを得た．従来技術であるMBRTalkを超並列計算機上に実装し，試作ハードウェアとの性能比較を行った．この結果，試作ハードウェアの推論精度はMBRTalkとほぼ同等で，MBRTalk以上の高速推論が可能であることが分かった．これより，LoDETTが，実用的な規模の問題に適用可能である見通しを得た．

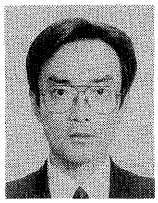
謝辞 英単語発音推論問題用のデータベースを提供いただいた，Prof. T.J. Sejnowskiに感謝いたします．なお，本研究の一部は，平成10年度文部省科学研究費補助金萌芽的研究（課題番号10875072）による．ここに，感謝いたします．

参考文献

1) Stanfill, C. and Waltz, D.: Toward Memory-Based Reasoning, *Comm. ACM*, Vol.29, No.12, pp.1213–1228 (1986).
2) Zhang, X., Waltz, D. and Mesirov, J.: *Protein Structure Prediction by Memory-based Reasoning*, Thinking Machine Corporation (1988).
3) Blelloch, G. and Rosenberg, C.R.: Network Learning on the Connection Machine, *Proc. IJCAI '89*, pp.323–326 (1989).
4) Creecy, R., Masand, B., Smith, S. and Waltz, D.: *Trading MIPS and Memory for Knowledge Engineering: Automatic Classification of Census Returns on a Massively Parallel Supercomputer*, Thinking Machine Corporation (1990).
5) Kitano, H. (Ed): *Massively Parallel Artificial Intelligence*, AAAI press/MIT Press (1994).
6) Nikola, S.B.: Simulating Artificial Neural Networks on Parallel Architectures, *IEEE COMPUTER*, Vol.29, No.3, pp.56–63 (1996).
7) Yasunaga, M., Yamada, A. and Okahashi, T.: Performance of a Bus-based Parallel Computer with Integer-Representation Processors Applied to Artificial Neural Network and Parallel AI Domains, *2nd Int. Conf. on Knowledge-Based Intelligent Electronic Systems*, Vol.3, pp.519–527 (1998).
8) Brown, S. and Rose, J.: FPGA and CPLD Architecture: A tutorial, *IEEE DESIGN and TEST of COMPUTERS*, summer (1996).
9) 末吉敏則：リコンフィギュラブルロジック，電子情報通信学会学会誌，Vol.81, No.11, pp.1100–1106 (1998).
10) Goldberg, D.E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley (1989).
11) Higuchi, T., Iwata, M. and Liu, W (Eds.): *Evolvable Systems: From Biology to Hardware*, Lecture Notes in Computer Science, Vol.1259, Springer Verlag (1996).
12) Keymeulen, D., Iwata, M., Konaka, K.,

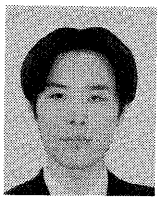
Kuniyoshi, Y. and Higuchi, T.: Evolvable Hardware: A Robot Navigation System Testbed, *New Generation Computing*, Vol.16, pp.97–122, (1998).
13) Fukunaga, K.: *Introduction to Statistical Pattern Recognition*, San Diego, Academic Press (1990).
14) Hillis, D.H.: *The Connection Machine*, MIT Press (1985).
15) Sato, Y., Shibata, K., Asai, M., Ohki, M., Sugie, M., Sakaguchi, T., Hashimoto, M. and Kuwabara, Y.: Development of a high-performance general purpose neuro-computer composed of 512 digital neurons, *Proc. IJCNN '93*, pp.1967–1970 (1993).

(平成 11 年 2 月 26 日受付)
(平成 11 年 5 月 7 日採録)



安永 守利（正会員）

昭和 56 年筑波大学第三学群基礎工学類卒業．昭和 58 年同大学院工学研究科修士課程修了．同年（株）日立製作所入社．同社中央研究所において大型計算機ハードウェア，ウェーハスケール集積回路の研究に従事．昭和 63 年より，ニューラルネットワーク，記憶ベース推論，遺伝的アルゴリズムの研究を開始．平成 3〜4 年米国カーネギーメロン大学客員研究員を兼任．平成 6 年筑波大学電子・情報工学系助教授．現在，進化ハードウェア，ニューロハードウェア，並列計算機の研究に従事．工学博士．電子情報通信学会，IEEE，INNS（国際神経回路学会）各会員．



高橋 雅聡

平成 8 年筑波大学第三学群情報学類卒業．平成 10 年同大学院理工学研究科修了．同年（株）日立超 LSI システムズ入社．大規模集積回路の設計に従事．

吉原 郁夫（正会員）

昭和 44 年東京工業大学理工学部物理学科卒業．昭和 46 年東京教育大学大学院理学研究科物理学専攻修士課程修了．同年（株）日立製作所入社，中央研究所を経て，現在システム開発研究所主任研究員．平成 9 年東北大学客員助教授，平成 11 年宮崎大学客員教授．ビル防災，環境制御，数値計算，ニューラルネットワーク，遺伝的アルゴリズムの研究に従事．工学博士．電気学会，計測自動制御学会，日本応用数理学会，日本火災学会各会員．