免疫アルゴリズムをベースとする NSP 専用エンジニアークテクチャとその評価*

吉川 雅弥[†]・寺井 秀一[†]

Immune Algorithm Processor for Nurse Scheduling Problem*

Masaya YOSHIKAWA[†] and Hidekazu TERAI[†]

Recently, novel information processing mechanisms inspired by biological immune system are proposed and showed better capability than Neural Network and Genetic Algorithm. Biological immune system consists of high information processing mechanisms such as diverse antibody production mechanisms, self-regulating mechanism and primary and secondary immune responses based on antigen's specificity and immunological memory. Immune Algorithm (IA) based on this biological immune system can obtain multi optimal solutions without the constraint. However, in order to require very much computational time, it is not suitable for the actual problems. Therefore, the dedicated hardware is important in order to apply IA to actual problems. In this paper, we proposed Immune Algorithm processor for nurse scheduling problem. The proposed architecture is realized parallel processing on the calculation for affinity degree. Furthermore, the architecture is proposed not only the pipeline at evaluation phase, but also the pipeline on the whole at reconstruction phase. Experiment result evaluating the proposed architecture was shown to achieve more than 20 times the speed keeping the scheduling quality compared with software processing.

1. はじめに

ニューラルネットワーク、遺伝的アルゴリズムなどの適応アルゴリズムは、生体の優れた情報処理機能を模倣したアルゴリズムであり、様々な工学的な応用がなされている。一方、脳神経系、遺伝適応系と並ぶ生体システム系として免疫アルゴリズム (IA) が近年、注目されている [1–6]。IA は、生物のもつ自己防御機構を工学的にモデル化したアルゴリズムであり、強力な解探索能力を有している。しかし、非常に多くの計算処理時間を要するため、実際的な問題には適さない。そのため、IA の実際的な問題への適用を考えた場合、専用ハードウェアによる IA が重要になってくる。そこで本研究では、専用ハードウェアによる高速化の見通しを得るために、ナーススケジューリング問題 (NSP) [7,8] を対象とした専用エンジニアークテクチャ (Processor of Immune Algorithm for Nurse scheduling prOblem:以下、「PIANO」とよ

ぶ) を提案する。NSP とは、医療機関において看護師にできるだけ良い労働条件を提供するために、勤務負荷のバランスを取りながら勤務シフトを決定する問題であり、互いに競合する複数の評価項目をもつ多目的性と、互いに干渉する制約条件をもつ多重制約性の二つを併せもつ問題である [8]。このような実際的な応用問題に現れる多目的性と多重制約性を併せもつ問題を対象とすることは、今後の拡張性の観点から重要である。また NSP は、前月のスケジュールを考慮して当該月のスケジュールを評価するため、IA の記憶細胞を有効に活用できる組合せ最適化問題である。

これまで IA を組合せ最適化問題に適用した研究報告例としては、関数最適化 [1] や巡回セールスマン問題 [2] があるが、いずれもソフトウェア処理である。本論文で展開する専用ハードウェアによる処理の高速化の発表は、これまで見当たらない。

2. 免疫システム

生物は、有害な物質に対する自己防御機構をもっており、それを免疫システムという。この免疫システムは、多様性のある抗体の産生およびその自己調節機構、抗体の特異性や免疫学的記憶に基づいた免疫応答など様々な

* 原稿受付 2005 年 10 月 21 日

[†] 立命館大学 電子情報デザイン学科 Department of VLSI System Design, Ritsumeikan University; 1-1-1, Nojihi-gashi, Kusatsu city, Shiga 525-8577, JAPAN

Key Words: immune algorithm, dedicated hardware, pipeline, nurse scheduling problem, FPGA.

情報処理機構によって構成されている[3]。免疫システムでは、病原菌やウイルスなどの抗原が体内に侵入してくると、それらを排除するのに有効な抗体を選択的に産生する。しかし、抗原を排除し、抗体が体内で異常に増えすぎた場合には、その抑制を行う。

ここで免疫システムを基本とする免疫アルゴリズム(Immune Algorithm:IA) [1]は、多点探索による大域探索を基本動作とし、記憶細胞による有効解の記憶、サプレッサー細胞による記憶解の再探索の抑制を行う。組合せ最適化問題にIAを適用することを考えた場合、侵入した抗原が与えられた問題であり、抗原を排除するために再生する抗体が問題に対する最適解である[2,3]。Fig.1にIAのフローチャートを示す。

IAでは、まず抗原の認識(Step1)と初期抗体群の生成を行う(Step2)。ここで、2回目以降の探索の場合には、記憶細胞から過去に有効であった抗体群を生成する。つぎに、抑制処理を行うかどうかのチェック(Step4)を行う。抑制処理では抗体どうしの親和度(二つの抗体に共通する遺伝子の割合)を計算し(Step4.1)、抗体産生の促進と抑制を行う(Step4.2)。具体的には、次世代に残る抗体の期待値を計算し、抗体のいくつかを消滅させる。ここでは、Step3~Step5の繰り返し1回を1世代とよぶことにする。さらに、Step4.2で消滅させた抗体に代わる新しい抗体を生成し(Step4.3)、抗原を排除できる抗体を構築するために抗体の再構築を行う(Step5)。以上の処理を終了条件が満たすまで繰り返し(Step3)、最終的に得られた抗体を記憶細胞に記憶(Step3.1)させ処理を終了する。

Fig. 1 Immune Algorithm

Fig. 2 Example of coding

3. IAのハードウェア化

3.1 コーディング

NSPは、看護師*i*の*j*日の勤務 X_{ij} を要素とする行列を決定する問題である。 X_{ij} は、基本的な勤務パターンの日勤・準夜勤・深夜勤・休日のいずれかである。これらを日勤00、準夜勤01、深夜勤10、休日11とし抗体のコーディングを行う。Fig.2に本研究で用いたコーディングを示す。PIANOでは、実用的なシステムの構築を目的に看護師数を可変とし、最小看護師数を16人、最大看護師数を48人とした。また、対応日数は31日とし、32日目の列には各抗体のIDを格納した。抗体数は1024抗体まで扱うことが可能であり、各抗体の評価値は、最大1024抗体の評価値の分布を十分に反映させるために2Byteとした。

3.2 制約条件

本研究で採用した制約条件を以下に示す。

- (1) 婦長・主任は夜勤をしない。
- (2) 夜勤者はあるレベル以上
- (3) 日勤者 $\geq$ 必要日勤者
- (4) 深夜勤者=必要深夜勤者
- (5) 準夜勤者=必要準夜勤者

- (6) 基本夜勤パターンを守る。
- (7) 連続夜勤日数を守る。
- (8) 夜勤パターンと夜勤パターンの間は少なくとも1週間あける。
- (9) 日勤-深夜勤の組合せは禁止
- (10) 準夜勤-日勤の組合せは禁止
- (11) 準夜勤-深夜勤の組合せは禁止
- (12) 休日の前日に準夜勤は禁止
- (13) 休日の翌日に深夜勤は禁止
- (14) 公休日数=指定公休日数
- (15) 夜勤日数の均一化

ここで、(1)と(2)は人に関する制約であり、(3)~(5)は1日に関する制約であり、(6)~(13)は勤務状態の前後関係に関する制約であり、(14)と(15)は各看護師の1ヶ月分のスケジュールに関する制約である。本研究では、(1)~(5)をスケジュールリングの絶対条件とした。すべての制約条件を評価関数に取り込むと、絶対条件を満たさない解(スケジュール)が大多数を占める。そこで、制約条件の取り扱いとして、以下の二つの手法を導入する。

- (i) 制約条件を崩さないように抗体の再構成に工夫を加える。
- (ii) 制約条件をペナルティとして評価関数に取り込む。

具体的に PIANO では、絶対条件 ((1)~(5)) に対しては (i) の手法を採用し、その他の制約条件に対しては (ii) の手法を採用する。ここで Fig. 2 のコーディングにおいて、1ヶ月 (31 日) のスケジュールを日ごとで見た縦方向の制約と、看護師ごとに見た横方向の制約に分ける。具体的には、(1)~(5) が縦方向の制約、(6)~(15) が横方向の制約になる。縦方向の絶対条件に対しては、あらかじめ制約条件を満たした抗体を発生させ、抗体の再構成においても縦方向の制約を満たす処理を行うことで、絶対条件を充足させるようにする。一方、横方向の制約に対しては、制約違反があるごとに評価値に反映させるペナルティ方式を採用した。したがって、優良抗体ほど評価値 (ペナルティ) が低い。具体的な評価方法は (6)~(15) の制約条件の違反数をペナルティとし、その線形和を評価とする。

3.3 抗体の再構成

免疫系は、遺伝子の再構成によって多様な抗体を産生し、生体内に侵入する抗原に対応する。これは、遺伝的アルゴリズム [9,10] と同じように免疫系が抗体遺伝子の再構成による多点探索処理を行うことを示している。そこで、本研究での抗体の再構成は、選択・交叉・突然変異の三つの処理で構成する。

3.3.1 選択オペレーション

選択オペレーションとしてはトーナメント戦略とエリート保存戦略を実現することとした。さらに、世代モデルとしては連続世代モデルと離散世代モデルの二つを可能とした。離散世代モデルとは Generation Gap (現世代の何割の抗体が入れ替わるかの割合) が 1 の場合、すなわち次世代の抗体をすべて交叉・突然変異によって生成する世代モデルであり、それ以外は、連続世代モデルである。ここで、トーナメント戦略は、抗体群からランダムに任意の数 (あらかじめトーナメントサイズとしてパラメータで与える) だけ抗体を選択し、その中で最も評価の良い (評価値はペナルティ方式なので低い) 抗体を選択することによって実現する。またエリート保存戦略は、抗体群の中で最も評価の良い抗体を次世代に残すことで実現する。

3.3.2 交叉・突然変異オペレーション

交叉についてはマスクを制御することによって、1 点交叉、複数点交叉、一様交叉をそれぞれ実現する。具体的に 1 点交叉を例に Fig. 3 を用いて説明する。Fig. 3 の場合、交叉位置が 3 ビット目と 4 ビット目の間にある。このとき交叉位置の前、すなわち 1~3 ビット目までマスク '1' を、4 ビット目以降にマスク '0' をそれぞれ割り当てる。ここで、マスクが '0' の場合は直通、'1' の場合は交叉を行う回路を用いる。一様交叉は、この 0・1 のマスクの割当てをランダムに行うことによって実現する。

また突然変異は、点突然変異とペア交換突然変異をそれぞれ実現する。ここで、点突然変異とは一つの抗体の中

Fig. 3 Example of one-point crossover

で、任意の格子 (Fig. 2 における行列の要素) に対して、ランダムに対立遺伝子へ置き換える突然変異のことである。ただし、突然変異によって制約違反が起きる場合には、突然変異は行わない。また、ペア交換突然変異は、任意の二つの格子に対し、入れ替えを行う突然変異のことである。ただし、点突然変異と同様に、突然変異によって制約違反が起きる場合には、突然変異は行わない。

3.4 自己調節機構

自己調節機構は、親和度計算、期待値計算、抗体産生の促進と抑制を行うことで実現する。まず、親和度計算では、抗体群の抗体同士の親和度 (類似度) を考え、抗体群の中に自分と類似した抗体がどの程度存在するかを計算する。抗体同士の親和度は、それぞれの抗体で対応する格子を順に比較し、一致する格子の数があらかじめ設定した閾値を超えた場合に類似していると判断する。ここでは、類似している 2 抗体のうち、評価の悪い抗体の親和度に 1 を加えることとした。つぎに、求めた親和度を用いて全抗体に占める各抗体の濃度 C_n を以下のように求める。

$$C_n = \frac{A_n}{W} \quad (1)$$

ただし、 A_n は抗体 n の親和度、 W は全抗体数を表している。この濃度を用いて期待値 E_n を求める。

$$E_n = \frac{1}{C_n \times F_n} \quad (2)$$

ただし、 F_n は抗体 n の評価値を表している。この評価値はペナルティ方式を採用しているため、優良抗体ほど評価値 (ペナルティ) が低い。ここではこの期待値があらかじめ定めた閾値以下の抗体を淘汰する。

3.5 抗体産生機構

抗体群として抗体の数を保つため、期待値計算によって淘汰した抗体に代わる新たな抗体をランダムに産生する。ただし、この場合も (1)~(5) の絶対条件を満たす抗体を発生させる。

4. インプリメンテーション

4.1 回路構成

PIANO の回路ブロック図を Fig. 4 に示す。Fig. 4 でのデータフローを Fig. 1 のフローチャートに沿って説明

れぞれ行い、選択したアドレスをデータバススイッチへ送る。連続世代処理では、乱数発生器で全抗体数より小さい乱数を発生させ、その乱数の値をアドレスとする。データバススイッチからのリクエスト信号が立ち上がると抗体の ID をデータバススイッチへ送信する。また同じ抗体のアドレスをデータバススイッチへ送信しないように制御する。エリート保存戦略では、連続世代処理で選択していない抗体の中から、最も評価の良い抗体のアドレスを選択する。そして、データバススイッチからのリクエスト信号が立ち上がると、選択した抗体の ID をデータバススイッチへ送信する。トーナメント戦略では、世代管理回路から送られてくるトーナメントサイズの数だけ、全抗体数より小さい乱数（アドレス）を発生させる。そして、その中で最も評価の良い抗体のアドレスを選択する。データバススイッチからのリクエスト信号が立ち上がると、抗体の ID をデータバススイッチへ送信する。また、連続で同じアドレスをデータバススイッチに送らないように制御する。

再構成モードから抑制モードには、あらかじめ設定した世代に 1 回だけ切り替える。抑制モードでは、最初の抗体アドレス (0) から最後の抗体アドレス (全抗体数) までを、データバススイッチへ送信する。最後の ID (全抗体数) の送信後に、再び再構成モードに切り替える。以上の動作を抗体の評価値が評価値基準を満たすまで繰り返す。

4.2.2 交叉回路

交叉回路では、縦の制約を崩さないように抗体の縦列（日にち）について、1 点交叉、複数点交叉、および一様交叉を行う。Fig. 6 にブロック図を示す。前処理として、世代管理回路からパラメータ（交叉率や交叉手法の情報）を受け取る。データバススイッチから入ってくる一つ目の抗体データ A はすべて MUX-1 を使って、まず RAM へ格納する。つぎに二つ目の抗体データ B は、レジスタ 2 から MUX-2 を通過する時に、制御回路からの

Fig. 6 Crossover circuit

交叉マスクによって、RAM に格納した抗体 A と交叉を行い、突然変異回路へ交叉済み抗体 A' を出力する。ここで、もう一方の交叉済み抗体 B' は、MUX-1 を使って RAM に格納する。つぎに三つ目の抗体 C が入ってくると、RAM に格納した交叉済み抗体 B' を MUX-2 から突然変異回路へ出力し、抗体 C を RAM へ格納する。そして、四つ目の抗体 D と抗体 C を同様に交叉をさせ、一方を突然変異回路へ出力、もう一方を RAM へ格納する。この処理を繰り返す。

4.2.3 突然変異回路

突然変異回路は、交叉回路同様に、縦の制約を崩さないように抗体の縦列（日にち）について、ペア交換や点突然変異を行う。Fig. 7 にブロック図を示す。

世代管理回路から必要日勤者数、看護師のレベル、看護師の人数、突然変異率を受け取る。乱数を発生させ、突然変異箇所（格子位置）と突然変異方法を決定する。これらは、抗体を受け取るたびに決定する。まず、イネーブル信号と突然変異率によって突然変異を行うか行わないかを判断する。イネーブル信号の立ち上がりで交叉回路より 1 抗体を 12 ビットずつ受け取る。この時、乱数で指定した突然変異箇所が、条件を満たすかどうか判定する。すなわち、突然変異を行った場合に絶対条件を満たすかどうかを判定する。突然変異箇所が条件に合わない場合、乱数に 1 を足し、次の格子の条件を判定する。条件を満たす格子が見つかるまでこの処理を繰り返す。そして、データを出力する時に突然変異箇所の変異を行い、12 ビットずつ出力する。ただし、日勤を休日に変異させる点突然変異は、その日の日勤者数をカウントして、必要日勤者数を超えた時のみ突然変異を行うように制御する。

Fig. 7 Mutation circuit

4.2.4 評価回路

評価回路では、各抗体の評価値を計算し、結果を選択回路へ送るとともに、評価値RAMへ格納する。この評価値RAMに格納したデータは、期待値計算で使用される。Fig. 8にブロック図を示す。まず、抗体変換回路と格子カウント回路に抗体データを送る。抗体変換回路では先月の最後の4日分の抗体データと今月の抗体データの連結を行う。これにより、月の終わりと初めにおいても制約条件を守ることができる。また格子カウント回路では看護師ごとに休日と夜勤の数をカウントする。そして各評価値計算回路にその結果を送る。評価値計算回路では、各評価の評価値を計算する。評価値計算回路ではそれぞれ処理時間が異なるため終了した回路から評価値合計格納回路に結果を送る。

Fig. 8 Evaluation circuit

4.2.5 親和度計算回路

親和度計算回路は、抑制処理に必要な抗体同士の親和度を計算し、そのデータを期待値計算回路へ送る。ここで抗体同士が類似していると判断した場合は、評価の悪い抗体に対して親和度を1加算する。計算した各抗体の親和度は、 10×1024 ビットRAMに格納する。PIANOでは回路規模と処理速度の観点から、抗体を格納するための 24×128 ビットRAMを70個使用することとした。これにより、同時に複数の抗体間の親和度を計算することで、高速化を実現する。Fig. 9にブロック図を示す。

データバススイッチからのイネーブル信号によって回路の動作を開始する。各抗体は24ビットずつ送られてくる。抗体比較回路内では、各抗体のRAMへの格納(RAMは70個使用)、そしてその抗体とデータバススイッチから送られる各抗体の格子ごと(人数×月の日数)の比較を行う。評価値比較回路では評価値の比較を行う。親和度合計・格納回路では、抗体比較回路と評価値比較回路からそれぞれ結果を受け取り、データバススイッチから受け取った抗体の評価値が、RAMに格納した抗体の評価値より低い場合のみ親和度に1を加える。抗体70個分の親和度の計算後に、その結果をRAMに格納する。

Fig. 9 Affinity circuit

以上の処理を各世代の全抗体に対して行う。

4.2.6 期待値計算回路

期待値計算回路は、親和度、評価値、抗体数から期待値を計算する回路である。期待値が閾値以上の抗体には1、閾値以下の抗体には0を出力する。Fig. 10にブロック図を示す。

期待値計算回路では、前処理として世代管理回路から抗体数と閾値を取り込む。ALUでは親和度と評価値の積算を行い、その結果を抗体数に応じてシフトする。この値が期待値となる。つぎに制御回路(Controller 2)によって期待値と閾値の比較を行い、その結果を結果RAMに送る。

Fig. 10 Expectation circuit

4.2.7 新抗体生成回路

新抗体産生回路は、抑制処理で淘汰した抗体を新たに作り出すことで、抗体群の抗体の数を保つ働きをする。抗体産生は、縦方向(1日ごと)の制約を満たす抗体産生を行う。具体的には、1日あたりの深夜勤者、準夜勤者は夜勤が可能なレベルの看護師の中から決定し、その日の日勤者は、全看護師の中で、休日者、深夜勤者、準夜勤者の人数を引いたの中から決定する。余った人数は日勤と休日にランダムに振り分ける。Fig. 11にブロック図を示す。

Fig. 11 Production circuit

新抗体生成回路は、世代管理回路からシステムスタート信号を受け取ると、パラメータ（看護師の人数、希望休日者、希望休日者の人数、必要日勤者数、深夜勤者数、準夜勤者数、夜勤可能者数、夜勤可能者の希望休日者数）をもとに抗体を生成する。まず、看護師に0から最大47までのIDをつける。パラメータはレジスタとメモリに格納し、システムスタート信号を入力すると、8日分ずつ4回に分けて1抗体の生成を行う。このとき、1日ごとに希望休日者、深夜勤者、準夜勤者、必要日勤者を決定し、残った人を日勤と休日にランダムに振り分ける。生成方法は、希望休日者をIDの大きい順に一人ずつ入力する。つぎに、深夜勤者を夜勤ができるレベルの人（希望休日が入っている人は除く）の中から必要人数だけランダムに決定する。たとえば、夜勤ができるレベルの人が20人の場合、19以下の乱数を発生させ、その値のIDの人を深夜勤者とする。準夜勤者も深夜勤者と同様に決定する。PIANOでは、乱数発生器としてLFSR（Linear Feedback Shift Register）を用いる。つぎに、日勤者を残りの人の中から必要人数だけランダムに決定する。最後に、残った人を日勤と休日にランダムに振り分ける。生成した抗体は、8個のRAM（2ビット×512ワード）に格納する。ここでは、RAM1からRAM8に8日間ずつ格納するため2抗体まで格納でき、1抗体目の生成が終了したら、2抗体目を生成する。その後は、抗体出力信号を入力し、データの出力が終了したら次の抗体の生成を開始する。

5. 多重パイプライン

5.1 メモリアクセスに着目したパイプライン

各モジュール間のデータ転送用バス幅は、メモリのバス幅（24ビット）と、回路規模を考慮し、12ビットを基準とした。つまり、メモリから読み出した24ビットデータを、12ビットごとに分割して交叉回路へと出力する。

メモリからの24ビットデータ読み出しは、2クロックに1回の割合で行うが、交叉回路へは毎クロック12ビットずつ転送するので、ここでは毎クロック12ビットのデータ読み出しを行っていると考えられる。また、書き込みについても同様の動作を行う。Fig. 12に、逐次処理を行った場合のデータフローを示す。

これは看護師数が32人の場合を表した図であるので、1抗体の読み出しと書き込みには171クロック（32(看護師数)×64(看護師1人あたりのデータ)/12(データ転送用バス幅))要する。しかし、メモリアクセスに着目すると、アクセスを行っていない時間が多く存在し、空き時間が多いことがわかる。そこでメモリアクセスに着目し、再構成したパイプラインをFig. 13に示す。

Fig. 13では、1抗体をメモリから読み出すのに要するクロック数を1フェーズ(t)とよぶことにする。PIANOでは、抗体データを突然変異回路から、データパススイッチと評価回路へ分岐して処理する。そのため、評価回路が評価値計算時間を待つことなく、抗体データを次世代メモリ(T+1世代)に書き込むことが可能である。これは、コーディングで各抗体が内部に評価値をもたないからである。逆に評価値を内部にもつ場合には、評価値計算が終了し、評価値部分の書き換えが終了しないと、メモリへの書き込むことはできない。

PIANOではFig. 13に示すように、読み出し、交叉・突然変異、書き込みのフェーズが等しくなり、メモリアクセスに空き時間のないパイプラインを実現した。

R:Read C:Crossover M:Mutation E:Evaluation W:Write

Fig. 12 Sequential processing

R:Read C:Crossover M:Mutation E:Evaluation W:Write

Fig. 13 Pipeline of memory access

5.2 世代間パイプライン

ここでは、二つの世代にまたがったパイプラインを考える。T世代の抗体群からT+1世代の抗体群へ変化する時のデータフローをFig. 14に示す。これは、看護師数が32人、抗体数が512の場合を示している。抗体の

Fig. 14 Pipeline with whole transaction

選択には評価値が必要なため、抗体の選択にはすべての評価が終わり、それらすべての抗体のソーティング処理が終了する必要がある。

そのため、 $T+1$ 世代メモリからの抗体選択を開始できるのは、ソートが終了したフェーズ $t+8$ からである。ソートにかかるクロック数は、抗体数によるが、512 抗体の場合、522 クロック必要である。メモリアクセスに着目すると、フェーズ $t+3$ からストールが生じる。Fig. 14 では省略しているが、実際にはフェーズ $t+8$ の途中までの 5 フェーズのストールが生じる。そこで、スループットを向上させるために、ソート結果を必要としない処理、すなわち連続世代処理において、次世代に残る抗体をランダムに選び出す処理を行う。これにより、フェーズ $t+3$ から抗体の選択と読み出しができ、メモリアクセスの空き時間をなくすることが可能になる。このように提案パイプラインはメモリへのアクセスを基本にしているため、NSP の制約に追加や変更があった場合でも適用することが可能である。

6. 評価実験

Fig. 4 に示した回路ブロックを HDL で記述し、論理検証、論理合成（動作周波数 30MHz）、および FPGA への実装を行った。論理検証ツールは Cadence 社の NC-Verilog を使用した。また、FPGA は Xilinx 社の Virtex2XC2V6000 を使用した。Fig. 15 に実際に PIANO を実装した FPGA ボードシステムを示す。また、各回路のゲートサイズを Table 1 に示す。ここで Table 1 の ESBs はメモリブロックのことである。

本論文で提案した PIANO の性能を評価するために、ま

Fig. 15 FPGA board system

ず、ソフトウェアとの処理速度の比較実験を行った。比較実験では、看護師数を 32 人、世代数を 10000 とし、抑制間隔と抗体数を変えて実験を行った。ソフトウェアについては C 言語でプログラミングし、PC 上（CPU: Pentium IV 2.4[GHz], OS: Vine Linux, メモリ: 1[GHz]）で実行した。Table 2 に、処理時間およびソフトウェアに対する速度比を示す。抗体数が多く、抑制間隔が短い場合、親和度計算量が増えるので、ソフトウェアでは処理時間が大幅に増加した。一方、PIANO では、抗体数や抑制処理間隔の増減に関係なく、平均で 19.3 倍の高速化を実現し、PIANO の有効性を確認した。また、今回の評価実験では FPGA の制約上 30[MHz] で実装したが、PIANO を LSI 化することによってさらに高速化を図ることが可能である。

つぎに、解の質を評価するために GA との比較実験を行った。IA と GA 共に、世代数を 4000、交叉率を 0.7、突然変異率を 0.01 とした。Table 3 に評価値の比較結果を示す。Table 3 は、10 回の実験の結果であり、括弧内の数字は抑制間隔を示している。すべての実験において、抑制間隔にかかわらず、GA よりも優良解を獲得することができた。

Table 1 Gate size

Block name	Logic resource	ESBs resource
Crossover	263/69920	1/144
Mutation	1326/69920	2/144
Evaluation	1602/69920	2/144
Selection	894/69920	3/144
Datapass	1108/69920	1/144
Production	9664/69920	10/144
Affinity	8597/69920	72/144
Expectation	205/69920	1/144
Memory	26/69920	48/144

Table 3 Comparison with genetic algorithm

ex#	GA	IA(50)	IA(100)	IA(200)
1	6973	5136	5291	5206
2	7657	5203	5219	5322
3	5844	5053	5193	5264
4	6690	5045	5424	5508
5	7928	5196	5235	5267
6	6955	5133	5183	5199
7	7419	5100	5174	5444
8	7406	5101	5181	5213
9	7704	4993	5296	5624
10	6554	5048	5340	5437
av.	7113	5101	5254	5348

Table 2 Comparison PIANO with software processing

The interval of suppression(generation)	The number of antibodies	HW(30MHz) [s]	SW [s]	Speed ratio
50	128	6.90	111.19	16.11
50	256	13.60	256.34	18.85
50	512	28.67	692.36	24.15
50	1024	60.22	2061.69	34.24
100	128	6.82	103.29	15.15
100	256	13.69	215.69	15.76
100	512	28.33	523.02	18.46
100	1024	57.75	1393.95	24.14
200	128	6.71	99.10	14.77
200	256	13.56	204.85	15.11
200	512	27.79	446.65	16.07
200	1024	55.72	1060.86	19.04

Fig. 16 Transition of evaluation value

さらに、IA の HW と SW の評価値の推移を Fig. 16 に示す。Fig. 16 より、HW 処理での演算精度に問題がないことが確認できる。最後に、抑制間隔が 50 世代の場合のスケジューリング結果を Fig. 17 に示す。

7. おわりに

本論文では、免疫アルゴリズムをベースとする NSP 専用エンジンアーキテクチャを提案した。提案アーキテクチャでは、親和度計算の並列化や再構成処理のパイプラインだけでなく、メモリアクセスに着目したパイプライン処理を実現した。また FPGA に実装した評価実験では、ソフトウェア処理に比べて、平均で 19.3 倍の高速化を実現した。これにより、様々な実的な応用問題に IA を適用することの見通しを得た。今後は、NSP 以外の組合せ最適化問題に適用ができるような汎用性を高めるアーキテクチャの拡張を行う予定である。

Fig. 17 Scheduling result

謝 辞

本研究の一部は文部科学省科学研究費補助金若手研究 (B) (課題番号:16700153) による助成のもとで行われたものである。ここに謝意を表す。

参 考 文 献

- [1] 森, 築山, 福田: 免疫アルゴリズムによる多峰性関数最適化; *T.IEE Japan*, Vol. 117-C, pp. 593-598 (1997)
- [2] N. Toma, S. Endo, K. Yamada and H. Miyagi: The immune distributed competitive problem solver with MHC and immune network; *Intelligent Engineering Systems through Artificial Neural Networks*, Vol. 10, pp. 317-322 (2000)
- [3] N. Toma: A study on immune optimizations for multimodal functions and division-of-labor problems; Doctoral thesis, Doctoral Course in Interdisciplinary Intelligent Systems Engineering, Graduate School of Science and Engineering, University of the Ryukyus, pp. 317-322 (2003)
- [4] W. D. Sun, Z. Tang, H. Tamura and M. Ishii: An artificial immune system architecture and its applications; *IEICE Trans. Fundamentals*, Vol. E86-A, No. 7, pp. 1858-1868 (2003)

- [5] WD. Dasgupta, ed.: *Artificial Immune Systems and their Applications*, Springer Verlag (1999)
 - [6] Y. Ishida and Y. Tokimasa: Diagnosis by a dynamic network inspired by immune network; *Proc. World Congress on Neural Networks*, pp. 508–511 (1996)
 - [7] J. L. Arthur and A. Ravindran: A multiple objective nurse scheduling model; *AILE Transactions*, Vol. 13, No. 1, pp. 55–60 (1981)
 - [8] 山村, 小林, 山岸, 阿瀬: 遺伝的アルゴリズムによるナーススケジューリング; 遺伝的アルゴリズム 2 (北野編), pp. 89–125 (1995)
 - [9] J. Holland: *Adaptation in Natural Artificial Systems*, the University of Michigan Press (Second edition ; MIT Press) (1992)
 - [10] D. E. Goldberg: *Genetic algorithms in search optimization, and machine learning*, Addison Wesley (1989)
-