定常状態ＧＡに基づくパイプライン型 GA-VLSI の設計と評価

4AG－9

安岡 智宏　　森木 俊臣　　吉田 紀彦
九州大学大学院システム情報科学研究科

1. はじめに

複雑な問題を解くことに用いられる解法として遺伝的アルゴリズム（ＧＡ）がある．このアルゴリズムは通常ソフトウェア上で作成され，その処理時間をなんとか短縮しようと並列化や分散化などが行なわれてきている．そこで本研究ではＧＡの決定的な高速化のためにＧＡ専用のハードウェア回路 GA-VLSI を製作することを目指した．これにはＧＡＰ (Genetic Algorithm Processor) と名付けている．設計にはハードウェア記述言語を用いた．

2. ハードウェア記述言語

ハードウェア記述言語とは抽象的手続きによりハードウェアを導出するものであり，ソフトウェアでの高級言語に対応する．ハードウェア記述言語によるハードウェア開発を高級言語によるソフトウェア開発と照らし合わせると，論理シミュレーションはインタプリタによる解釈実行に相当し，論理合成によるネットリスト導出はコンパイラによる実行モジュール生成に対応する．今回設計に用いたハードウェア記述言語は SFL[2] である．

3. ＧＡＰの全体構成

ＧＡＰの全体構成図を以下に示す．

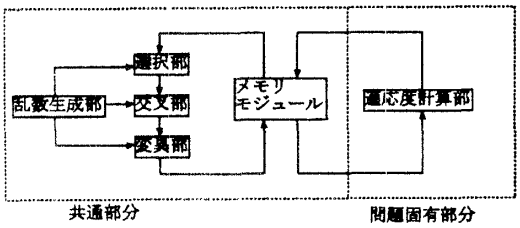

図１．全体構成図

選択部ではメモリを走査し乱数により個体を選択する．選択した個体は交叉／変異部におくられ遺伝的操作が行なわれる．乱数生成部は毎クロック乱数を生成し続け，また適応度計算部では与えられた個体の適応度を計算して返す．適応度計算部のみ問題に固有のものであり，逆にいえばこの部分を変えるだけでさまざまな問題に適応できる．

Design and Evaluation of Pipeline GA-VLSI Based on Steady-State GA
Tomohiro Yasuoka, Toshiomi Moriki, Norihiko Yoshida, Kyushu University, Graduate School of Information Science and Electrical Engineering

4. ＧＡＰの構成モジュール

ＧＡＰを構成するモジュールについて説明する．

4.1 乱数生成部

疑似乱数の生成は Xellar automata による方法[4] を用いた．この方法は，ハードウェア上での疑似乱数生成の方法の中で最も良質な乱数を作る方法の１つである．この乱数生成の方法では，Rule 90 と Rule 150 という二つのルールを用いる．これらはそれぞれ

$$\text{Rule90: } S_i^+ = S_{i-1} \oplus S_{i+1}$$
$$\text{Rule150: } S_i^+ = S_{i-1} \oplus S_i \oplus S_{i+1}$$

と定義される．S_i はビット列の i 番目のビットを表し，S_i^+ は S_i の次のフェーズの状態を表している．$\oplus$ は EOR（排他的論理和）である．これを現在の乱数値に交互に適応してやることにより次状態の乱数が生成されるというものである．適応の順序は以下の通りで，ビット列からはみ出したものは０とみなす．MSB と LSB の部分は例外的に必ず 150 をとる．

150,150,90,150,90, ... (交互) 150,90,150

4.2 交叉変異部

遺伝的操作である交叉，変異を行なう．処理の内容はまず乱数値と閾値の判定で閾値の値を越えた場合操作を行なう．そのときは別入力の乱数をデコードしてどの位置に対して操作を行なうかを得て，その位置に対してビットの反転やビット列の入れかえをおこなう．

4.3 選択部

ここではルーレット方式親選択法を用いた．このアルゴリズムは以下のような手順を取る．

１） 集団の全ての個体の適応度を合計する．
２） ０から適応度の合計までの間の乱数を生成する．
３） 集団の先行する個体の適応度に個体の適応度を加えてゆき，以上の値に達した時，その時の個体を親として返す．

4.4 適応度計算部

適応度の計算を行なう．この部分は解くべき問題に応じて作成することとなる．個体を入力として取り，その適応度を出力するというインターフェースで作成する．

5. 定常状態ＧＡ

今回，ハードウェア化を従来のＧＡだけでなく，定常状態ＧＡ (Steady State GA) についても行なった．
定常状態ＧＡとは世代という概念の無いＧＡであり，交

叉変異を選ばれた個体に対して行なう所までは共通であるが，その個体を現在の個体群の中で適応度が最悪のものと取り替えるということを行なう．
この二つの違いは操作後の個体をメモリにすぐに書き込むという所で，従来型ＧＡはメモリが２つ必要なのに対し，１つですむため，ハードウェア化に向いている．具体的には，集団内の最悪の適応度を持つ個体の所に新しい個体を書き込むということを行なっている．
この最悪の個体を特定する方法は，選択部で集団を走査する際に同時にチェックを行ない，選択し終った範囲までででの最悪のものを選ぶようにした．

6. パイプライン化

従来型ＧＡの並列処理化の行なえる部分は選択部と交叉／変異部である．選択終了時に交叉／変異部に選択された個体を渡し，すぐに次の選択を開始するようにした．

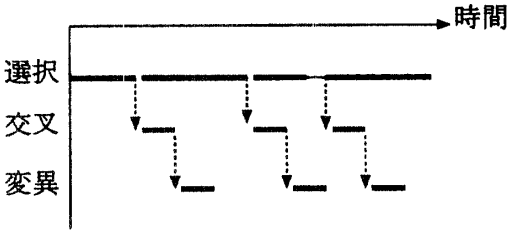

図２．従来型ＧＡの並列処理

定常状態ＧＡの方は，選択，交叉／変異と適応度計算の３つについてパイプライン的に並列処理を行なった．上と同様に選択終了後，交叉／変異部に個体を渡して次の選択を開始する．交叉／変異が終了すると適応度計算を行なう．今回，メモリが１つしかないため選択が行なわれている間はメモリ２重アクセスとなるため書き込みが出来ない．そこで選択終了時に前回の操作後の個体を書き込んでいる．

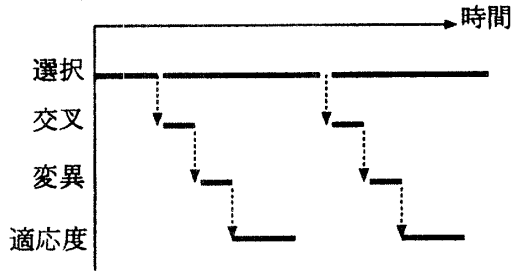

図３．定常状態型ＧＡの並列処理

7. 回路の規模

設計したＧＡの規模は１世代における個体数は６４であり，個体のビット幅は３２ bit である．また適応度計算部は現在は動作確認用としてもっとも簡単な例題の１つである De Jong の評価関数１について作成した．それを以下に示す．

$$f(x_1,x_2,x_3)=\sum_{i=1}^{3}x_i^2 \quad (-5.12\le x_i\le 5.12)$$

これらについての回路のゲート数は以下のようになった．

交叉部	変異部	適応度計算	乱数生成
590	391	7335	599

並列処理の有無	従来型ＧＡ	定常状態ＧＡ
並列化前	23602	22590
並列化後	24464	24114

8. シミュレーションによる評価

ハードウェア記述言語 SFL のシミュレーションによって評価を行なった．シミュレーション上でのクロック数を計測することにより，以下の結果を得た．
１）従来型ＧＡの１世代辺り平均クロック数
並列化なし：２０７７
並列化あり：１９３６
２）定常状態型ＧＡの１個体辺りの平均クロック数
並列化なし：６７．８
並列化あり：５２．７

並列化によりゲート数は増加したが，高速化は実現した．また，従来型が収束までに 40 世代，つまり約 80000 クロックかかるのに対し，定常状態型は約 45000 クロック程度で収束する．従来型と定常状態型の比較では，定常状態型の方が収束までが早い．しかし世代という概念がなく同じ個体が多く選ばれるため，全体が同じ値をもつようになり，それゆえ局所解に陥る危険性も高いと思われる．

9. おわりに

今回は，我々がハードウェア記述言語により設計したＧＡＰの仕様についての説明を行なった．今後の課題としては，実際の VLSI の実装と実地での評価が挙げられる．またそれにともない，実用的な評価関数の実装も行なわなければならない．

参考文献

[1] 中村, 小野, “ULSI の効率的な設計法” オーム社 1994
[2] http://www.kecl.ntt.jp/car/parthe/
[3] 小栗 清, 名古屋 彰, 野村 亮, 雪下 充輝, “はじめての PARTHENON” *CQ* 出版社 1994
[4] Micaela Serra, Terry Slater, Jon C. Muzio, and D. Michael Miller. “The analysis of one-dimensional linear xellar automata and their aliasing properties.,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,9(7)*, pp. 767-788, July 1990
[5] 吉田紀彦, 森木俊臣, 安岡智宏,“ SFL による遺伝的アルゴリズム VLSI の設計”, 第 10 回パルテノン研究会, April 1997