九州大学学術情報リポジトリ
Kyushu University Institutional Repository

定常状態型遺伝的アルゴリズムのためのVLSIアーキテクチャ

吉田, 紀彦
九州大学大学院システム情報科学府知能システム学専攻

安岡, 智宏
九州大学大学院システム情報科学府知能システム学専攻 : 修士課程

森木, 俊臣
九州大学大学院システム情報科学府知能システム学専攻 : 修士課程

https://doi.org/10.15017/1498341

出版情報：九州大学大学院システム情報科学紀要. 3 (1), pp.69-74, 1997-12-22. 九州大学大学院システム情報科学研究院
バージョン：
権利関係：

九州大学大学院
システム情報科学研究科報告
第3巻　第1号　平成10年3月

Research Reports on Information Science and
Electrical Engineering of Kyushu University
Vol.3, No.1, March 1998

定常状態型遺伝的アルゴリズムのためのVLSIアーキテクチャ

吉田紀彦* ・ 安岡智宏**・ 森木俊臣**

VLSI Architecture for Steady-State Genetic Algorithms

Norihiko YOSHIDA, Tomohiro YASUOKA and Toshiomi MORIKI

(Received December 22, 1997)

Abstract: Genetic Algorithms (GA) are a class of stochastic optimization algorithms based on natural evolution. We have designed "GAP" (Genetic Algorithm Processor), a general-purpose VLSI for GA. It is based on Steady-State GA, and achieves better performance by pipeline parallelization than other GA-VLSIs, which are based on conventional generation-based GA. Moreover, GAP introduces "Simplified Tournament Selection" scheme for implementing the selection module. This scheme is proved much simpler, faster, and better in convergence compared to "Roulette Wheel Selection" scheme, which is widely used in other GA-VLSI's.

Keywords: Genetic algorithms, VLSI hardware

1. はじめに

確率的探索および最適化の手法として近年急速に注目を集めているのが，遺伝的アルゴリズム（Genetic Algorithms; 以下GAと略す）である[1]．これは生物集団の適応進化に範を得た手法であり，個別の解候補を「個体」ないし「遺伝子」とみなし，多数の個体からなる集団に交叉や変異などの「遺伝的操作」と適応度に基づく「選択淘汰」を施しながら個体を進化させる，すなわち解候補の質を向上させていって最適解を目指すというものである．GAは，解候補の「個体」への写像および適応度の評価関数が用意できれば，どのような問題にも適用できる汎用の手法である．

GAでは高品質の解を得るためには集団内の個体数を増やす必要があり，これは処理性能の低下を招く．一方で，集団内の個体（解候補）を独立に評価するため，本質的に並列処理との親和性が高い．そこで，並列分散処理によってGAを高速化する研究が数多く進められている．しかし，ソフトウェアによる実装では高速化には限界がある．

GAの通常の実現では，個体はビット列で表現される．そして，これに施す遺伝的操作そのものは単純である．そこで，GAを直接VLSI化してハードウェアで高速実行しようとする研究が始まっている．特定の問題に特化したもの[2),3),4)]，汎用を目指したもの[5),6),7),8)]，ともに幾つかの事例が報告されている．

本研究はこの一連の流れの延長上にあるもので，問題を特定しない汎用のGA-VLSIの実現を目指している．他のGA-VLSIと異なり，定常状態GA（Steady-State GA）という方式の採用によって，パイプライン並列処理による高速化と収束性能の向上を達成した[9),10),11),12)]．さらに，「単純化トーメント選択」（Simplified Tournament Selection）と名付けた選択方式を新たに導入して，回路量の削減とさらなる高速化，および収束性能の向上を実現した[13)]．

以下，まずGAの概要を紹介した後，我々が研究を進めているGA-VLSIである"GAP"（Genetic Algorithm Processor）について述べる．そして，他のGA-VLSIで採用されている従来からの方式に対して，定常状態GA方式および単純化トーナメント選択方式に基づく設計について，回路設計およびシミュレーション実験の結果を示す．

2. 遺伝的アルゴリズム

遺伝的アルゴリズム（GA）の典型的な手順は次のようになっている．

1. 集団の中から適当な個体を選別
2. 遺伝的操作を適用して増殖
3. 集団の更新
4. 1. に戻って繰返し

これは解くべき問題に係わらず同一である．個体（遺伝子）はビット列であり，集団は数十から数千の個体を含む．初期集団はランダムな個体から構成される．この個体集団に選択淘汰による進化過程をそのままモデル化した一連の操作を施し，次世代集団を生成することを繰り返す．これを一定回数，または収束するまで行う．

解くべき問題が与えられた時に必要なのは，問題に対する解候補をビット列に写像する規則，および解候補の適応度すなわち良し悪しを決める評価関数である．裏返

平成9年12月22日受付

* 知能システム学専攻
** 知能システム学専攻修士課程

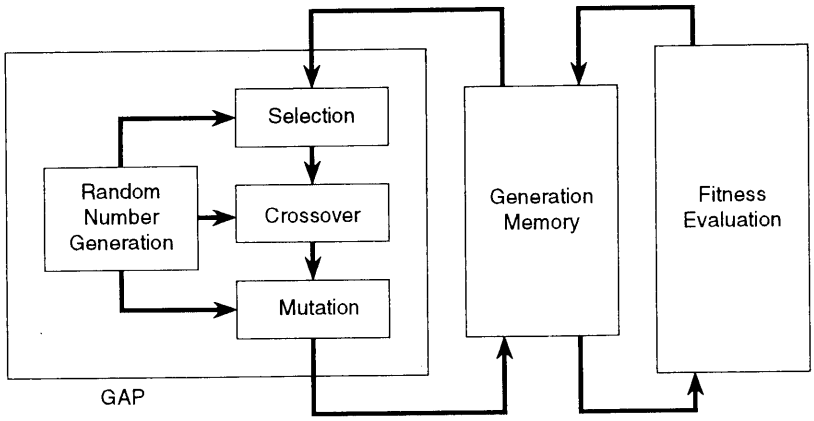

Fig.1　Overview of GA-VLSI "GAP"

せば，写像規則および評価関数が作れれば，どのような問題にもGAを適用することができる．

遺伝的操作には，次の交叉と変異がある．

交叉　2本のビット列について，ランダムな位置での部分列の入れ替え

変異　ランダムな位置のビットの反転

これによって確率的に新たな解候補を生成する．交叉のみのGAもあるが，理論的に可能な全ての解候補の探索を保証するためには，変異もなければならない．適応度の高い個体を選別して増殖させることを繰り返して，高適応度個体すなわちより質のよい解候補を得る．

探索・最適化手法としてのGAの長所は

- 汎用で適応範囲が広い．
- 組合わせ爆発を起こすような広大な探索空間を持つ問題にも適用できる．

一方で短所は

- 局所探索は不得手．
- 解品質を高めるため個体数を増やすと計算量が増える．

3. GAP の基本構成

GAは次のような性質を持つことから，近年VLSI化が検討され始めている．

- 個体がビット列で表現されており，また遺伝的操作も単純である．
- 個体の操作や評価が個体ごとに独立しており，並列化やパイプライン化の可能性が高い．

GA手順そのものは問題から独立であるので，これをVLSI化することで汎用GA-VLSIが実現できる．問題に依存する適用度評価のVLSIを別途作成して組み合わせて用いることで，その問題に特化したハードウェアGAシステムを構築することになる．

本研究で設計開発を進めているGA-VLSIであるGAPは，先に示したGAの手順に従う形で **Fig. 1** に示す全体構成を持つ．これは基本的には，他の汎用的GA-VLSIにも共通するものである．間に介在する世代メモリは世代交代GAでは2世代分の，定常状態GAでは1世代分の個体集団を格納するものであり，各個体をビット列とその適応度評価値の対で格納している．以下にGAPの基本形について，それを構成するモジュールについて述べる．

3.1 乱数生成部

疑似乱数の生成は，多用されている剰余計算ではなく，現在最も良質の乱数を生成するとされ，かつVLSI化にも向いている有限状態オートマトンによっている [14]．

この方法では，乱数ワードの i 番目のビット S_i について，次の状態 $S_i{}^+$ を下記の規則 Rule 90 および Rule 150 に従って得ていく（$\oplus$ は排他的論理和）．

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

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

これをワードのMSBからLSBまで交互に適用する．なお，MSBとLSBの部分は例外的に必ず150をとり，ワードからはみ出たビットは0とみなす．すなわち，次のような順で適用することになる．

$$150, 150, 90, 150, 90, \ldots (\text{交互}) \ldots, 90, 150$$

3.2 選択部

本節では，まずルーレット選択法（Roulette Wheel Selection）について説明する．これは他のGA-VLSIの研究開発事例でも1つ [8] を除いて採用されている方式である．この選択部の改良については，後に改めて述べる．

ルーレット選択法の手順は次のようになっている.

1. 集団内個体の適応度を全て合計し，その合計値までの乱数を生成する.
2. 個体の適応度を順に累積していき，累積値が1.の乱数を越えたところで，その個体を選択する.

3.3 遺伝的操作部

第2章で示した遺伝的操作の交叉と変異を実行する.GAの中心的な処理であるが，回路そのものは，乱数をデコードして個体ビット列に対するインデクスを得，それに従って入替えやビット反転を行う組合せ回路である.この処理を，与えられた乱数値が内部的に持つ閾値を越えた場合に適用する.

3.4 適応度評価部

個体のビット列を入力として取り，その適応度を出力する.これは解くべき問題に応じて作成することになる.ただし，出力する適応度は，0または正数，適応度が高いほど大きい，の2条件を満たしていなければならない.関数最少化問題や最適化問題のように評価値が小さいほど適応度が高い場合など，条件を満たすように数値を正規化する必要がある.

4. 定常状態GAとパイプライン化

4.1 定常状態GA

基本的なGAでは，世代ごとに全個体を一括して集団を更新する.これに対して，世代の概念をなくして新しい個体が生成されるごとに漸進的に集団を更新していくのが「定常状態GA」(Steady-State GA) である.具体的には，集団内で評価値が最悪の個体を新しい個体で順次置き換えていく.収束の性質などが理論的な方面で研究されており，一部のGAシステムでも採用されている.

従来の世代交代GAでは，1世代全体の適応度評価と選択・遺伝的操作とが，世代単位に交互になされることになる.これに対して定常状態GAに基づくと，適応度評価と選択・遺伝的操作とがパイプライン的になされることになり，並列化による処理効率の改善が期待できる.

4.2 手法のVLSI化と回路の変更

定常状態GAによるパイプライン化に向けたGA-VLSI回路の本質的変更は，世代メモリが2つから1つに減らせることである.世代交代GAでは個体集団を一括して更新するため，現世代と次世代の2セットを保持する必要がある.これに対して定常状態GAでは集団を順次更新していくため，1セットしか必要でない.

最大の問題は，置き換えられるべき「集団内で評価値が最悪の個体」の同定である.ルーレット選択方式に従う定常状態GAでは，その度に集団全体を走査またはソートしなければならず，このオーバーヘッドは性能を大幅に低下させパイプライン並列化の効果を損なう.そこでこれに代えて，ルーレット選択で走査した範囲の個体の内から最悪のものを選び出すことにした.必ずしもその時点での最悪の個体を選び出すことにはならなくなるが，ルーレット選択の開始位置を前回の終了位置として順次ずらしていくことで，いずれは集団全体を走査することになる.

4.3 パイプライン化

世代交代GAでは，選択部と遺伝的操作部のパイプライン化が可能である.選択処理が終了したら遺伝的操作部に選択された個体を渡し，すぐに次の選択を開始する.

これに対して定常状態GAの採用によって，選択部，遺伝的操作部，適応度計算部の3モジュールについてパイプライン化が可能になる.選択処理が終了したら遺伝的操作部に個体を渡して次の選択を開始する.一方で，遺伝的操作処理が終了したら適応度計算を行う.ただし，世代メモリが1つなのでアクセス競合を避けるために，選択処理の終了まで待ってから結果の書き込みを行う.その他のアクセス競合は，レジスタや回路の多重化で対処することができる.

5. 選択部の改良

ルーレット選択に基づいたGA-VLSI回路では，ゲート量・実行ステップ数とも選択部が大半を占めて大きなボトルネックとなっていること，収束性能もさほど良くないことが，GAPの試作設計とそのシミュレーションから明らかになった.一方でGAの理論的研究では，ルーレット選択に代わる様々な選択方式が検討されつつあり，より収束性能のよい方式も提案され始めている[15].

そこで，収束性能がよくかつVLSI化に適した，すなわちゲート量・実行ステップ数とも少なくて済む選択方式を目指して，理論的研究の成果も参考に，「単純化トーナメント選択」(Simplified Tournament Selection) と名付けた方式を新たに導入した.

5.1 単純化トーナメント選択

これは下記の手順に従う非常に単純な選択方式である.

1. 集団内から個体をランダムに2つ選ぶ.そして，適応度の良い方を選択個体1，悪い方を淘汰個体1とする.
2. 集団内から個体をランダムに2つ選ぶ.そして，適応度の良い方を選択個体2，悪い方を淘汰個体2とする.
3. 選択個体1と2を親とし，遺伝的操作を施して，子を2つ作る.
4. 結果の子2つで淘汰個体1と2を置き換える.淘汰

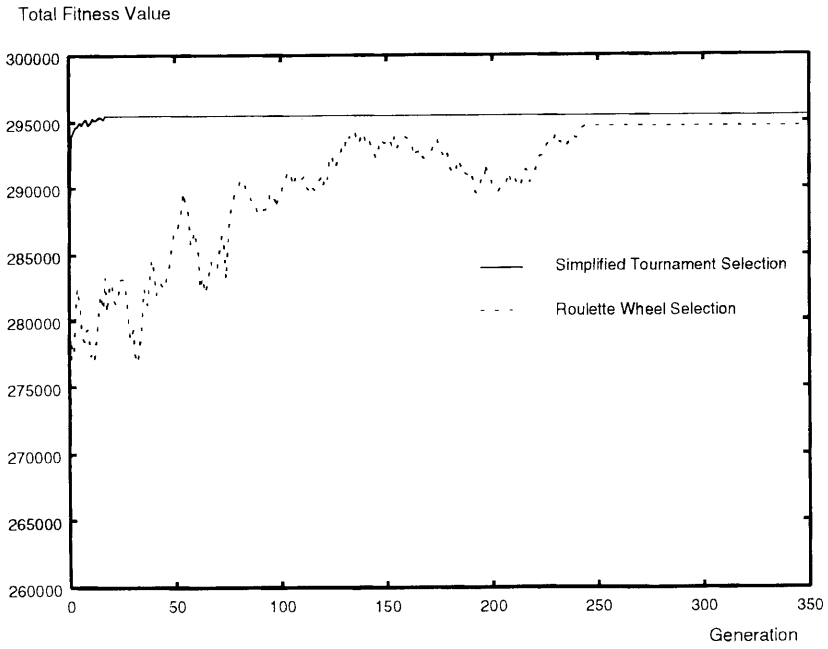

Fig.2　Convergence comparison

個体1と2は捨てる.

いわゆるトーナメント選択方式で部分集団の大きさを最少の2にしたものに似ることから,「単純化トーナメント選択」方式と名付けている. なお, 淘汰個体を子2つで置き換えるのでなく, 親子4つの個体の内から適応度の高い2つを選んでそれで置き換えるようにすると, いわゆるエリート保存方式になるが, ここではそこまでは採用していない.

5.2 ソフトウェアでの予備実験

単純化トーナメント選択方式の収束性能を事前に評価するために, ソフトウェアで予備実験を行った. 対象問題として用いたのは, GAの研究でシステム評価に標準例題として多用されている"De Jongの標準問題集"の内の1と2である. これらは関数最少化問題, すなわち関数値を最少にするような引数の組を求めるもので, 関数はそれぞれ次の形をしている.

De Jong's Function No. 1

$$f1(x_1,x_2,x_3) = {x_1}^2 + {x_2}^2 + {x_3}^2$$

$$(-5.12 \leq x_i < 5.12)$$

De Jong's Function No. 2

$$f2(x_1,x_2) = 128 * ({x_1}^2 - x_2)^2 + (1 - x_1)^2$$

$$(-2.048 \leq x_i < 2.048)$$

単純化トーナメント選択とルーレット選択の収束過程の比較結果を, 関数2について **Fig. 2** に示す. 関数1についても同様の結果を得ている. なお, この図の縦軸(適応度) g は,「大きいほど良い」とするために, 関数値 $f2$ に対して $g = -\alpha * f2 + \beta$ の形でスケーリングしたものである.

図に見られるように, 単純化トーナメント選択方式は非常に単純であるにもかかわらず, ルーレット選択方式と比べて収束性能において遜色ない.

6. 回路設計とシミュレーション

6.1 ハードウェア記述言語による VLSI 設計

GAPの設計は, VLSI CADの分野で近年急速に発展しているハードウェア記述言語を用いて行った. ここで用いたのはNTTで開発されたシミュレーション・回路合成の両方の機能を持つ言語SFL, およびこれに基づくCADシステムPARTHENONである [16),17),18)].

ハードウェア記述言語によるVLSI設計はプログラム的な記述からハードウェアを導出するもので, ソフトウェアになぞらえると, 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, 論理合成によるネットリスト導出はコンパイラによる機械語生成に相当する.

本研究では, SFLによるGAPの設計記述についてシミュレーションと回路合成を行い, 回路規模・処理性能の評価, パイプライン化・選択法の比較などを行った. 対象問題として用いたのは, 先に述べたソフトウェア実験と同じく, De Jongの標準問題の1と2である.

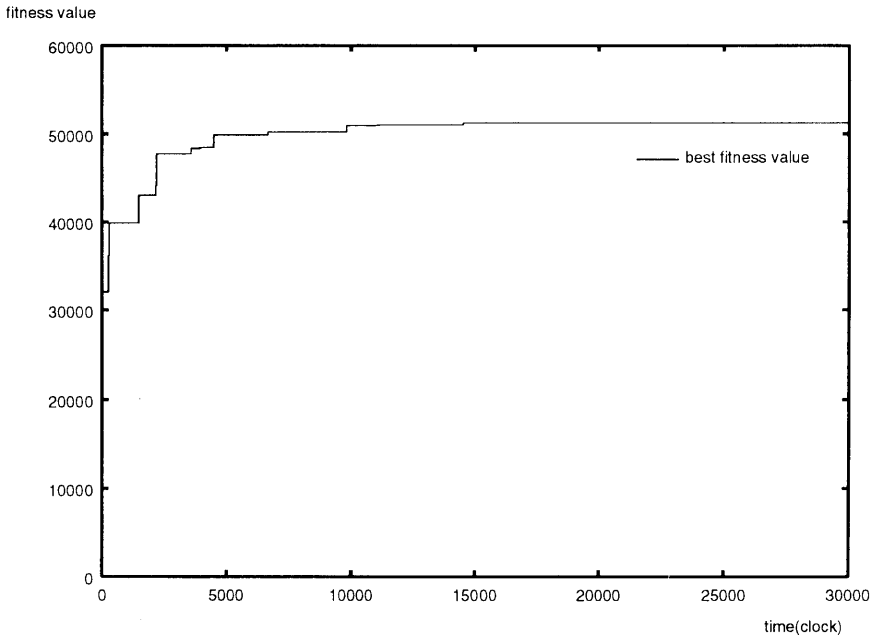

Fig.3 Convergence on GAP: De Jong's F2

6.2 回路規模の評価

GAPの実験仕様は次のように定めた.

集団内個体数: 64
個体ビット長: 32
評価値ビット長: 24

そして，次の5種類のVLSIを設計した.

GB+RW	世代交代・ルーレット選択
GB+RW+P	世代交代・ルーレット選択・パイプライン化
SS+RW	定常状態・ルーレット選択
SS+RW+P	定常状態・ルーレット選択・パイプライン化
SS+ST+P	定常状態・単純化トーナメント選択・パイプライン化

関数1について，これら各々の適応度評価部も含めた総ゲート数を次に示す.

GB RW	GB RW P	SS RW	SS RW P	SS ST P
23602	24464	22590	24114	20762

そして，選択部以外のモジュール各々のゲート数を次に示す.

乱数生成	交叉	変異	評価 ($f1$)	評価 ($f2$)
599	590	391	7335	7314

なお，乱数生成部について，特に単純化トーナメント選択方式ではGA処理時には32ビットをインデクシングできる5ビット乱数があればよいが，初期個体集団の生成に32ビット乱数が必要なので，その大きさのモジュールを用意している.

パイプライン化では，アクセス競合を避けるために回路の多重化を行っており，そのためにゲート数が増加している．単純化トーナメント選択はルーレット選択に比べて単純なため，ゲート数が減少している.

6.3 処理性能の評価

実行速度については，シミュレーション実験で実行クロック数を計測した．関数1について，個体あたりの平均クロック数を次に示す.

GB RW	GB RW P	SS RW	SS RW P	SS ST P
* 64.9	* 60.5	67.8	52.7	22.0

なお，"*" の世代交代型については，世代あたりの平均クロック数から換算した値である.

特に選択方式についてクロック数を比較すると，1回の選択にルーレット選択では平均で概ね40クロックほど要するのに対して，単純化トーナメント選択では6クロックと約 1/6 で済んでいる.

また，収束までに要するクロック数は，次の通りであった.

世代交代・ルーレット選択	約80,000
定常状態・ルーレット選択	約45,000
定常状態・単純化トーナメント選択	約20,000

GAPの「定常状態・単純化トーナメント選択」の収束過程を，関数2について **Fig. 3** に示す．

なお，この実験では対象問題が単純であったが，これが複雑な場合には，処理性能は適応度評価計算の実行速度に依存するものと予想される．

7. おわりに

GAのVLSI化における新たな方式，すなわち定常状態GAによるパイプライン化，および選択方式の単純化による回路量と処理効率の改善を提案し，プロトタイプを設計するとともに，その可能性と効果をシミュレーションによって評価した．

本研究では，VLSI設計に際してハードウェア記述言語を採用することで，小規模のプログラミングおよびハードウェア製作の経験しかなくても，新しいアイデアのハードウェアを設計・評価できることが明らかにできたことも，成果の一つといえる．

今後の課題としては，実際のVLSI実装と実地での評価が最重要であり，カスタムVLSI製作に近年多用されているFPGA (Field Programmable Gate Array) 技術を利用した実現に向けて準備を進めている．現在の設計には各所に改良の余地があり，それらへの対応も行いつつある．そして，実用規模の問題への適用実験を通じて，本研究の成果を実証する予定である．

将来的には，動的移住方式[19]など，並列分散GAに関するこれまでの研究成果も応用して，次の2レベルで並列分散処理を行う“Multi-GAP”の実現に向けて研究を進めていく．

1. 1台の遺伝的操作モジュールに複数の適応度評価モジュールを接続して並列評価値計算を行う細粒度並列処理
2. 1. のセットを複数個バス接続して複数集団の並行進化を行う粗粒度並列分散処理

謝辞

本研究にアドバイスを頂いたP. G. Anderson (Rochester Institute of Technology)，山村雅幸（東京工業大学），荒木涼二（九州大学，現在沖電気株式会社），犬尾武（九州大学，現在日本電気株式会社）の各氏に感謝する．

参考文献

1) 北野編, 遺伝的アルゴリズム, 産業図書 (1993)
2) B. C. H. Turton, T. Arslan and D. H. Horrocks, “A Hardware Architecture for a Paralle Genetic Algorithms for Image Registration”, Proc. IEE Colloquium on Genetic Algotithms in Images Processing and Vision, pp. 11/1-11/6 (1994)
3) B. C. H. Turton and T. Arslan, “A Parallel Genetic VLSI Architecture for Combinatorial Real-Time Applications - Disk Scheduling”, Conf. Proc. on Genetic Algorithms in Engineering Systems: Innovations and Applications, pp. 493-500 (1995)
4) 大島, 松本, 平木, “再構成可能な遺伝的算法エンジンの研究”, Proc. Third Japanese FPGA/PLD Design Conference and Exhibit, pp. 541-548 (1995)
5) S. D. Scott, A. Samal and S. Seth, “HGA: A Hardware-Based Genetic Algorithm”, Proc. 1995 ACM/SIGDA Third Intl Symp. on FPGA, pp. 53-59 (1995)
6) I. M. Bland and G. M. Megson, “Implementing a Generic Systolic Array for Genetic Algorithms”, Proc. First Online Workshop on Soft Computing (1996)
7) 井上, 稲富, 佐野, 高橋, “SIMD型遺伝アルゴリズムと専用プロセサ要素の設計”, 情報処理学会「計算機アーキテクチャ」研究会報告 116-8 (1996)
8) B. Shackleford, E. Okushi, M. Yasuda, H. Koizumi, K. Seo, T. Iwamoto and H. Yasuura, “A High-Performance Genetic Algorithm Machine”, Proc. IPSJ Symp. on Information Systems and Technologies for Network Society, pp. 113-120 (1997)
9) 森木, 遺伝的アルゴリズムVLSIの全体設計とシミュレーション評価, 九州大学卒業論文 (1997)
10) 安岡, 遺伝的アルゴリズムVLSIのハードウェア記述言語による設計, 九州大学卒業論文 (1997)
11) 吉田, 森木, 安岡,“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論文集, pp. 63-70 (1997)
12) N. Yoshida, T. Moriki and T. Yasuoka, “GAP: Generic VLSI Processor for Genetic Algorithms”, Proc. Second Int'l ICSC Symp. on Soft Computing, pp. 341-345 (1997)
13) 安岡, 森木, 吉田, “定常状態GAを直接実行するVLSIハードウェアの設計と評価”, 情報処理学会「数理モデル化と問題解決」研究会報告 (1997)
14) M. Serra, T. Slater, J. C. Muzi and D. M. Miller, “The Analysis of One-Dimensional Linear Cellular Automata and Their Aliasing Properties”, IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, Vol. 9, No. 7, pp. 767-788 (1990)
15) 佐藤, 小野, 小林, “遺伝的アルゴリズムにおける世代交代モデルの提案と評価”, 人工知能学会誌, Vo. 12, No. 5, pp. 734-744 (1997)
16) 中村, 小野, ULSIの効率的な設計法, オーム社 (1994)
17) 小栗, 名古屋, 野村, 雪下, はじめてのPARTHENON, CQ出版社 (1994)
18) http://www.kecl.ntt.jp/car/parthe/
19) N. Yoshida and R. Araki, “Efficient Implementation of Distributed Genetic Algorithms on Network of Workstations”, Proc. Second Int'l ICSC Symp. on Soft Computing, pp. 336-340 (1997)