

パラメータの適応的調整を伴う遺伝的アルゴリズムのハードウェア化†

4 F - 5

中山喜勝

八田浩一

若林真一

小出哲士

広島大学 工学部

1 まえがき

工学における様々な分野において複雑な制約を持つ多くの大規模最適化問題が知られているが、それらの問題は数理的手法や組合せ最適化アルゴリズムを用いて実用的な計算時間で最適解を求めることは困難なため、ヒューリスティック手法を用いて解を求めることが一般的である。このようなヒューリスティック手法の一つとして遺伝的アルゴリズム (Genetic Algorithm, GA) が知られている [1]。GA は生物の進化の過程にヒントを得た手法であり、アルゴリズムの実行中は複数の解 (個体) を保持し、これに遺伝的操作を適用して解の改善を行なう。

GA は多くのパラメータを持つため、それらのパラメータの値を調整して GA の探索能力を最大限に利用することは一般に難しい。そこで Davis[2] らは GA を用いて問題を解くのと並行して、内部パラメータの値を動的に調整し、効率的に GA を実行する方法を提案し、比較的良好な結果を得ている。また著者らは文献 [3] においてエリート度概念を提案し、これに基づく複数の交叉手法の適応的選択手法を提案している。

本稿では著者らが提案したエリート度に基づく交叉の選択手法を組み込んだ GA のハードウェア化を議論する。GA をハードウェアで実現することによりソフトウェアによる実行に比べて、非常に高速なアルゴリズムの実行が可能となる。また、並列処理やパイプライン処理などの導入によりさらに効率の良い解の探索が期待できる [4]。一方、GA の適用対象問題を特定すればその問題に対する GA をハードウェア化するのは容易であるが、適用対象問題を特定しない場合、ハードウェア化した GA に汎用性を持たせる必要が生じる。これを解決するために、問題に依存した部分については FPGA 等の再構成可能なハードウェアで実現し、問題に依存しない部分は通常のハードウェアで実現することが考えられる。本稿で提案する GA のハードウェア化においても、問題に依存する部分は FPGA、依存しない部分は LSI で実現することとしている。

2 エリート度を用いた適応的手法

実際の問題に対して GA の探索能力を最大限に引き出すためには、適切な遺伝オペレータの選択とオペレータ適用確率 (パラメータ) の選択が必要であり、従来のパラメータ調整では、経験と勘に基づいてパラメータをある値に絞り込んだ後で、実験を繰り返すことにより調整を行ってきた。しかし、これには膨大な時間を必要とし非常に無駄が多い上に、適切なパラメータ設定が難しいという問題点があった。そこで最初からパラメータの値を固定しておくのではなく、GA の実行中に動的に変更し、適応させていく手法が、近年、多く提案されている。

その手法の一つとして著者らはエリート度という概念を導入し、GA 実行中に適応的に交叉手法を選択する方法を提案している。最大化問題に対しては世代 T における個体 x_i^T のエリート度は以下の式で表すことができる。

$$Elite_degree(i) = \frac{\sum_{j=1}^{level_max} \{|Elite_i^T(j)| \times \beta^j\}}{\sum_{j=1}^{level_max} \{|Anc_i^T(j)| \times \beta^j\}}$$

ここで、 $Anc_i^T(j)$ は個体 x_i^T の世代 $T-j$ における先祖の集合、 $Elite_i^T(j)$ は、 $Anc_i^T(j)$ の中でエリートである個体の集合、 $\beta(0 \leq \beta \leq 1)$ はエリート影響度係数とする。 β の値を小さくすることにより遠い祖先の影響を減少させることができる。最小化問題に対するエリート度の定義も同様に行なうことができる [3]。

エリート度に基づく交叉方法の選択については、本研究ではハードウェア化が容易なように、ある個体がエリートかどうかの判定を文献 [3] の元の手法より簡略化し、以下の式に基づいて行なう。すなわち最大化問題に対しては

$$Elite_i^T(j) = \{x_k^{T-j} | x_k^{T-j} \in Anc_i^T(j),$$

$$f_{ave}^{T-j} + \alpha \times (f_{best}^{T-j} - f_{ave}^{T-j}) \leq f(x_k^{T-j})\}$$

ただし、 $f(x)$ は個体 x の評価 (フィットネス) 値、 f_{ave}^T 、 f_{best}^T は世代 T における個体の評価値の平均と最良値を表す。最小化問題に対しても同様に定義できる。エリート度を用いて、以下の操作により交叉方法を適応的に選択する。

[交叉方法の適応的選択]

- step 1: 交叉を行なう 2 つの親についてエリート度の和を求める。
- step 2: 2 つの親がエリートかどうかを判定し、もしエリートならばそれぞれのエリート度の和に 1 を加える。
- step 3: エリート度の和が一定のしきい値以上ならば 2 点交叉を実行し、そうでなければ一様交叉を実行する。

エリート度の計算をハードウェアで効率良く行なうために、各世代の個体がエリートかどうかの全ての場合に対してエリート度をあらかじめ計算しておく、計算結果を ROM に格納しておく。

提案手法のフローチャートを図 1 に示す。

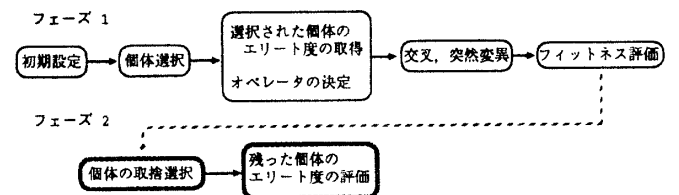


図 1

†“Hardware Imprementation of an Adaptive Genetic Algorithm”, Yoshikatsu NAKAYAMA, Koichi HATTA, Shin’ichi WAKABAYASHI, Tetsushi KOIDE, Faculty of Engineering, Hiroshima University. e-mail:neal@ecs.hiroshima-u.ac.jp

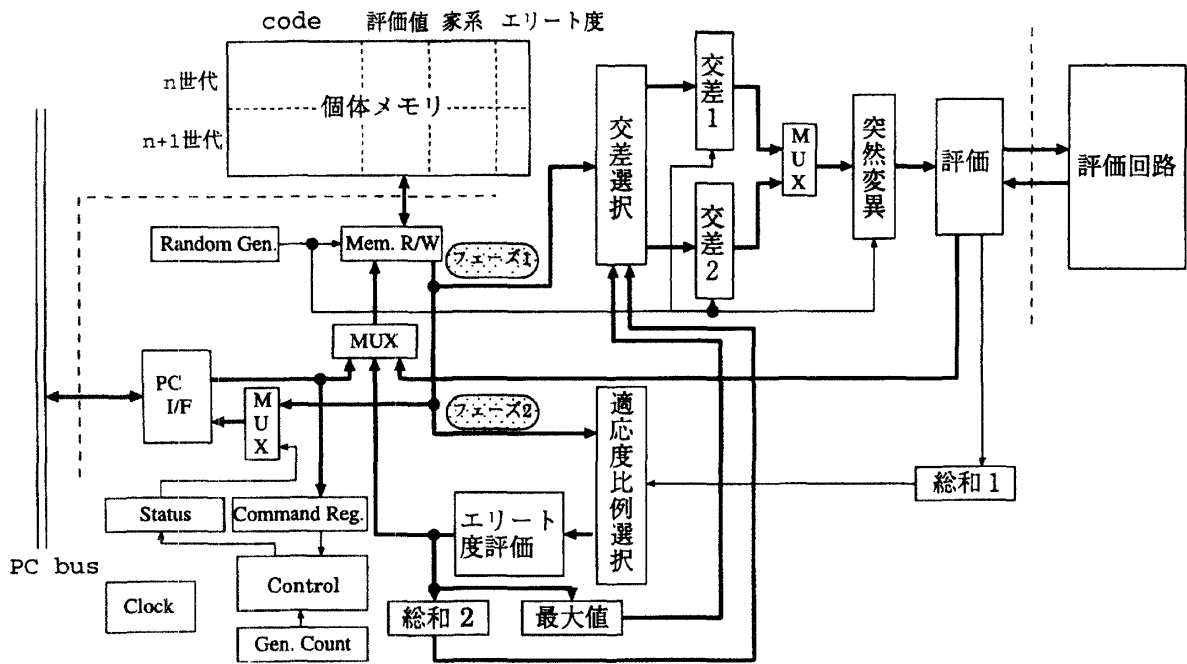


図 2

3 回路構成

回路の全体構成を図 2 に示す。回路設計は Verilog-HDL で記述することにより行なう。

GA の実行過程は二つのフェーズからなっている。まず最初に初期個体集合等の初期設定が終るとホストコンピュータから実行開始の信号が送られ、乱数によって選択された個体がメモリから読み込まれ、交差選択モジュールに送られる。そこで個体のエリート度に応じて 2 つの交叉手法のうちどちらかが選ばれ、交叉モジュールで処理される。その後、突然変異が施され、評価モジュールにおいて個体のフィットネス評価を行なう。評価された個体は個体メモリに格納される。評価モジュールからは評価値が評価値合計モジュールに送られ、合計されていく。

決められた人口数だけ個体生成のフェーズを実行すると、次に世代更新フェーズが開始される。このフェーズでは、まず個体メモリより個体が選択モジュールに送られる。選択モジュールでは個体のフィットネスと評価値合計モジュールで計算された評価値の平均よりルーレット選択が行なわれる。ルーレット選択では個体のフィットネスからフィットネスの平均値を引いていき、その値が 0 未満になるまで個体の選択機会を与える。選ばれた個体はエリート度計算モジュールでエリート度が評価され、個体メモリに戻される。またその世代の個体のフィットネスの総和と最大値も計算される。

この 2 つのフェーズを与えられた世代数繰り返し、動作を終了する。

このハードウェアは問題に依存しない部分は LSI により実現し、フィットネス評価部分は様々な問題に対して適用可能にするために FPGA で実現する。

4 あとがき

本稿では GA において、各個体における潜在的な優劣の度合を示す指標であるエリート度を用いて、アルゴリズムの実行中に交差手法を適応的に選択する手法を導入し、これを専用ハードウェア化し、より効率的かつ高速な解の探索を実現することを提案した。

今後、実際にこの回路を設計・製作し、評価・検証を行なっていく予定である。

文献

- [1] D.E.Goldberg: "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley Publishing Company (1989).
- [2] L.Davis: "Adapting operator probabilities in genetic algorithms", Proc. the 3rd International Conference on Genetic Algorithms, pp. 61-69 (1989).
- [3] 松田憲治, 八田浩一, 若林真一, 小出哲士: "遺伝的アルゴリズムに対するメタヒューリスティクスに基づくパラメータ値設定手法", 情処学研報 MPS-11, pp. 17-24 (1997).
- [4] S. D.Scott, A. Samal and S. Seth: "HGA: A Hardware-Based Genetic Algorithm", Proc. ACM/SIGDA Third Int'l Symposium on FPGA, pp. 53-59 (1995).
- [5] L.Davis: "Handbook of Genetic Algorithms", Van Nostrand Reinhold (1991).