完全グラフ表現による遺伝的アルゴリズム

3P－3

井手 一郎,　毛利 隆夫,　田中 英彦
{ ide,mohri,tanaka} @mtl.t.u-tokyo.ac.jp
東京大学工学部*

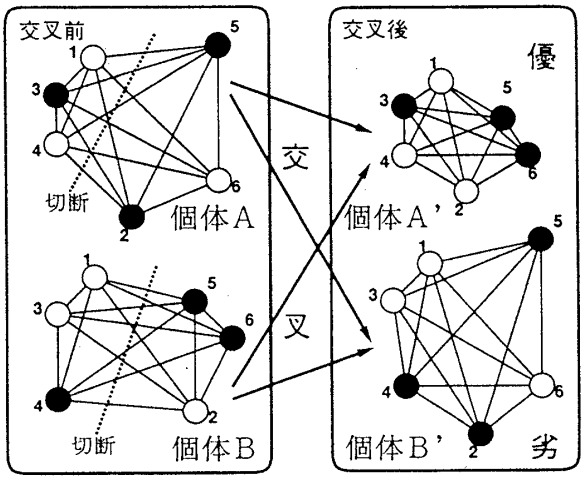
1　はじめに

従来の遺伝的アルゴリズムでは、主に遺伝子を一次元に並べ、ランダムな切断による交叉を行っていたが、それではよい性質をもつ遺伝子を散逸させかねず非効率的である。そこで、よい性質をもつ遺伝子の間での切断をなるべく避け、次世代に残すようにすることにより、効率的な進化を行おうと思い、完全グラフを用いたシステムを構築した。

そこで、簡単な応用問題 (ナップザック問題) の実験を行い、結果としては、処理速度には若干問題が残るものの、収束値や収束速度では一定の成果が得られた。

2　完全グラフ表現を用いた交叉

従来の一次元表現した遺伝子をランダムに切断する交叉では、特定の (複数の) 遺伝子座に評価値をよくする情報が存在することが分かっていても、交叉の際に散逸してしまい、次世代には凡庸な個体が生じるにとどまってしまうことが多発すると考えられる。そこで、遺伝子座を、評価値をよくする情報が存在するもののグループと、そうでないもののグループとに分けて、その境界で交叉すればよいと考えた。そうすれば、交叉の結果、評価値をよくする遺伝子が散逸しにくくなるだろう。このようなグループ分けをするためには、隣接する遺伝子座の組のみではなく、あらゆる遺伝子座間の組の結合係数を考えなければならない。これは、完全グラフを用いることによって実現される (図 1)。



注：図中の番号は遺伝子座の番号

図 1: 完全グラフを用いた交叉

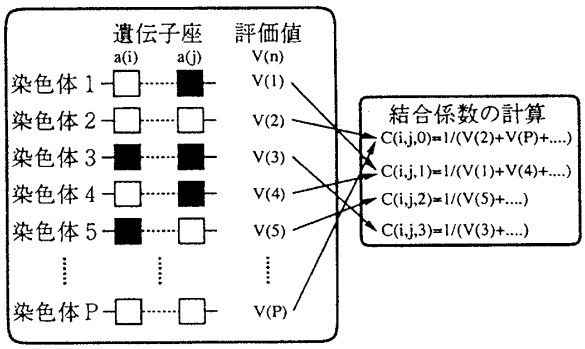


図 2: 結合係数の計算法

ここで述べた結合係数を決めるために、図 2に示すような手法を用いている。この手法では、あらゆる遺伝子座間の組について全染色体に渡って調べることになるので、遺伝子長の累乗と染色体数に比例した時間がかかり、全体の処理時間に大きな影響を及ぼすことになる。よって、完全グラフ交叉は、よい遺伝子を散逸させずに残せるという利点がある反面、単純交叉と比較して処理速度は相当遅くなるのが欠点である。

なお、以下、完全グラフを用いたこの手法を、「完全グラフ交叉」と呼ぶ。

3　実験とその結果

以上のような方針で実際にシステムを構築し、性能を評価するために、最初に 2 通りの簡単な例題について、次に具体的な応用問題としてナップザック問題について実験を行った。性能を評価するために、1 点でランダムに交叉する単純交叉、複数点でランダムに交叉する複数点交叉と比較した。

なお、遺伝子集団の遺伝子コードの初期値は、平均的な動作を見るために、10 通りの乱数系列で与えた。以下の結果では 10 回の実験の平均値を示す。

*Genetic Algorithm Using Complete Graph Expressions
Ichiro IDE, Takao MOHRI, Hidehiko TANAKA,
University of Tokyo, Department of Engineering,
7-3-1 Hongou, Bunkyo-ku, Tokyo 113, Japan

3.1 簡単な例題への適用

次の 2 つの簡単な例題について実験を行った。

- **例題 1**
  評価関数は、遺伝子中の 1 の個数の割合。
  $$f = \sum_{i=1}^{L} \frac{a_i}{L}$$
  (ただし、L は遺伝子長, a_iは各遺伝子 (0,1 の二値))

- **例題 2**
  パターンマッチング (遺伝子は 0,1 の二値)。評価関数は、どの程度一致しているかの割合。

評価関数の値が、例題 1 では遺伝子座に全く依存しないのに対して、例題 2 では極度に依存している点で、両者の性質は大きく異なる。

結果は、例題 1 では思ったほどの効果は上がらず、ランダムな複数点交叉と大差ない結果しか得られなかった。これは、評価関数が遺伝子座に依存しないので、完全グラフを用いて結合係数を計算してもあまり意味がないからだと思われる。

次に、例題 2 では、完全グラフ交叉の立ち上がりが早くなっていて、収束値も若干高くなっている (図 3)。これは、評価関数が遺伝子座に依存するので、完全グラフを用いて切断する場所を考えていることが効果を上げているのだと思われる。

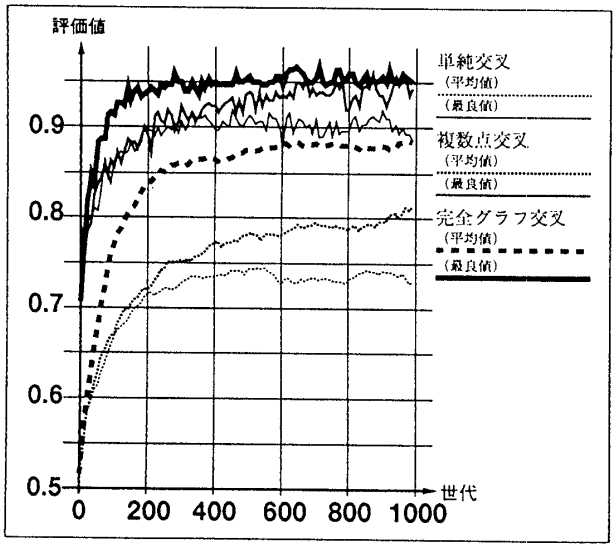


図 3: 例題 2 の実験結果

このように、適用する問題 (評価関数) によって本アルゴリズムの有効性は異なる。

3.2 ナップザック問題への適用

ナップザック問題とは、「重さ」と「値打ち」をもつ物品がいくつかあり、一定の重さまで収納できるナップザックに、どれだけ値打ちの高いものを詰め込めるか、というものである。前節の例題で扱ったものには局所的最適解は存在しないが、この問題では多くの局所的最適解が存在する。

ここで用いた物品は、32 個で、全解探索で得られた真の最適解は、493 であった。

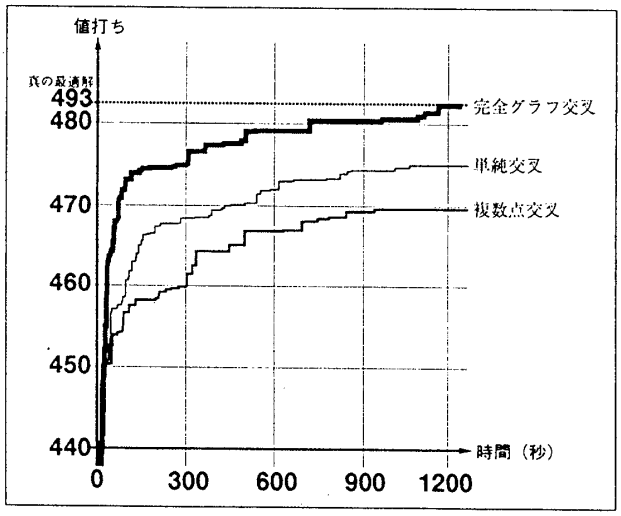


図 4: ナップザック問題の実験結果 (最良値の推移)

当初は 1 世代の処理時間が、完全グラフ交叉では単純交叉の約 9 倍かかっていたので、その段階ではあまり性能がよいとはいえなかった。そこで、結合係数の計算頻度を毎世代から、n 世代に 1 回に下げたところ、ほぼ同等の性能を維持したまま、単純交叉の約 5 倍の処理時間にまで高速化できた。図 4は、この改善された処理時間を加味して、実時間で各手法を比較したグラフである。なお、ここでは計算頻度を 100 世代に 1 回にしたものの結果を示した。

この結果から分かるように、性能を維持できる範囲で結合係数の評価頻度を下げれば、完全グラフ交叉はランダムな交叉と比較して、真の最適解にかなり早く収束するので、一定の成果が上がっているといえよう。

4 おわりに

現在のところ、完全グラフ交叉は、収束値や収束の早さでは一定の成果を上げている。

しかし処理速度は、改善はされたものの、まだ十分ではないので、一層の高速化を図りたい。また、今までは簡単な例題への適用しか行っていないが、今後は他の例題についても実験したい。

参考文献

[1] 北野 宏明; “遺伝的アルゴリズム”, 1993 産業図書