M4　遺伝的プログラミングにおける交叉と個体構造との相性について

今野 大　　元木 達也
新潟大学大学院自然科学研究科　　新潟大学工学部

1　はじめに

遺伝的プログラミング (以下 GP) では、個体に遺伝操作を施すことで、模索的に最適解を探索していく。GP の交叉は Koza[1] 以来、部分木交叉が主流である。しかし、ripple 交叉 [2]、一点交叉 [3] のように、木を斜めに切って対応する部分木の組を全て交換する様なものも提案されている。一方では、GP における個体 (木構造) の重要な部分というのは、終端子、非終端子の選定を始めとした問題の設定によって違ってくる。個体に施す遺伝操作によって、その重要な部分を失っていくのはあまり好ましくないので、問題の設定を考慮して扱う遺伝操作を考える必要がある。そこで本研究では、部分木交叉、ripple 交叉、ripple 交叉の変形版である右 ripple 交叉の 3 つを考え、これら 3 つと個体構造の相性について考察する。ripple 交叉 (以下 左 ripple 交叉)、右 ripple 交叉はそれぞれ左部分木、右部分木が重要な問題に対して効果があるのではないかと予想される。

2　本研究で用いる交叉について

本研究では、次の 3 種類の交叉を考える。

部分木交叉 [1] : 各々の親から節点を任意に選び、その節点を根とする部分木を親同士交換する。

左 ripple 交叉 [2] : 木を先行順に途中まで走査した時、複数の部分木が未訪問になる。この様にしてできる複数の部分木を、2 つの親の間で 1 つずつ組にして交換する (図 1)。木の半分程度の要素が交換の対象となる。この交叉は、木の左部分に比べて右部分が交換されやすくなるので、個体の左部分が重要と思われる問題に対して有効であると考えられる。
(補足)　交換する部分木の数が違うと木が未完成になったり、部分木が余ったりしてしまう。これを防ぐために、部分木のリストを個体の隠れた要素として持たせる。木が未完成の時は部分木リストから部分木を補充し、部分木が余ったら部分木リストに格納する。

右 ripple 交叉 : 木を後行順に途中まで走査した時、複数の部分木が既訪問になる。この様にしてできる複数の部分木を 2 つの親の間で 1 つずつ組にして交換する。右 ripple 交叉の場合は、左 ripple 交叉とは反対に、木の右部分に比べて左部分が交換されやすくなるので、個体の右部分が重要と思われる問題に対して有効であると考えられる。

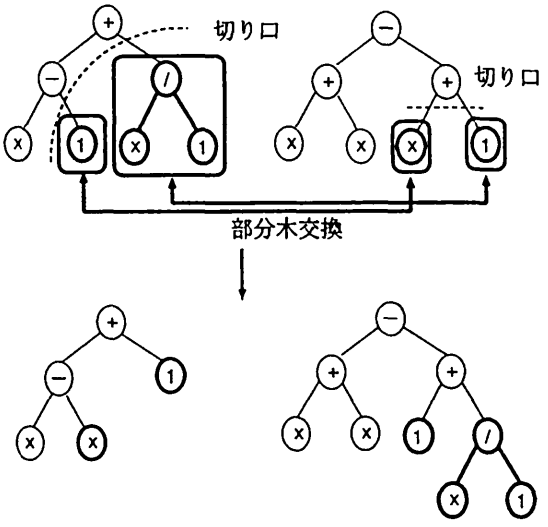

図 1: 左 ripple 交叉

3　人工蟻問題

本稿では、蟻の行動ルールを個体とし、フィールドにある 89 個の餌を制限時間内に全て回収するするような個体を探索目標とする、Santa Fe trail 問題を扱う。個体を評価する際には、餌を見つけたら個体プログラムを最初から実行し直す様にする。ここで扱う非終端子、終端子は表 1 の通りである。この問題設定では、次の理由で個体の左側が重要と考えられる。

- prog の第一引数で餌が見つかると、それ以降の引数が参照されない。
- if-food-ahead の引数に関しても、餌が前にあった状態での行動が重要と思われる。

このような特徴を持った問題を使って、木の左側の部分を壊しにくいと思われる左 ripple 交叉の効果がどれほどのものであるか、また逆に個体の右側を保存し、あえて重要な部分を切り捨てていく右 ripple 交叉を用いたときはどのような結果になるかを次の節で検証する。

表 1: 人工蟻問題の非終端子、終端子

終端子、非終端子	引数	説明
if-food-ahead	2	餌が前にあったら第一引数、なかったら第二引数を実行
prog2	2	第一引数、第二引数の順番で実行を行う
prog3	3	第一引数、第二引数、第三引数の順番で実行を行う
move-forward	0	一つ前に進む
turn-right	0	右を向く
turn-left	0	左を向く

4 実験

前節で説明した人工蟻問題に部分木交叉、左 ripple 交叉、右 ripple 交叉を適用して GP 探索を行い、問題の成功率と、交叉の成功率を計測した。問題の成功率は、最良個体の 100 回試行平均を載せている。交叉の成功率においては、子 2 つのうち、どちらかが親より良い適合度が得られたら成功とみなしており、交叉した場合全てにおいて計測している。

実験結果: 問題の成功率については、左 ripple 交叉を用いた場合が一番良い結果となり、右 ripple 交叉を用いた場合が一番悪い結果となった (図 2)。交叉成功率については、左 ripple 交叉、部分木交叉は中盤においてもある程度交叉が成功してるのに対し、右 ripple 交叉はほとんど成功しなくなっていった (図 3)。個体の左側が重要な問題に対して、左 ripple 交叉は有効で、右 ripple 交叉は相性が悪いという、予想通りの結果となった。

5 考察

本研究では、個体の構造と交叉の相性について考えてみた。そこで、3 種類の交叉法をある程度個体の構造がわかっている問題に適用して、その効果を確かめてみた。その結果、個体の左側が重要な問題設定において左 ripple 交叉は有効であることが示された。本来、交叉というのは重要な部分を失わないようにしていくべきもので、個体の一部がずっと保護される傾向になることから、世代が進むにつれて集団内に同じ部分解を持つ個体が増えてくる。そしてこの部分解の性能によって、集団全体の進化が左右される。しかし、個体の左 (右) 部分が重要と思われる問題における左 (右)ripple 交叉は, 完全には左 (右) 側にある部分解を保護せず、程よく部分解を残していくので、探索が一方向に偏らない有能な交叉であると思う。このことが、左 ripple

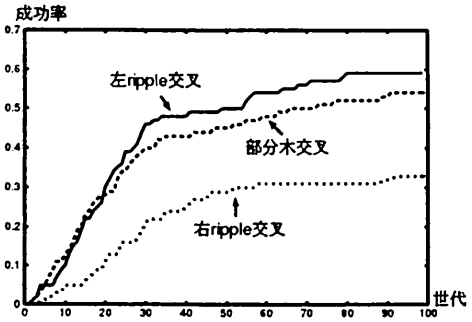

図 2: 成功率の比較

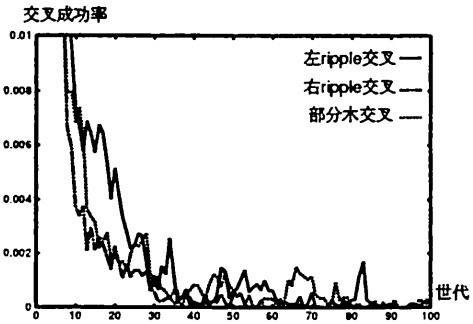

図 3: 交叉成功率の比較

交叉が左側が重要な問題に対して有効であった要因であると考えられる。

また、今回考えた交叉法の別の活用法として、問題の特徴がつかみづらい問題に対してこれらの交叉を行うことにより、その特徴を推測することができる。定義した非終端子について、どの引数が重要なのか、どのような効果があるのかを調べる際の手助けになると思う。

参考文献

[1] Koza.J.R., *Genetic programing*, The MIT Press, 1992

[2] Maarten Keijizer,Conor Ryan,Michael O'Neill, Mike Cattolico, and Vladan Babovic, Ripple Crossover in Genetic Programing, in *Genetic Programing,EuroGP 2001*, Springer, pp.74-86

[3] Nao Tokui and Hitoshi Iba, Empirical and Statictical Analysis of Genetic Programming with Linear Genome, *1999 IEEE International Conference on Systems,Man and Cybernetics* (SMC99), pp.610-615

[4] Miguel Nicolau and Conor Ryan,How Functional Dependency Adapts to Salience Hierarchy in the GAuGE System, in *EuroGP 2003*, Springer, pp.153-163