GPにおける交叉法の改良と多目的化

宗久研究室

T03KF018　鈴木修行

目次

- GAとGPについて
- 人工蟻の餌探索問題の概要
- 実験結果
- GPの問題点と改善方法
- 改良後の実験結果
- まとめ

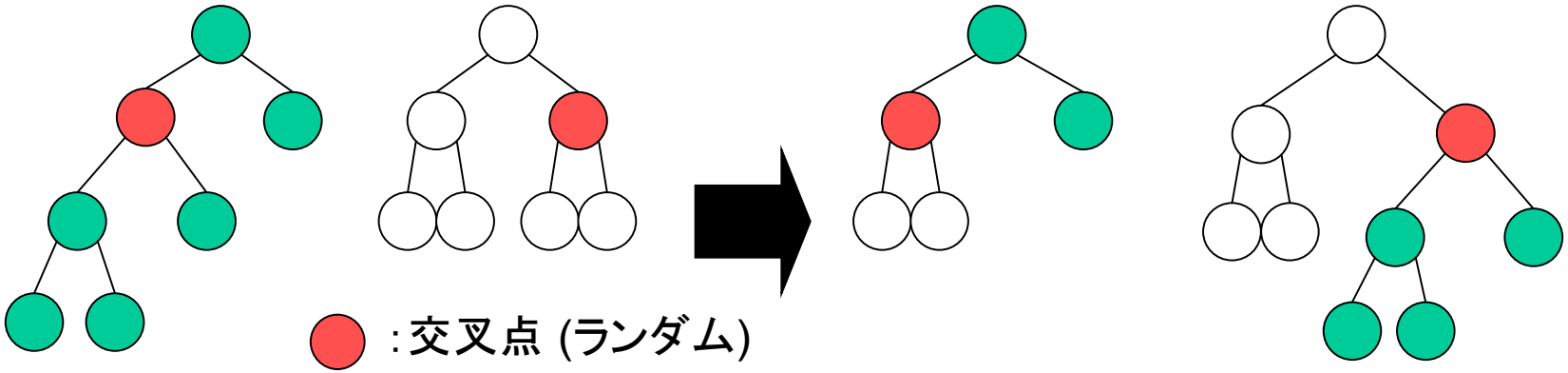
遺伝的アルゴリズムと遺伝的プログラミング

- 遺伝的アルゴリズム(Genetic Algorithms : GA)
  - 生物の淘汰, 交配などの遺伝のメカニズムに似た操作を取り入れた最適解の探索アルゴリズム.
  - 現在の集団に世代交代(選択,交叉,突然変異)が行われ，次の世代の集団を生成し,これを繰り返すことにより最適解を探索する.

- 遺伝的プログラミング(Genetic Programming : GP)
  - グラフ構造(とくに木構造)を扱えるように遺伝的アルゴリズムの手法を拡張.
  - 遺伝的アルゴリズムでは表現できなかった数式やプログラムのコードなどを表現することができ, 進化させることが可能となる.

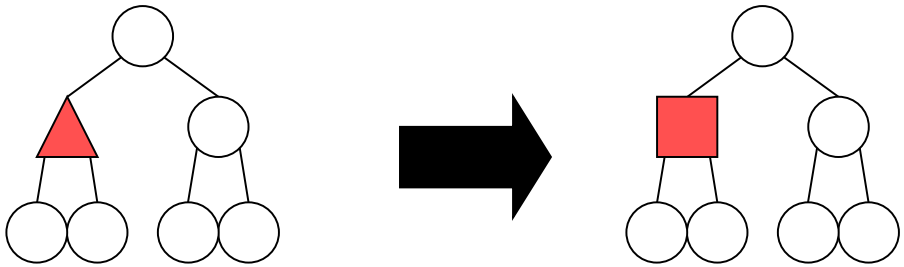
木構造に対するGPオペレータ

- 選択－適合度比例方式などでの個体(木)の選出
- 交叉－部分木の交換



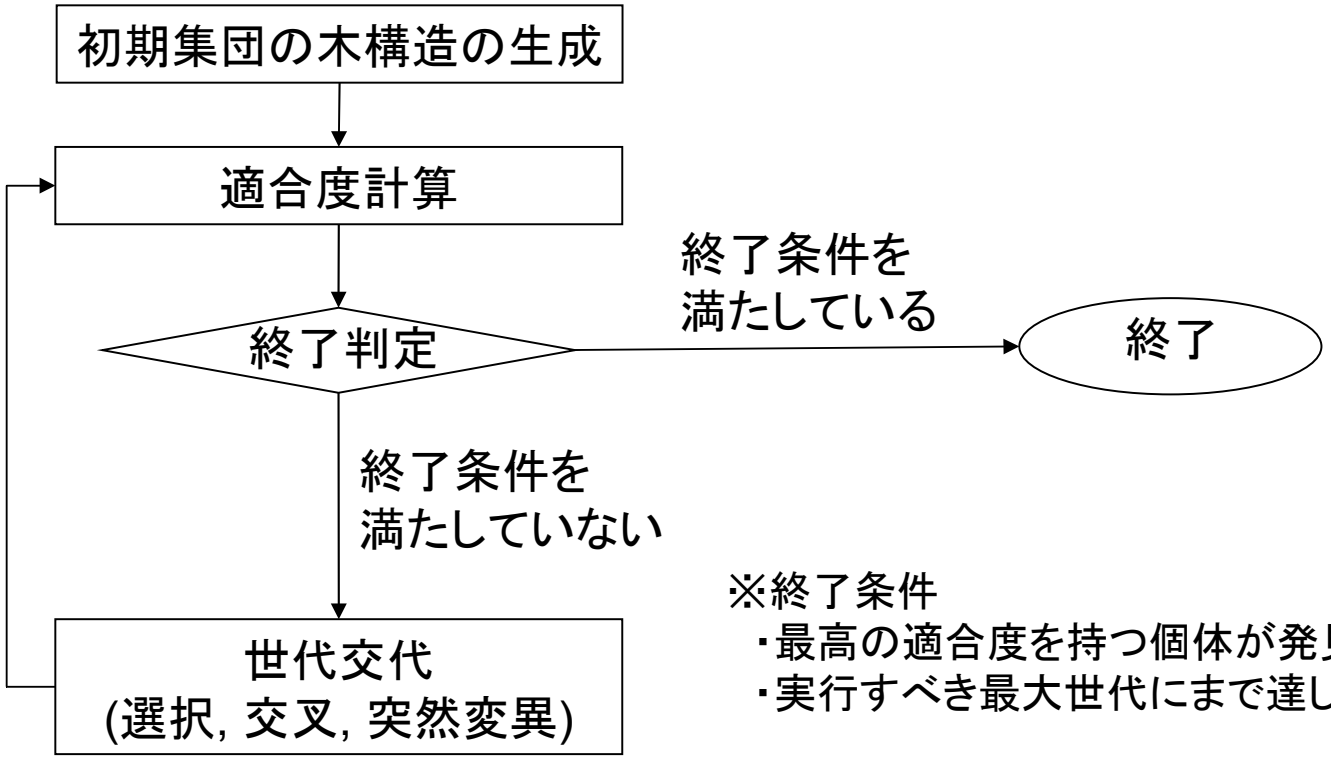
：交叉点 (ランダム)

- 突然変異－ノードの変化



：突然変異点(ランダム)

GPの処理の流れ



※終了条件
- 最高の適合度を持つ個体が発見された
- 実行すべき最大世代にまで達した

人工蟻の餌探索問題

- 人工蟻(Ant)は与えられた空間を自由に動き回る人工生命である.
  人工蟻が空間に配置されている餌を, エネルギーを有効に使いながら探索する.

- 人工蟻の動作
  - 向きを変えながら前進する.
  - 動作(方向転換,前進)を行うたびにエネルギーを 1消費する.
  - センサを持ち, 前方に餌があるかどうかわかる.

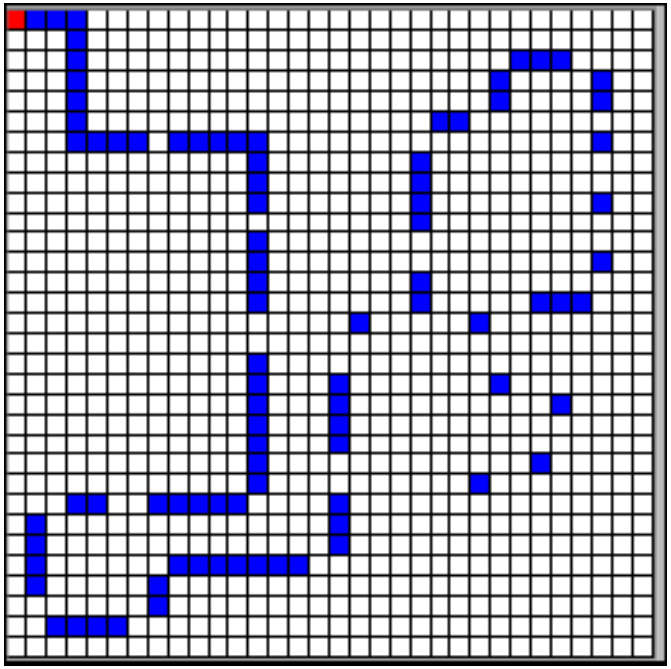
- 以下に挙げる終端記号・非終端記号をノードとして,人工蟻の行動を制御するルール木を生成する.
  このルール木の生成を効率的,かつ,より良いルール木を生成するために遺伝的プログラミングを適用する.

使用する記号と適合度

- 終端記号
  - TurnRight() ・・・右に90度向きを変える
  - TurnLeft() ・・・左に90度向きを変える
  - Forward() ・・・前に一歩前進する
- 非終端記号
  - If_Food_Ahead() ・・・引数を2つもち, 目の前に餌がある時, 第1引数を, ない時は第2引数を実行
  - Prog2() ・・・引数を2つもち, 第1引数,第2引数を順に実行
  - Prog3() ・・・引数を3つもち, 第1引数から順に実行
- 適合度の評価
  - 次ページに挙げる訓練例において人工蟻をルール木に従って動かした時に, エネルギーを使い果たした時点で人工蟻が獲得した餌の数を適合度とする. (最高の適合度は89)

実験

訓練例



Antの実行環境：32×32のマス目

- 餌が配置されている場所
- Antの初期位置

- 餌探索問題のパラメータ
  - Antの初期エネルギー　: 400
  - 配置される餌の数　: 89
- GPのパラメータ
  - 集団数　: 300
  - 最大世代　: 200
  - 交叉率　: 0.8
  - 突然変異率　: 0.05
  - 選択法　: 適合度比例方式
  - 最大の木の深さ　: 20に制限

実験

- エリート戦略

  世代交代をする時に現在の世代の最良個体を次世代に残す.

  ー利点：適合度の最大値は単調増加する.

  ー欠点：集団内に似た個体が増え局所解に陥りやすい.

- 局所解からの脱出

  15世代の間,最良適合度に変化がない場合には局所解に陥ったと判断し,集団数/10個のランダムな個体を現在の集団と入れ替える

- 前記のパラメータ, エリート戦略,局所解からの脱出を取り入れたGPを今回の実験の基本モデルとし,以後の改良を行う.

実験結果(基本モデル)

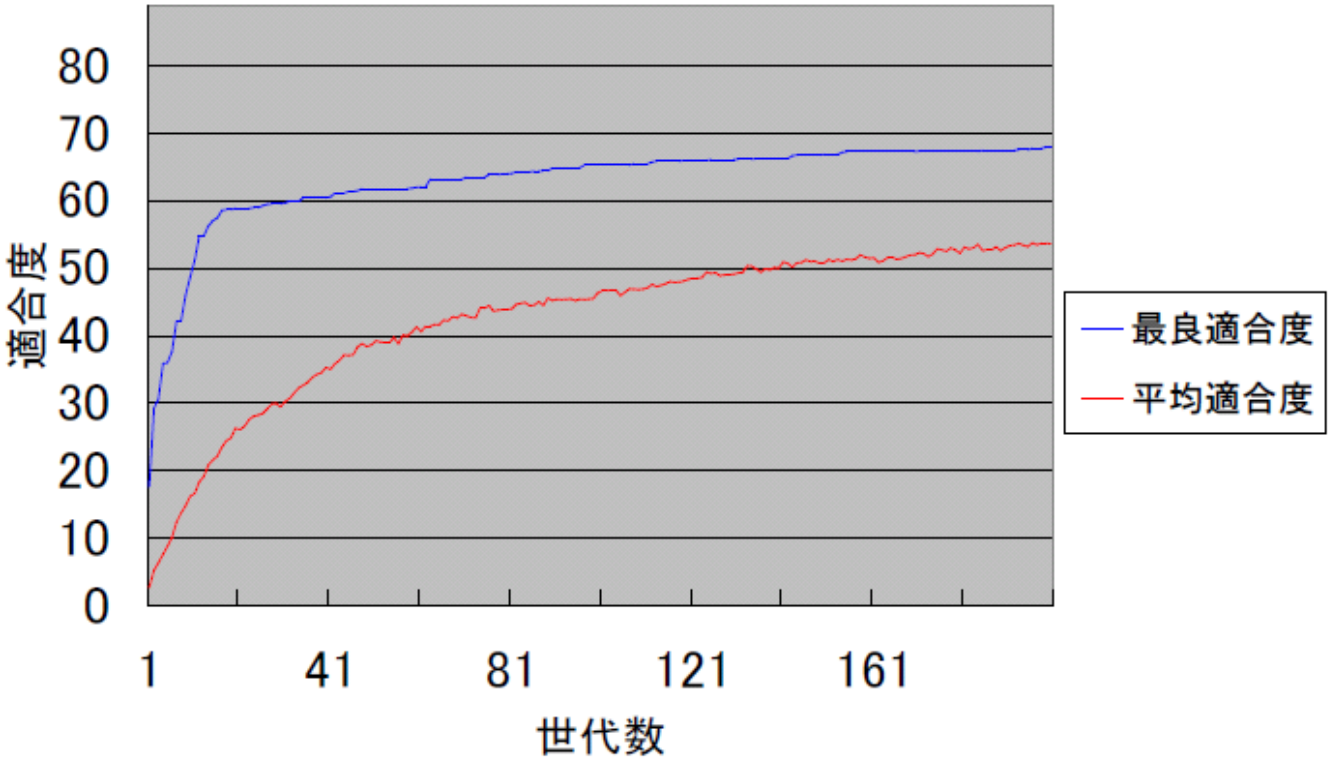
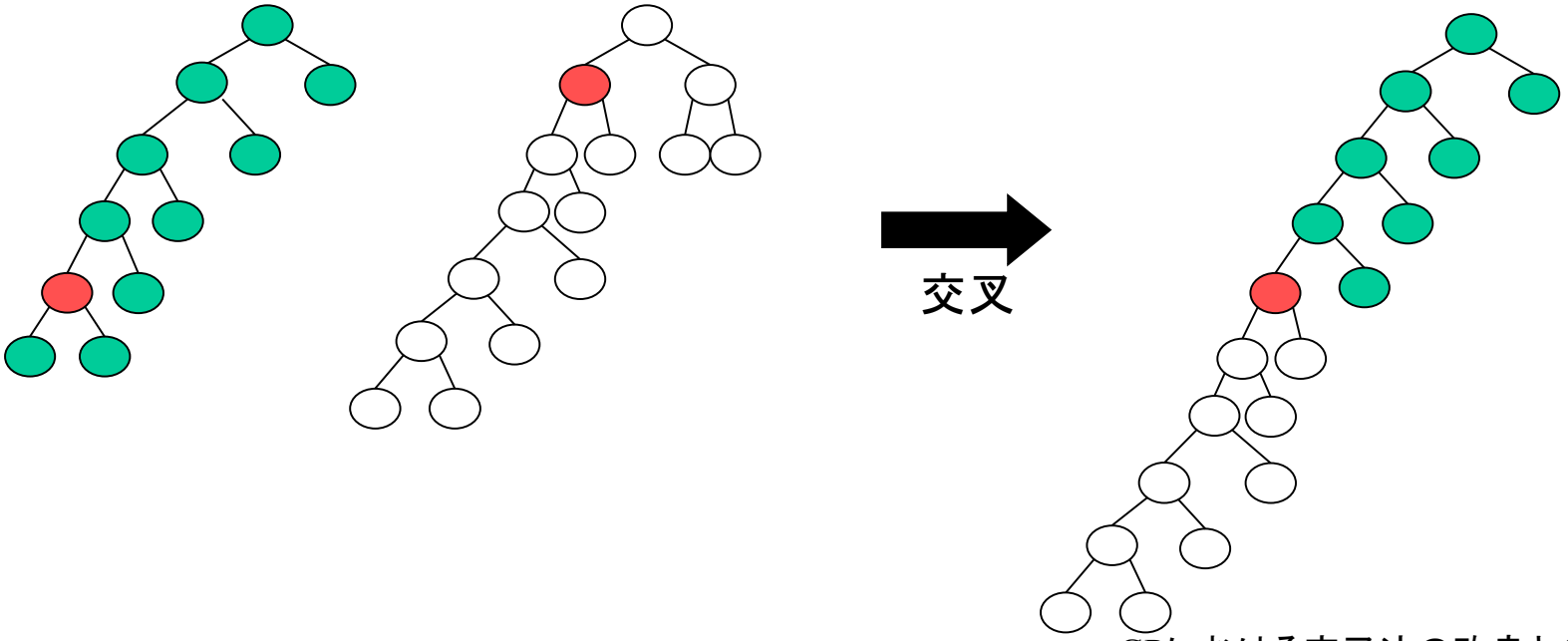


図1．最良適合度と平均適合度の推移(100回の平均)

GPにおける問題点(ブロート)

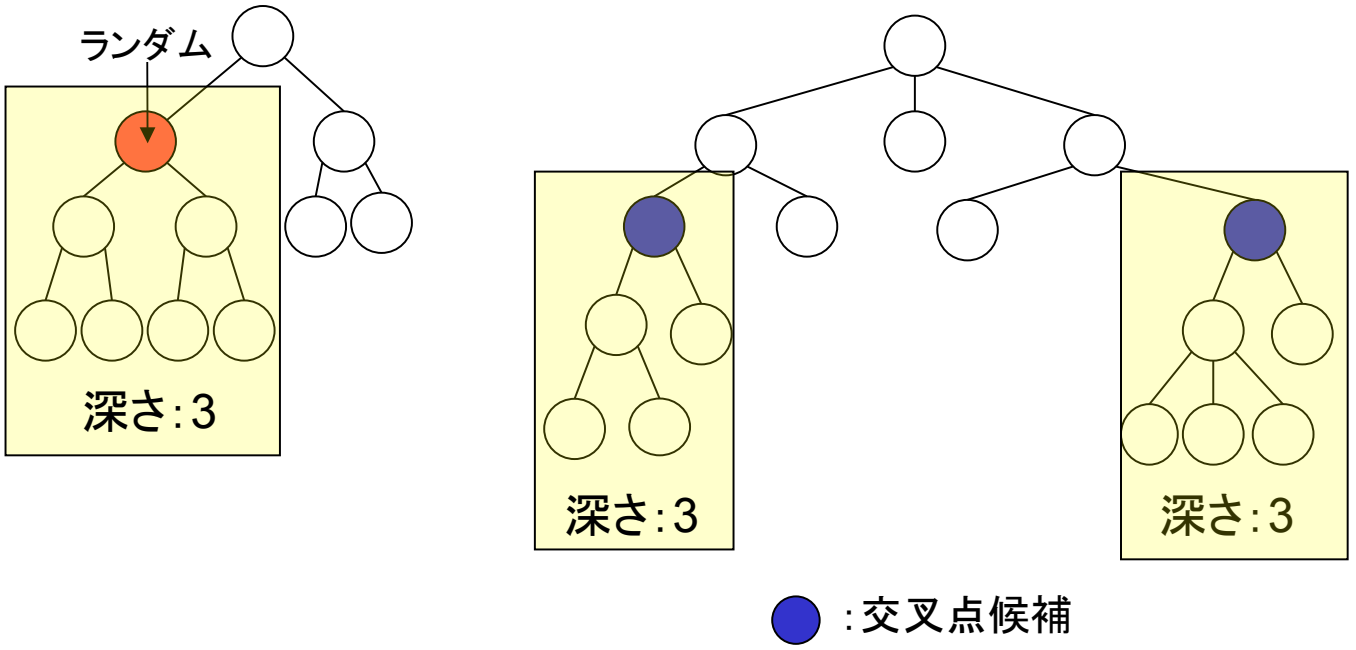
- ブロートとは遺伝的プログラミングの解の探索過程でプログラムの長さ(ノード数)が増大すること.
- これにより遺伝的プログラミングの効率的な探索が阻害される.

- ブロートの起きる原因は主に交叉にある.



ブロートの改善方法

– サイズ平等交叉
  - 木の深さが同じ部分木対を交換する.



サイズ平等交叉の実現方法

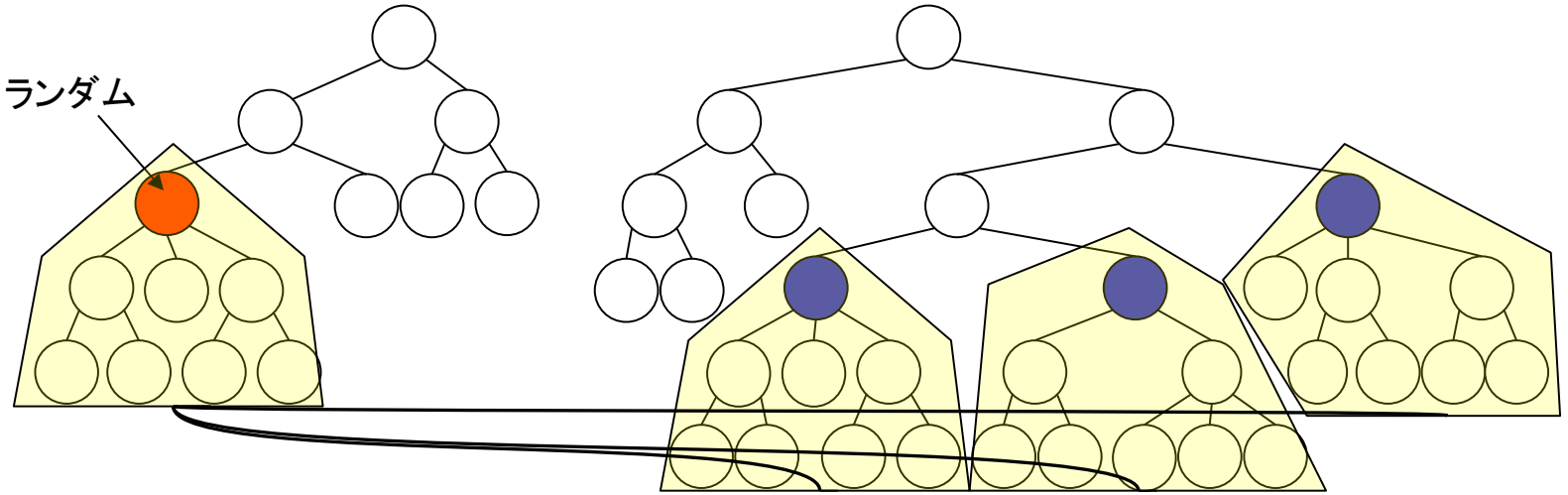
1. 基本モデルの交叉と同様に両方の親の木の交叉点をランダムに選び,部分木を取り出す.

2. 部分木の深さが違ったら,親の木の深さが浅い方の交叉点は決定し, 親の木の深さが深い方の木の交叉点を再度ランダムに選択する.

3. 深さが同じ木が見つかるまで繰り返す.

ブロートの改善方法

- 相同性交叉
  - 交叉させる部分木の形が同じもしくは「似ている」, 相同性を持った部分木対同士を交換する.



相同性交叉の実現方法

- 相同性の定義
  - この制限を強くしすぎる(形が全く同じやノードの種類まで同じなど)と局所解に陥りやすいや交叉の効果が薄れるなどの副作用が考えられる.

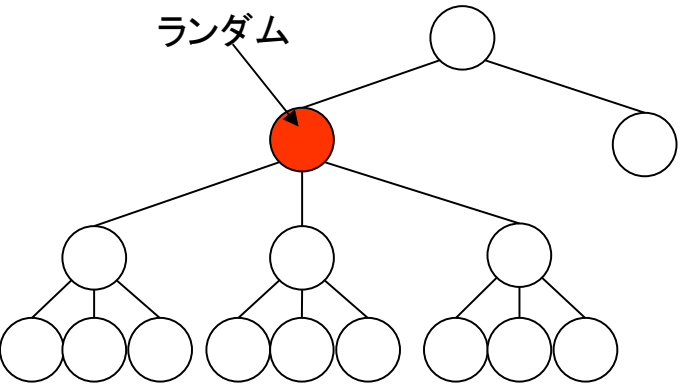
- 部分木の深さが同じで, かつ, 終端ノードの数が近い時に部分木は相同性を持っているとした.

部分木の深さ	終端ノード数の差の許容範囲
2	1
3	2
4	3
5	8
6以上	10

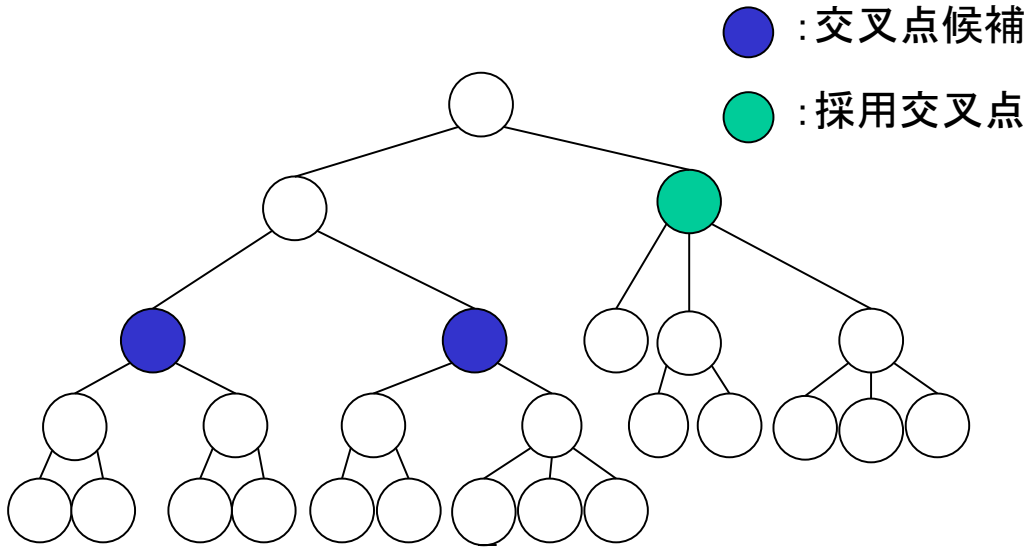
相同性交叉の実現方法

1. サイズ平等交叉の時と同じ要領で木の深さが同じ部分木を取り出す.
2. 終端ノード数の差が許容範囲外だったら, 親の木の深さによってどちらか一方の木の交叉点を決定し, もう一方の木の交叉点を再度ランダムに選択する.
3. 深さが同じで, 終端ノード数の差が許容範囲内である木が見つかるまで繰り返す.
4. 10回繰り返してもノード数の差が許容範囲内である木が見つからない場合は終端ノード数の差が一番小さかった交叉点を採用する.(深さは同じ)

交叉点



終端ノード数 ： 9



終端ノード数：4　終端ノード数：5

終端ノード数：6

ブロートの改善方法

- もうひとつの方法として適合度の計算を餌の数だけでなく，木のサイズも適合度に反映させる「多目的化」をする.
  - 適合度の評価

    適合度 = 獲得した餌の数 － α×Size

    Size : 木のノード数

    α　: 木のサイズの影響を決める定数

    今回は良い結果が得られるように,試行錯誤してα＝0.05としたが,これが最適であるとは言い切れない.

改良後の実験結果

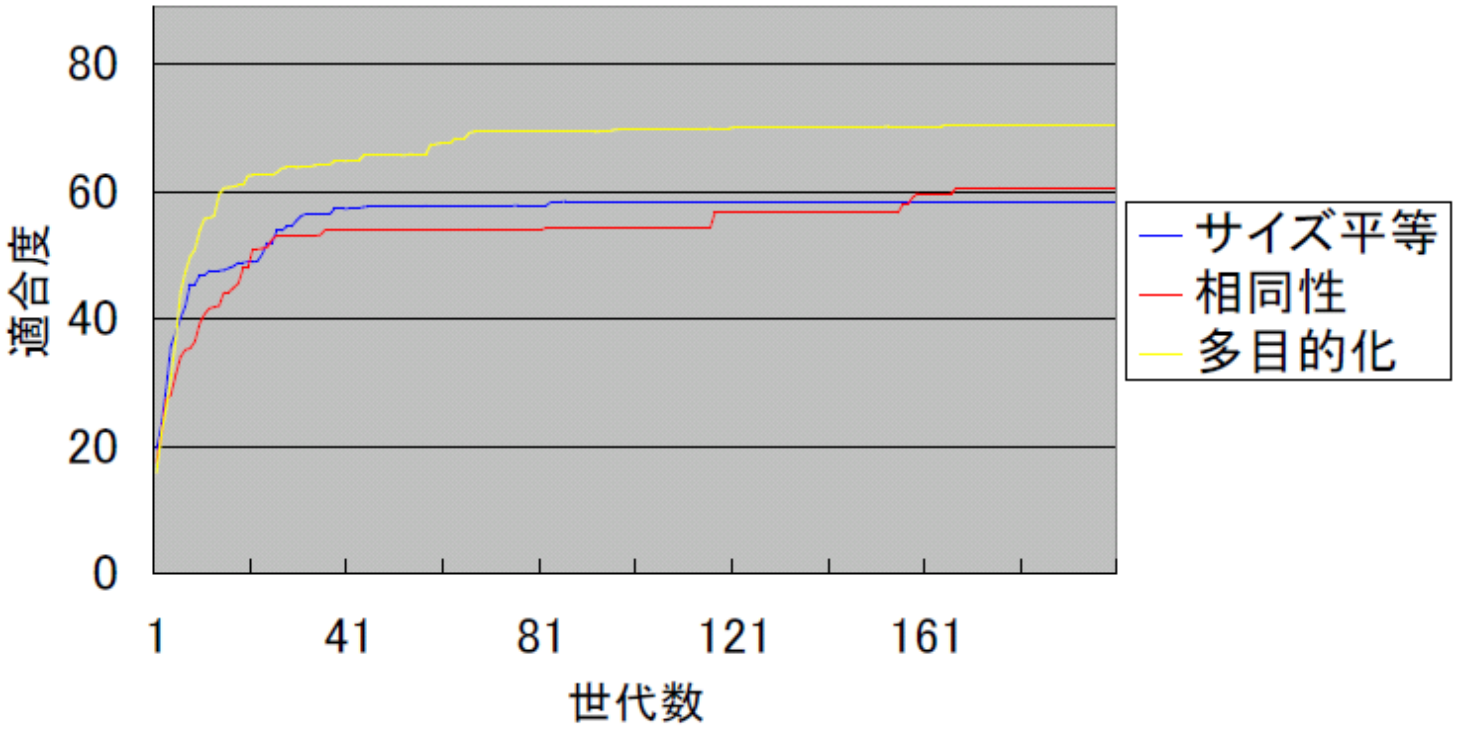


図2.各手法による最良適合度の推移(**実行50回の平均**)

改良後の実験結果

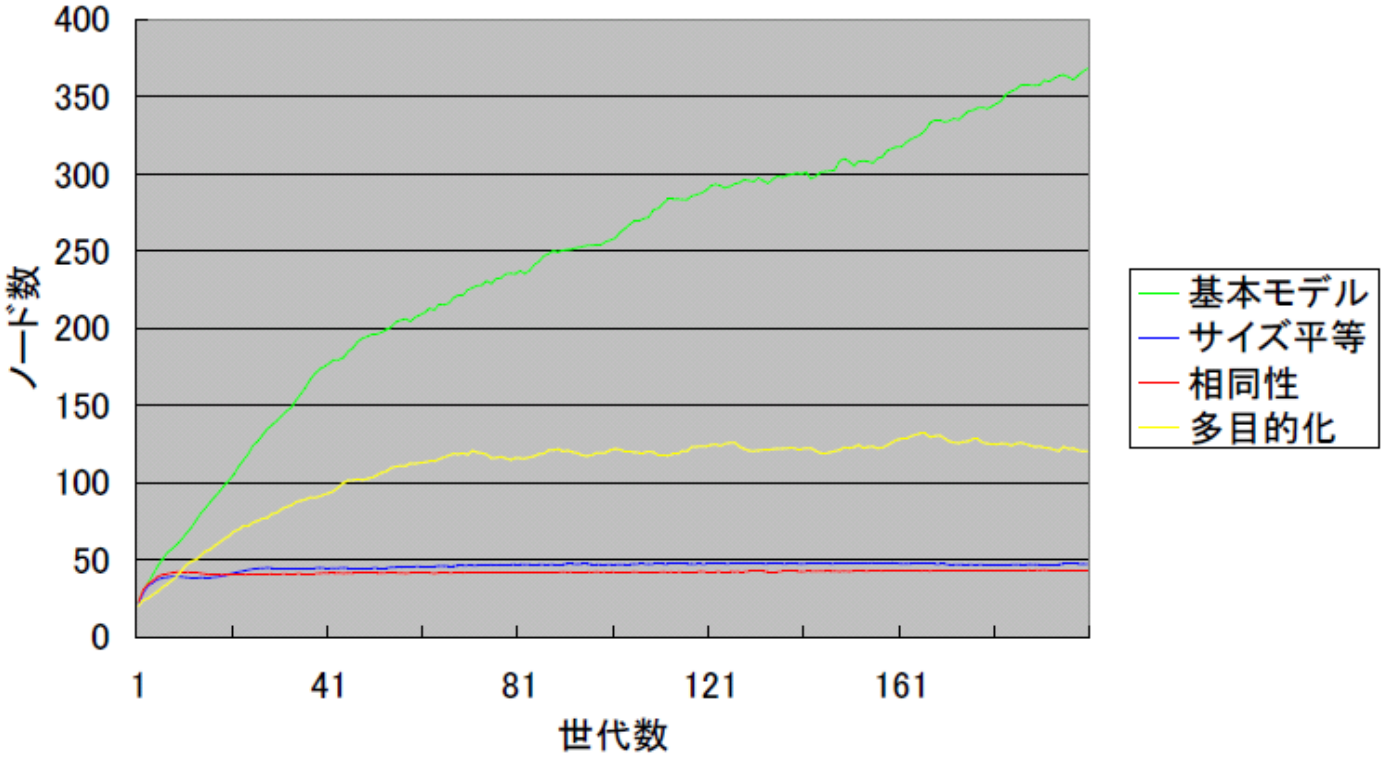


図3.各手法によるノード数の推移(**実行50回の平均**)

改良後の実験結果

最適解の発見回数(100回中)と平均実行時間

	最適解の発見回数	実行時間
基本モデル	11回	18.2秒
サイズ平等交叉	7回	7.2秒
相同性交叉	6回	8.1秒
多目的化	12回	8.9秒

- ノード数, 実行時間
  - どの手法においてもノード数の増大を抑え, 実行時間も基本モデルと比べて半分程度に抑えることができた
- 最適解の探索能力
  - 多目的化は基本モデルと同等程度の能力を示しているが, 他の手法では劣る

まとめ

- GPの探索能力が交叉によるところが大きい.
  - 交叉時に制限を加えた多目的化以外の手法で探索能力が落ちてしまっている.
- 今回の手法の中では多目的化が一番有用であった.
  - 基本モデルと同等程度の探索能力を示して, 実行時間も短縮.
- 多目的化以外の改良したプログラムでの探索能力の低下は, 通常の交叉と交互に行ったり, ノードが異常に増える場合にだけ制限を加えるなど,より工夫を加えることで避けることができるかもしれない.
- ノード数を抑えつつ,探索能力も向上させることは難しい.

参考文献

- 伊庭斉志:「遺伝的プログラミング入門」(2001).
- 鶴田寿男: 遺伝的プログラミング人工蟻の探索, 山梨大学工学部コンピュータメディア工学科卒業論文(2005).
- 辻康孝: 進化的計算手法による多目的最適化(online), <http://zeus.mech.kyushu-u.ac.jp/~tsuji/download/OR_kyushu.pdf>, (2006-12-10).