数理解析研究所講究録
第1857巻 2013年 1-18

巣形成と役割分担を用いた最適化手法の改良

広島修道大学商学部　阪井　節子 (Setsuko Sakai)
Faculty of Commercial Sciences, Hiroshima Shudo University
広島市立大学大学院 情報科学研究科　高濱　徹行 (Tetsuyuki Takahama)
Graduate School of Information Sciences, Hiroshima City University

1　はじめに

最適化問題，特に，非線形最適化問題は実世界にしばしば出現する重要な問題である．近年，これらの最適化問題に対する最適化アルゴリズムとして，進化的アルゴリズム (Evolutionary Algorithm, EA) が注目されており，多くの研究が行われている．EAは，生物進化の過程をモデル化した最適化アルゴリズムの総称であり，遺伝的アルゴリズム (Genetic Algorithm, GA)，進化戦略 (Evolution Strategy, ES)，差分進化 (Differential Evolution, DE)[1, 2] など多くのアルゴリズムが提案されている．

EAでは，集団の各要素が個別に最適化されるため，集団が急速に特定の解に集中することが少なく，その結果，比較的局所解に陥りにくいという特徴があるが，稜構造を持つ関数や多峰性関数など最適化が困難な問題では，探索点が局所解に収束したり，探索点の多様性が失われて移動が停止することにより，最適解の探索に失敗することもある．これに対処するためには，探索点の多様性を維持しながら大域探索を行う必要がある．一方，EAは確率的な多点探索を行うため，関数評価回数が多くなりがちである．近年，最適化問題が大規模化し，目的関数の評価コストが増大してきている．このため，目的関数の評価回数の削減は大きな課題となってきている [3, 4, 5]．これに対処するためには，良好な探索点の周辺を集中して探索する近傍探索を重視することが有効であると考えられる．このように，稜構造関数や多峰性関数などにも頑健な探索を効率的に行うためには，大域探索と近傍探索のバランスを適切に取る必要がある．

大域探索と近傍探索のバランスを実現する方法として，探索点集合からグラフを構成し，探索点の周辺での目的関数の概形を把握し，その状態に基づいて探索点毎にアルゴズムパラメータを調整する研究が行われている [6, 7, 8, 9]．

文献 [6] では，近接グラフ (proximity graph) を用いて各探索点の山谷判定を行っている．山谷判定では，探索点を谷点 (隣接するすべての点より良い点)，山点 (隣接するすべての点より悪い点)，谷近傍点 (山点でない谷点の隣接点)，山近傍点 (谷点でない山点の隣接点)，その他の点の5種類に分類し，それぞれに対してDEのアルゴリズムパラメータ F, CR の値を設定し，大域探索と近傍探索の切り換えを行う最適化手法NGDE(Neighborhood Graph Based DE) を提案し，標準的DEに比べて少ない評価回数で効率的に解探索が実現できることを示した．

文献 [7] では，NGDEと同じく探索点集合から近接グラフを構成し，探索点の山谷判定を行うが，探索点毎に探索効率をさらに向上させるために，蟻の巣形成と役割分担に着目した最適化アルゴリズムNRDE(DE using Nest building and Role sharing) を提案した．NRDEでは，探索点を女王蟻，雄蟻，働き蟻に分類する．女王蟻は谷点であり，この谷点と隣接する谷近傍点がその女王蟻に対する雄蟻である．女王蟻とそれに対する雄蟻が存在する領域は有望な探索領域である考えられるためこれを巣と見なし，女王蟻と雄蟻は巣の周辺の近傍探索を行う．それ以外の点は働き蟻とし，働き蟻はさらに探索蟻と帰巣蟻の2種類に分ける．帰巣蟻は，働き蟻の内で山点である蟻で，餌場の探索に失敗したと考え，帰巣本能により集団内で最良の巣に急ぎ帰る行動をとる，すなわち中域探索を行う．探索蟻は，それ以外の採餌行動をする働き蟻で，大域探索を行う．NRDEでは，役割毎にアルゴリズムパラメータの値を設定することに加えて，基本ベクトルの

選択戦略を変更した．これにより，NGDE よりも更に標準的 DE に比べて少ない評価回数で効率的に解探索が実現できることを示した．

文献 [7] では，近接グラフとして Gabriel グラフ (Gabriel Graph, GG) と相対近傍グラフ (Relative Neighborhood Graph, RNG) を用いた．その結果，Gabriel グラフを用いた NRDE(NRDE/GG) の場合，滑らかな関数では評価回数を大きく削減できたが，急峻な構造を持つ関数や多峰性関数では探索に失敗することが多くなった．一方，相対近傍グラフを用いた NRDE(NRDE/RNG) の場合，稜構造を持つ関数や多峰性関数で安定的に評価回数を削減できたが，滑らかな関数では NRDE/GG による評価回数の削減には及ばなかった．この結果の理由としては，GG と RNG の辺の数すなわちグラフの疎密の程度が関係していると考えられる．RNG は GG の部分グラフであり，RNG は GG に比べて辺の数が少ない，疎なグラフである．一方，GG は RNG に比べて辺の数が多い，密なグラフである．密なグラフの場合，滑らかな単峰性関数では目的関数の形状の近似精度が高くなり効率的な探索が行われが，稜構造を持つ関数や多峰性関数では局所解に陥り易く探索性能が低下する．一方，疎なグラフの場合，滑らかな単峰性関数では密なグラフに比べて最適解への収束が遅いが，稜構造を持つ関数や多峰性関数でも局所解に陥りにくく安定的な探索性能を持つ．

したがって，安定的な探索性能を実現するには近接グラフの辺の数，すなわち疎密の程度を制御する必要がある．しかし，GG や RNG は，与えられた点集合に対して辺が固定的に決定される近接グラフであり，辺の数を制御することができない．そこで，本研究では，新たに Delaunay グラフの近似グラフを用いた NRDE を提案する．Delaunay グラフの近似グラフは探索点集合に競合ヘブ則を適用することにより生成することができる．競合ヘブ則は，生成する近接グラフの辺の数を制御できる．本研究では，競合ヘブ則を用いて生成した近接グラフを NRDE に適用し，様々な形状の目的関数に対して，少ない関数評価回数で精度の高い解の探索を安定的に実現できるように NRDE を改良することを目指す．

本論文は次のような構成である．2. で最適化問題を定義し，3. で近接グラフとその生成方法について，4. で本研究のアルゴリズムを提案し，5. で実験結果を示す．6. はまとめである．

2 最適化問題

本研究では，決定変数の上下限制約のみを有する次のような最適化問題 (P) を扱う．

$$(P)\ \text{minimize} \quad f(\boldsymbol{x}) \qquad (1)$$
$$\text{subject to}\ \ l_i \le x_i \le u_i,\ i = 1, \ldots, n$$

ここで，$\boldsymbol{x} = (x_1, \cdots, x_n)$ は n 次元決定変数ベクトル，$f(\boldsymbol{x})$ は目的関数であり，f は線形あるいは非線形の実数値関数である．l_i, u_i はそれぞれ，n 個の決定変数 x_i の下限値，上限値である．さらに，以下では全ての上下限制約を満足する領域を探索空間 S と呼ぶことにする．

3 近接グラフと競合ヘブ則

3.1 近接グラフ

頂点集合 V と辺集合 E からなるグラフ G を $G(V, E)$ で表現する．近接グラフ (proximity graph) は，頂点集合 V の近傍構造を表現するグラフであり，最近傍グラフ (Nearest Neighborhood Graph)，Gabriel グラフ (Gabriel Graph, GG)[10]，相対近傍グラフ (Relative Neighborhood Graph, RNG)[11]，β skeleton[12],

競合ヘブ則によるグラフ [13] などが提案されている．近接グラフでは，任意の 2 頂点 $\boldsymbol{v}_i, \boldsymbol{v}_j \in V$ が近傍条件を満足するときのみ辺 $(\boldsymbol{v}_i, \boldsymbol{v}_j) \in E$ となる．

Gabriel グラフでは，$\boldsymbol{v}_i$ と $\boldsymbol{v}_j$ を結ぶ線分を直径とする超球内に他の頂点が含まれていなければ，$\boldsymbol{v}_i$ と $\boldsymbol{v}_j$ が近傍条件を満足する．Gabriel グラフ $GG(V,E)$ は，以下のように定義できる．

$$(\boldsymbol{v}_i, \boldsymbol{v}_j) \in E \iff HS\left(\frac{\boldsymbol{v}_i + \boldsymbol{v}_j}{2}, \frac{||\boldsymbol{v}_i - \boldsymbol{v}_j||}{2}\right) \cap V = \phi \tag{2}$$

ここで，$HS(\boldsymbol{v}, r)$ は頂点 $\boldsymbol{v}$ を中心とする半径 r の超球内の領域，すなわち

$$HS(\boldsymbol{v}, r) = \{\boldsymbol{x} \,|\, ||\boldsymbol{x} - \boldsymbol{v}|| < r\} \tag{3}$$

である．

この様子を図 1 に示す．任意の頂点 $\boldsymbol{v}_k$ が超球内に存在しない場合にのみ頂点を接続し，そうでない場合は接続しない．

相対近傍グラフでは，頂点 $\boldsymbol{v}_i$ および $\boldsymbol{v}_j$ を中心とする半径 $||\boldsymbol{v}_i - \boldsymbol{v}_j||$ の 2 つの超球の共通集合内に他の頂点が含まれていなければ近傍条件を満足する．相対近傍グラフは Gabriel グラフの部分グラフである．相対近傍グラフ $RNG(V,E)$ は以下のように定義できる．

$$(\boldsymbol{v}_i, \boldsymbol{v}_j) \in E \iff HS(\boldsymbol{v}_i, ||\boldsymbol{v}_i - \boldsymbol{v}_j||) \cap HS(\boldsymbol{v}_j, ||\boldsymbol{v}_i - \boldsymbol{v}_j||) \cap V = \phi \tag{4}$$

この様子を図 2 に示す．

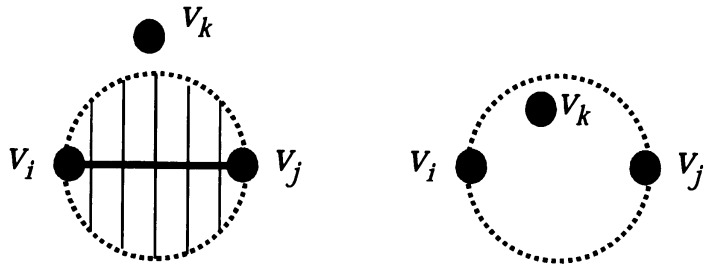

図 1: Gabriel グラフにおける辺の判定

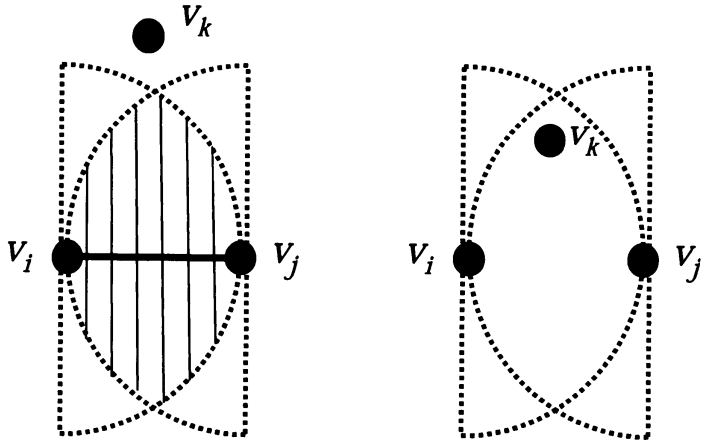

図 2: 相対近傍グラフにおける辺の判定

3.2 Delaunay グラフ

距離空間 S 内に N 個の頂点 $V = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$ が存在するとき，点集合 V 内のどの点に最も近いかによって，空間 S を N 個の領域に分割することができる．この分割図形が空間 S に対する Voronoi 図 (Voronoi diagram) である．各点 $\boldsymbol{x}_i$ を母点 (generator)，対応する領域 $R(\boldsymbol{x}_i)$ を Voronoi 領域 (region) と呼ぶ．Voronoi 領域は，距離関数 d を用いて以下のように定義できる．

$$R(\boldsymbol{x}_i) = \{\boldsymbol{x} \in S \,|\, d(\boldsymbol{x}, \boldsymbol{x}_i) \leq d(\boldsymbol{x}, \boldsymbol{x}_j), \forall \boldsymbol{x}_j \in V \setminus \{\boldsymbol{x}_i\}\} \tag{5}$$

Voronoi 領域に隣接関係が存在するとき，隣接する領域の母点間を結合したグラフが Delaunay グラフ (Delaunay diagram) である．Delaunay グラフは Voronoi 図の双対である．図 3 に Voronoi 図と Delaunay グラフの例を示す．点線が Voronoi 図，実線が Delaunay グラフである．Delaunay グラフも近接グラフの一つであり，最近傍グラフ，相対近傍グラフ，Gabriel グラフは Delaunay グラフの部分グラフである．

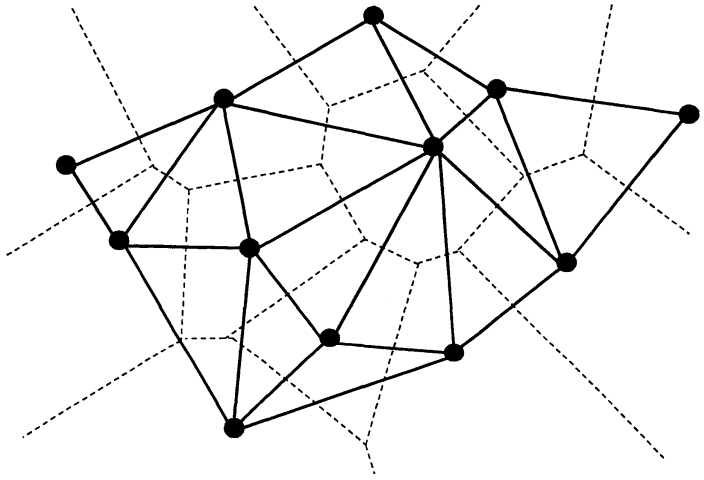

図 3: Voronoi 図 (点線) と Delaunay グラフ (実線)

3.3 競合ヘブ則と Delaunay グラフ

Martinetz ら [13] は，位相関係を表現するニューラルネットワークを構成するために，競合ヘブ則により Delaunay グラフの部分グラフを生成する以下の方法を提案した．

1. 全頂点 $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ 間の結合強度 C_{ij} とし，$C_{ij} = 0$ とする．
2. 空間 S における確率分布 $P(\boldsymbol{x})$ に従い，入力パターン $\boldsymbol{x}^p$ を生成する．
3. $\boldsymbol{x}^p$ に最も近い頂点 $\boldsymbol{x}_{i1}$ と 2 番目に近い頂点 $\boldsymbol{x}_{i2}$ を以下のように決める．

$$\begin{aligned} d(\boldsymbol{x}^p, \boldsymbol{x}_{i1}) &\leq d(\boldsymbol{x}^p, \boldsymbol{x}_j), \forall \boldsymbol{x}_j \in V \\ d(\boldsymbol{x}^p, \boldsymbol{x}_{i2}) &\leq d(\boldsymbol{x}^p, \boldsymbol{x}_j), \forall \boldsymbol{x}_j \in V \setminus \{\boldsymbol{x}_{i1}\} \end{aligned} \tag{6}$$

4. 競合ヘブ則に従い，$\boldsymbol{x}_{i1}$ と $\boldsymbol{x}_{i2}$ の結合強度を強化する．すなわち，$C_{i1,i2}$ が 0 ならば，$C_{i1,i2} > 0$ とし，$\boldsymbol{x}_{i1}$ と $\boldsymbol{x}_{i2}$ を結合する．なお，接続関係のみが必要な場合には $C_{i1,i2} = 1$ とすれば良い．

5. 終了条件を満足するまで，2. に戻る．

競合ヘブ則のアルゴリズムによって生成されたグラフは，Delaunay グラフの部分グラフである．同じく Delaunay グラフの部分グラフである Gabriel グラフや相対近傍グラフでは，与えられた頂点集合 V に対して一意に決定されるグラフであり，グラフの疎密の程度を制御することはできない．一方，競合ヘブ則を用いると入力パターンの個数 M によって，生成するグラフの辺の数，すなわちグラフの疎密の度合を制御することができる．一般に，競合ヘブ則によるグラフは M が増加すれば密になり，$M \to \infty$ のとき Delaunay グラフに収束する．

本研究では，頂点集合 V からランダムに 2 点を M 組選択し，選択した 2 点の中点を入力パターンとする．M の値は $2N$ とする．ここで，N は頂点の総数である．

4 巣形成と役割分担に基づく最適化アルゴリズム

本研究では，Differential Evolution (DE) に，競合ヘブ則によって生成した近接グラフを用いた巣形成と役割分担の考えを導入した最適化アルゴリズム NRDE(Differential Evolution with Nest-building and Role-sharing) を提案する．

4.1 Differential Evolution

Differential evolution (DE) は，Storn and Price[1, 2] によって提案された進化的アルゴリズム (Evolutionary Algorithm) の 1 つである．DE は解集団を用いた多点探索を行う確率的直接探索法であり，非線形問題，微分不可能な問題，非凸問題，多峰性問題など，様々な最適化問題に適用されてきており，高速で頑健なアルゴリズムであることが示されている．

DE では，探索空間中にランダムに初期個体を生成し，初期集団を構成する．各個体は，決定ベクトルに対応した n 個の決定変数を遺伝子として持つ．各世代において，全ての個体を親として選択する．各親に対して，次のような処理が行われる．突然変異のために，選択された親を除く個体群から互いに異なる $1+2\,num$ 個の個体を選択する．最初の個体が基本ベクトル (base vector) となり，残りの個体対が num 個の差分ベクトル (difference vector) となる．差分ベクトルにスケーリングファクタ F(scaling factor) が乗算され基本ベクトルに加算され，変異ベクトル (mutant vector) が得られる．変異ベクトルと親が交叉し，交叉率 CR(crossover rate) により指定された確率で親の遺伝子を変異ベクトルの要素で置換することにより，子ベクトル (trial vector) が生成される．最後に，生存者選択として，子が親よりも良ければ，親を子で置換する．

DE には幾つかの形式が提案されており，DE/best/1/bin や DE/rand/1/exp などがよく知られている．これらは，DE/*base*/*num*/*cross* という記法で表現される．"*base*" は基本ベクトルとなる個体の選択方法 (選択戦略) を指定する．rand 戦略では基本ベクトルとして集団からランダムに選択した個体を採用し，best 戦略では集団内の最良個体を採用する．例えば，DE/rand/1 では，各親個体 $\boldsymbol{x}^i$ に対して，個体 $\boldsymbol{x}^{p1}$, $\boldsymbol{x}^{p2}$, $\boldsymbol{x}^{p3}$ を $\boldsymbol{x}^i$ および互いに重ならないよう集団からランダムに選択する．変異ベクトル $\boldsymbol{x}^m$ は，基本ベクトル $\boldsymbol{x}^{p1}$ と差分ベクトル $\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}$ によって次式で与えられる．

$$\boldsymbol{x}^m = \boldsymbol{x}^{p1} + F(\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}) \tag{7}$$

ここで，Fはスケーリングファクタである．DE/best/1では，基本ベクトルとして集団内の最良個体$\boldsymbol{x}^{\text{best}}$を選択し，変異ベクトル$\boldsymbol{x}^m$を生成する．

$$\boldsymbol{x}^m = \boldsymbol{x}^{\text{best}} + F(\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}) \tag{8}$$

"*num*" は基本ベクトルを変異させるための差分ベクトルの個数を指定する．"*cross*" は子を生成するために使用する交叉方法を指定する．例えば，DE/*base*/*num*/bin は一定の確率で遺伝子を交換する二項交叉 (binomial crossover) を用い，DE/*base*/*num*/exp は，指数関数的に減少する確率で遺伝子を交換する指数交叉 (exponential crossover) を用いる．図4に，二項交叉と指数交叉を示す．新たな子個体$\boldsymbol{x}^{\text{new}}$は，親個体$\boldsymbol{x}^i$と変異ベクトル$\boldsymbol{x}^m$から生成される．ここで，$CR$は交叉率である．

```
binomial crossover DE/·/·/bin
  j_rand = randint(1,n);
  for(k=1; k ≤ n; k++) {
    if(k == j_rand || u(0,1) < CR)
      x_k^new = x_k^m ;
    else
      x_k^new = x_k^i;
  }
```

```
exponential crossover DE/·/·/exp
  k=1; j = randint(1,n);
  do {
    x_j^new=x_j^m;
    k = k+1; j = (j+1)%n;
  } while(k ≤ n && u(0,1) < CR);
  while (k ≤ n) {
    x_j^new = x_k^i;
    k = k+1; j = (j+1)%n;
  }
```

図 4: 二項交叉と指数交叉．randint(1,n) は区間 $[1,n]$ 上の整数乱数を生成する関数，$u(0,1)$ は区間 $[0,1]$ 上の一様乱数を生成する関数である．

DE のアルゴリズムは次のようになる．

Step1 集団の初期化：N 個の初期個体$\boldsymbol{x}^i$を初期探索点として生成し，初期集団$P = \{\boldsymbol{x}^i, i = 1, 2, \cdots, N\}$を構成する．すべての個体を評価する．

Step2 終了判定：終了条件を満足すれば，アルゴリズムは終了する．終了条件としては，最大の繰り返し回数や関数評価回数を用いることが多い．

Step3 DE 操作：全ての個体$\boldsymbol{x}^i$を親ベクトルとして選択する．変異ベクトル$\boldsymbol{x}^m$を，rand 戦略の場合は式 (7)，best 戦略の場合は式 (8) にしたがって生成する．図4に示された交叉方法を用いて，変異ベクトル$\boldsymbol{x}^m$を親$\boldsymbol{x}^i$と交叉し，子ベクトル$\boldsymbol{x}^{\text{new}}$を生成する．

Step4 生存者選択：子ベクトルを評価する．子ベクトル$\boldsymbol{x}^{\text{new}}$が親ベクトルよりも良ければ子ベクトルが生存者となる．そうでなければ，親ベクトルが生存者となる．全ての個体が選択されたならば，Step5に進む．そうでなければ，Step3 に戻る．

Step5 世代交代：生存者によって集団を置換する．Step2 に戻る．

図5に DE/rand/1/exp の擬似コードを示す．

```
DE/rand/1/exp()
{
// Initialize a population
 P=N individuals generated randomly in S;
 for(t=1; FE ≤ FE_max; t++) {
  for(i=1; i ≤ N; i++) {
// DE operation
     x^p1=Randomly selected from P(p1 ≠ i);
     x^p2=Randomly selected
                  from P(p2 ∉ {i,p1});
     x^p3=Randomly selected
                  from P(p3 ∉ {i,p1,p2});
     x^m=x^p1+F(x^p2 − x^p3);
     x^new=trial vector is generated from
          x^i and x^m by exponential crossover;
// Survivor selection
     if(f(x^new) ≤ f(x^i)) z^i=x^new;
     else                  z^i=x^i;
     FE=FE+1;
  }
  P={z^i, i = 1,2,···,N};
 }
}
```

図 5: DE の擬似コード．ここで，FE は関数評価回数，$FE_{\max}$ は最大評価回数である．

4.2 山谷構造の推定と山谷判定

本研究では，山と谷を判定するために，各点 $\boldsymbol{x}^i$ に対して谷度と山度を付与する．近接グラフの全ての辺について，その辺を構成する 2 点のうち，評価値の良い点の谷度を 1 増加させ，評価値の悪い点の山度を 1 増加させる．本研究では，各点の谷度と山度を用いて，3 種類の点，谷点，山点，谷近傍点を定義し，各点 $\boldsymbol{x}^i$ の山谷判定を行う．

谷点 近傍により悪い点が一つ以上存在し，より良い点が存在しない点である．すなわち，山度が 0 かつ谷度が 1 以上の点である．

山点 近傍により良い点が一つ以上存在し，より悪い点が存在しない点である．すなわち，谷度が 0 かつ山度が 1 以上の点である．

谷近傍点 谷点に隣接する点，すなわち 1 つの辺を共有する点でかつ，山点でない点である．

4.3 NRDE/CHR のアルゴリズム

以下にアルゴリズムの概要を示す．

1. 初期化：探索領域内に点をランダムに生成し，初期探索点集合 $P = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \cdots, \boldsymbol{x}^N\}$ を構成する．
2. 近接グラフの構成：競合ヘブ則により C_{ij} を強化し，Delaunay グラフの部分グラフを作成する．
3. 山谷判定：各点に山度と谷度を付与し，山谷判定を行う．
4. 探索点の生成：女王蟻 (谷点) とそれに対する雄蟻 (谷近傍点) は，女王蟻を中心に巣の周辺を探索する局所探索を行うため，局所的な DE 操作を行う．すなわち，スケーリングファクタ F に小さな値，交叉率 CR に大きな値を設定する．帰巣蟻は，標準的な探索をあきらめ，集団内の最良の女王蟻へ向かって移動する．すなわち帰巣行動（中域探索）を行う．このために，F に大きな値，CR は [0,1] の乱数で値を設定する．探索蟻については，大域探索を行う．すなわち CR は標準的 DE で用いられ値を用いるが，F はある程度大きな値を設定する．
5. 生存者選択：生成された探索点が元の点より良好ならば，元の点と置換する．2. に戻る．

4.4 探索点の生成

本研究では，山谷判定による各親ベクトル $\boldsymbol{x}^i$ の属性に応じて，以下のように基本ベクトルの選択戦略およびスケーリングファクター F と交叉率 CR の制御ルールを採用することにより，近傍探索，中域探索と大域探索を実現する．

谷点 (女王蟻) の場合: $\boldsymbol{x}^i$ の周辺で局所探索を行うため，基本ベクトルを $\boldsymbol{x}^i$ とし，$F = 0.3$，$CR = 1.0$ とする．

谷近傍点 (雄蟻) の場合: 隣接谷点 (女王蟻) の周辺で局所探索を行う．隣接谷点方向へ探索を進めるために，基本ベクトルを近傍の隣接谷点と $\boldsymbol{x}^i$ の中点とし，$F = 0.4$，$CR = 1 - \frac{1}{n}$ とする．

山点 (帰巣蟻) の場合: 帰巣蟻は標準的な探索をあきらめ，谷点 (女王蟻) へ向かう帰巣行動をとり中域探索を行う．そのため，基本ベクトルを集団内の最良谷点 (最良の女王蟻) とし，$F = 0.9$，CR は区間 $[0, 1]$ 上の一様乱数とする．

その他の点 (探索蟻) の場合: ある程度大きく移動して大域探索を行う．基本ベクトルは $\boldsymbol{x}^i$ を除く集団からランダムに選択し，CR は通常の標準的 DE で用いられる 0.9 とする．F は大域探索のために通常の DE で標準的に用いられる F=0.7 よりも大きくするために，

$$F = 0.7 + |C(0, 0.25)| \tag{9}$$

とする．ここで，$C(\mu, \sigma)$ は位置母数 μ，尺度母数 σ であるコーシー乱数を生成する関数である．

4.5 擬似コード

図 6 に DE/rand/1/exp の擬似コードを参考に，NRDE/CHR の擬似コードを示す．

5 実験

5.1 テスト問題

本実験では，変数間依存性，悪スケール性，および大谷構造を有する問題を用いる [14]．変数間依存性の強い問題として，稜構造を有する問題を用いる．悪スケール性のある問題として，稜構造でかつ変数によりスケールが異なる問題を用いる．大谷構造とは，微視的に見れば局所解となる多数の小さな谷が存在するが，巨視的に見れば大きな谷が一つだけ存在し，その谷が最適解となっている構造であり，Rastrigin 関数がその典型例である．

以下に，関数とその初期化領域を示す．なお，n は次元数を表している．

- f_1: Sphere 関数

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2, \quad -5.12 \leq x_i \leq 5.12 \tag{10}$$

単峰性の関数で，点 $(0, 0, \cdots, 0)$ で最小値 0 をとる．

- f_2: Rosenbrock 関数

$$f(\boldsymbol{x}) = \sum_{i=2}^{n} \{100(x_1 - x_i^2)^2 + (x_i - 1)^2\}, \quad -2.048 \leq x_i \leq 2.048 \tag{11}$$

単峰性の稜構造を有する関数で，点 $(1, 1, \cdots, 1)$ で最小値 0 をとる．

```
NRDE/CHR/1/exp()
{
  P=Generate N individuals {x_i}
                          randomly;
  Evaluate x^i, i = 1,2,···,N;
  for(t=1; t < T_max; t++) {
// 距離などの初期化
    for(i=1; i ≤ N; i++) {
      x^i.ups=x^i.downs=x^i.num=0;
      x^i.label=unknown;
    }
// 近接グラフの構成
    for(k=1;k ≤ M;k++){
      Select x^i, x^j from P randomly;
      x^p = (x^i + x^j)/2; // 入力パターン
      x^i1=the nearest individual to x^p;
      x^i2=the 2nd nearest individual
                                    to x^p;
// 山谷判定: 辺 (x^i1, x^i2) の生成
      x^i1.link[x^i1.num]=i2; x^i1.num++;
      x^i2.link[x^i2.num]=i1; x^i2.num++;
      if(f(x^i1) < f(x^i2)) {
        x^i1.ups++; x^i2.downs++;
      }
      else if(f(x^i1) > f(x^i2)) {
        x^i2.ups++; x^i1.downs++;
      }
    }
// 全ての個体について山谷判定を行う;
    for(i=1; i ≤ N; i++) {
      if(x^i.ups>0 && x^i.downs==0)
        x^i.label=谷点;
      else if(x^i.downs>0 && x^i.ups==0)
        x^i.label=山点;
    }
    for(i=1; i ≤ N; i++){
      if(x^i.label==谷点){
        for(j=0; j ≤ x^i.num; j++){
          k = x^i.links[j];
          if(x^k.label==unknown){
            x^k.label=谷近傍点;
            x^k.queen=i;
          }
        }
      }
    }
    CurrentToQueen=0;
    switch(x^i の種類) {
      case 谷点:
        p1=i; F'=0.3; CR'=1;
        break;
      case 谷近傍点:
        p1=x^i.queen; F'=0.4; CR'=1-1/n;
        CurrentToQueen=1;
        break;
      case 山点:
        p1=最良個体の添字;
        F'=0.9; CR'=[0,1] の乱数;
        break;
      default:
        p1=select randomly in [1,N]\{i};
        F'=F + |C(0,0.25)|; CR'=CR;
      }
// 探索点の生成
    p2=select randomly in [1,N]\{i,p1}
    p3=select randomly in [1,N]\{i,p1,p2}
    x^new = x_i;
    j=select randomly from [1,n]; k=1;
    do {
      x_j^new = x_j^p1+F'(x_j^p2 - x_j^p3);
      if(CurrentToQueen)
        x_j^new+ = 0.5(x_j^i - x_j^p1);
      j=(j + 1)%n;
      k++;
    } while(k ≤ n && u(0,1) < CR');
// 生存者選択
      if(f(x^new) ≤ f(x^i)) x^i = x^new;
  }
}
```

図 6: NRDE/CHR の擬似コード. ただし, x^i.ups, x^i.downs はそれぞれ点 x^i より良い隣接頂点数, 悪い隣接頂点数, x^i.queen は x^i の隣接谷点の添字である.

- f_3: ill-scaled Rosenbrock 関数

$$f(\boldsymbol{x}) = \sum_{i=2}^{n}\{100(x_1 - (ix_i)^2)^2 + (ix_i - 1)^2\}, \quad -2.048/i \leq x_i \leq 2.048/i \tag{12}$$

単峰性の稜構造を有する関数で，点 $(1, \frac{1}{2}, \cdots, \frac{1}{n})$ で最小値 0 をとる．

- f_4: Rastrigin 関数

$$f(\boldsymbol{x}) = 10n + \sum_{i=1}^{n}\{x_i^2 - 10\cos(2\pi x_i)\}, \quad -5.12 \leq x_i \leq 5.12 \tag{13}$$

多峰性の大谷構造を有する関数で，点 $(0, 0, \cdots, 0)$ で最小値 0 をとる．

表 1 に，テスト関数 $f_1 \sim f_4$ の特徴を示す [14].

表 1: テスト問題の特徴

関数	変数間依存性	悪スケール性	大谷構造
f_1	なし	なし	なし
f_2	強い	なし	なし
f_3	強い	あり	なし
f_4	なし	なし	あり

また，図 7, 8, 9 に，$n = 2$ のときの関数 f_1，f_2，f_4 のグラフを示す．

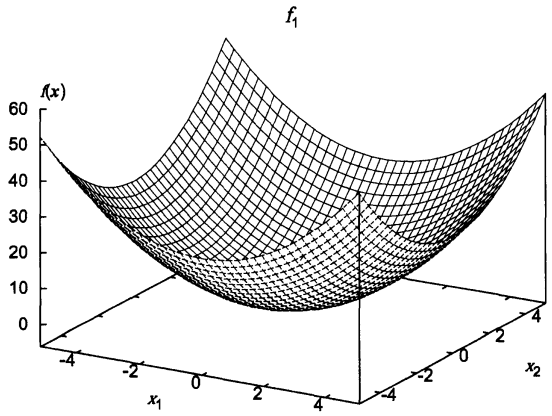

図 7: 関数 f_1 のグラフ

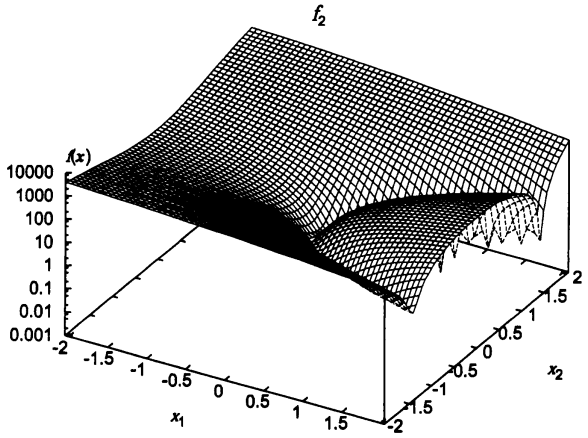

図 8: 関数 f_2 のグラフ

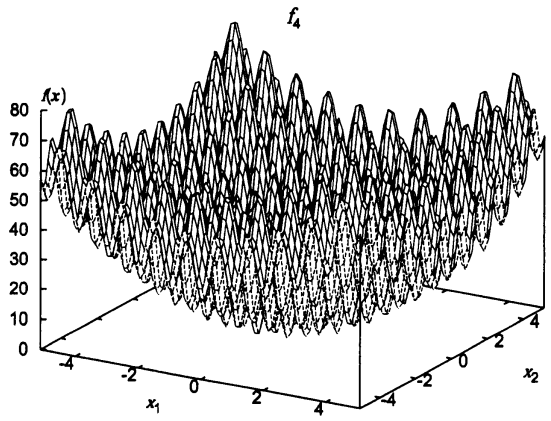

図 9: 関数 f_4 のグラフ

5.2 実験条件

次元数 $n = 30$ に設定し，f_1 から f_4 の関数を最適化する．rand 戦略を用いる標準の DE/rand，近接グラフとして競合ヘブ則を用いて生成したグラフ (CHR)，相対近傍グラフ (RNG) あるいは Gabriel グラフ (GG) を利用した NRDE の 4 つのアルゴリズムの性能を比較する．DE の設定は，個体数 $N = 50$, F と CR の組合せを $(0.7, 0.9)$ とする．NRDE/CHR の設定は，パターン数 $M = 100(= 2N)$ とする．各関数について 30 回の試行を行い，結果を考察する．個体数は $2n$ から $10n$ 程度が良いとされているが，効率性を重視し少し小さい値とした．なお，試行打ち切り条件は，文献 [14] と同様に，(1) 評価値が 1.0×10^{-7} 以下に達した，(2) 探索打ち切り評価回数が f_1 および f_2 は $2n \times 10^5$，f_3 は $5n \times 10^5$，f_4 は $3n \times 10^5$ に達した，のいずれかの条件を満足する場合とした．

5.3 実験結果

実験結果を表2に示す．許容精度を満足した回数を "$< 1e{-}7$"，実際に関数を評価した回数とその標準偏差を eval，そのうち子ベクトルが良かった回数を succ，悪かった回数を fail，成功率を rate に示した．

表 2: 実験結果

function	algorithm	$< 1e^{-7}$	eval	success	fail	rate(%)
f_1	rand(0.7,0.9)	30	57,899.20± 1281.11	11739.60	46108.60	20.29
	RNG(0.7,0.9)	30	34,877.43± 703.08	10741.40	24085.03	30.84
	CHR(0.7,0.9)	30	21,842.27± 598.07	7649.10	14142.17	35.10
	GG(0.7,0.9)	30	**20,348.47**± 1085.50	10246.53	10050.93	50.48
f_2	rand(0.7,0.9)	30	561,565.67±17282.92	19680.60	541834.07	3.50
	RNG(0.7,0.9)	30	102,809.20± 3506.04	14451.13	88307.07	14.06
	CHR(0.7,0.9)	30	**92,855.10**± 4377.13	15638.97	77165.13	16.85
	GG(0.7,0.9)	7	83,404.00± 2737.57	14709.00	68644.00	17.65
f_3	rand(0.7,0.9)	30	558,257.67±16191.24	19656.33	538550.33	3.52
	RNG(0.7,0.9)	30	**86,592.37**± 3187.05	15577.77	70963.60	18.00
	CHR(0.7,0.9)	30	88,212.13± 3257.75	15722.30	72438.83	17.83
	GG(0.7,0.9)	16	78,482.31± 3391.60	16903.69	61527.62	21.55
f_4	rand(0.7,0.9)	30	160,204.97± 4865.90	14865.50	145288.47	9.28
	RNG(0.7,0.9)	30	136,979.10± 4569.43	14429.90	122498.20	10.54
	CHR(0.7,0.9)	30	**100,669.47**± 3091.08	11599.17	89019.30	11.53
	GG(0.7,0.9)	29	222,728.83±14830.97	13333.55	209344.28	5.99

f_1 については，NRDE/GG が最も効率よく近似解を探索できており，次に NRDE/CHR の効率が優れているが，その差はわずかである．f_2 については，NRDE/CHR が最も効率よく近似解を探索できている．次に NRDE/GG の評価回数が最も少ないが 23 回の試行で近似解の発見に失敗しているため，実質的には NRDE/RNG の方が NRDE/GG より効率が優れていると考えられる．f_3 については，NRDE/RNG が最も効率よく近似解を探索できており，次に NRDE/CHR の効率が良いが，その差はわずかである．f_2 の場合同様，NRDE/GG の評価回数が最も少ないが，14 回の試行で近似解の発見に失敗しているため，NRDE/RNG および NRDE/CHR の方が NRDE/GG より効率が優れていると考えられる．f_4 については，NRDE/CHR が最も効率よく近似解を探索できており，次に NRDE/RNG の効率が優れている．以上のことから，安定的な探索の観点からは，NRDE/CHR がもっとも優れており，次に NRDE/RNG，DE/rand となる．NRDE/GG は問題に依存した不安定さがある．近似解の発見までに要する関数評価回数について NRDE/CHR と DE/rand を比較すると，NRDE/CHR は f_1 については 62.3%，f_2 については 83.5%，f_3 については 84.2%，f_4 については 37.2%の削減に成功している．同様に，NRDE/CHR と NRDE/RNG を比較すると，f_1 については 37.4%，f_2 については 9.7%，f_4 については 26.5%の削減に成功している．なお，f_3 については 1.9%程度増加しているが，増加率はわずかであると考えられる．以上のことから，関数評価回数の削減の観点からも NRDE/CHR は DE/rand に比して優れており，NRDE/RNG に比しても優れている又は同等の性能を示していることがわかる．

各問題における最良個体の関数値について，その平均値の変化を図 10 から図 13 に示す．横軸は世代数，縦軸が関数値である．

f_1 については，NRDE/GG と NRDE/RNG がほぼ同程度の探索効率である．次に，NRDE/RNG となり，DE/rand はかなり収束が遅い．f_2 については，NRDE/CHR が最も効率的に探索し，次に NRDE/RNG が探索に成功している．DE/rand と NRDE/GG は最大評価回数までに満足できる解を探索できなかった．f_3 については，NRDE/RNG がもっとも効率的に探索し，次に NRDE/CHR が探索に成功している．DE/rand は探索が遅く，NRDE/GG は最大評価回数までに満足できる解を探索できなかった．f_4 についても，NRDE/CHR が最も効率的に探索を行っている．次に NRDE/RNG が探索に成功しており，DE/rand がやや劣った効率である．NRDE/GG は探索が遅く満足できる解を発見できない場合があった．

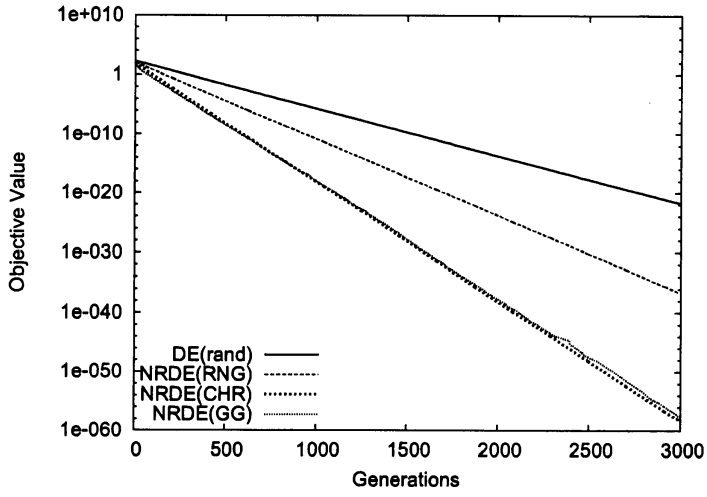

図 10: f_1 における関数値の変化

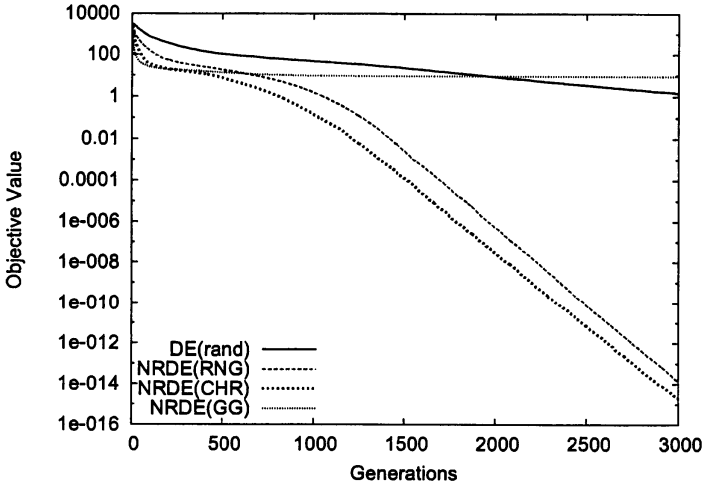

図 11: f_2 における関数値の変化

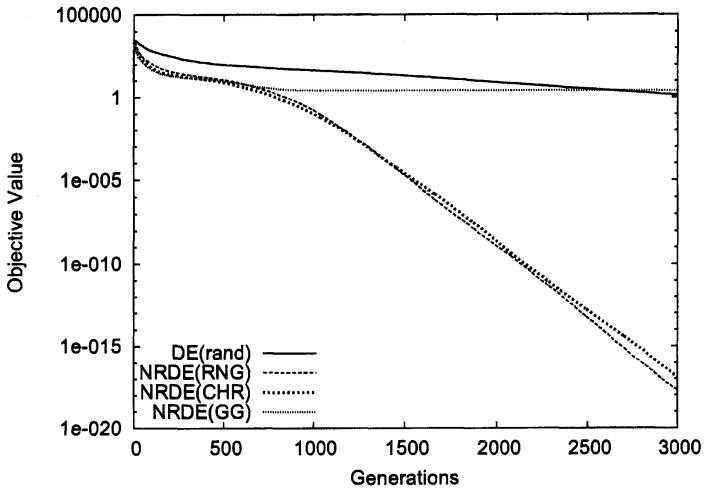

図 12: f_3 における関数値の変化

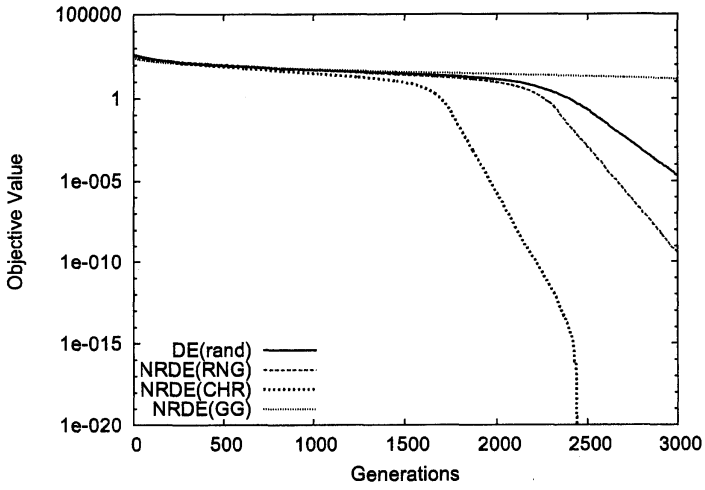

図 13: f_4 における関数値の変化

谷 (valley)，谷近傍 (near valley)，山 (hill) の点の数の変化 (左軸参照) と最良値 (best, 右軸参照) の変化について，NRDE/CHR の場合を図 14 から図 17 に示す．

f_1 については，谷点は定常的に約 5 個であり，谷近傍点は定常的に 12 個程度で，山点は定常的に 17 個程度である．すなわち，巣の近傍での近傍探索は 17 個体程度，巣への帰巣行動も 17 個体程度，通常の DE による大域探索を 16 個体程度が定常的に行っていると考えられる．f_2 については，初期は谷点が 6〜7 個体まで減少し，その後増加し，中盤以降は定常的に 10 個体程度となっている。谷近傍点も 7〜8 個体程度まで減少し，その後は谷点と同様の変化を見せている．山点は初期に 18 個体程度まで増加し，その後減少し，再度やや増加し，中盤以降は定常的に 18 個体程度となっている．最良値の変化とあわせると，探索の中盤までは，谷点として局所解である $(0,0,\cdots,0)$ しか発見できておらず，その後最適解である $(1,1,\cdots,1)$ も発見して，その結果谷点及び谷近傍点が増加したと考えられる．最良値の最適値への収束もこの変化に同期していることが分かる．f_3 については，初期は谷点が 8 個体，その後増加し，中盤以降は定常的に 14 個体程度となっている．谷近傍は初期に 12 個程度まで増加し，その後若干減少し再度増加し，中盤以降は定

常的に 10 個体程度となっている。山点は初期に 14〜16 個であるが，その後定常的に 18 個体程度となっている．最良値の変化とあわせると，f_2 と同様に，一度 $(0,0,\cdots,0)$ 付近に収束し，そこから狭小な谷を通って $(1,\frac{1}{2},\cdots,\frac{1}{n})$ に向かって移動するという探索方法がとられていると考えられる．f_4 については，序盤は谷点が 8 個体から 12 個程度，谷近傍点が 14 個体から 18 個程度へ増加し，山点は 15 個体から 12 個体程度へ減少している．中盤は定常的に谷点と山点が 12 個体程度，谷近傍点が 20 個体程度である．終盤には谷点は 5 個体程度，谷近傍点は 12 個体程度まで減少し，山点は 18 個体程度まで増加増加している．最良値の変化とあわせると，f_4 は巨視的には谷がただ一つだけ存在するため，序盤は探索空間内にランダムに生成した探索点は大域的最適値方向へ探索が行われる．中盤は f_4 の多数の小さな谷にとらえられるため谷点および谷近傍点が増加する．探索が大域的最適解の周辺まで進むと，大域的最適解である谷点の周辺は局所的には f_1 と同様な単峰性あるため，終盤では谷点と谷近傍点が再度減少し，最良値が急速に最適値に収束している．

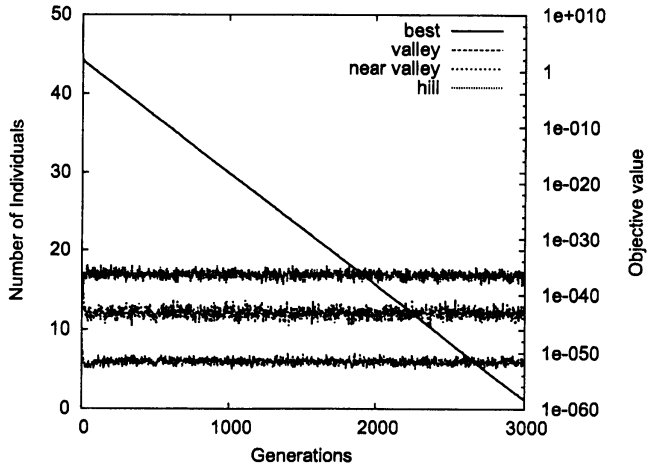

図 14: f_1 における山谷判定 (NRDE/CHR)

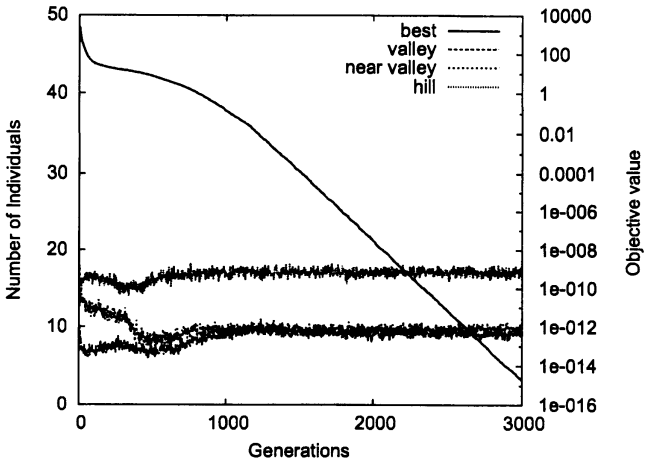

図 15: f_2 における山谷判定 (NRDE/CHR)

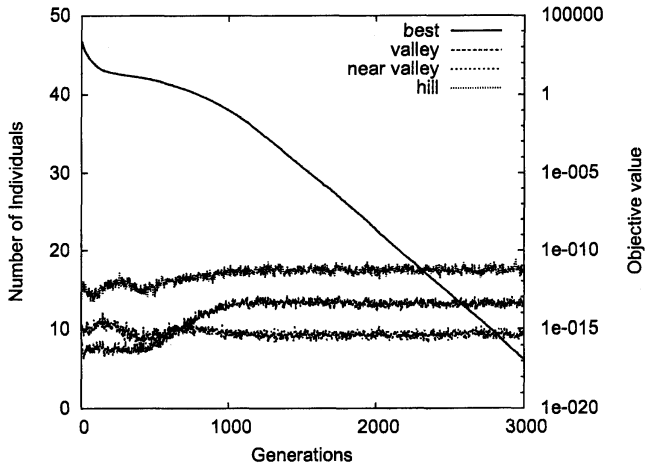

図 16: f_3 における山谷判定 (NRDE/CHR)

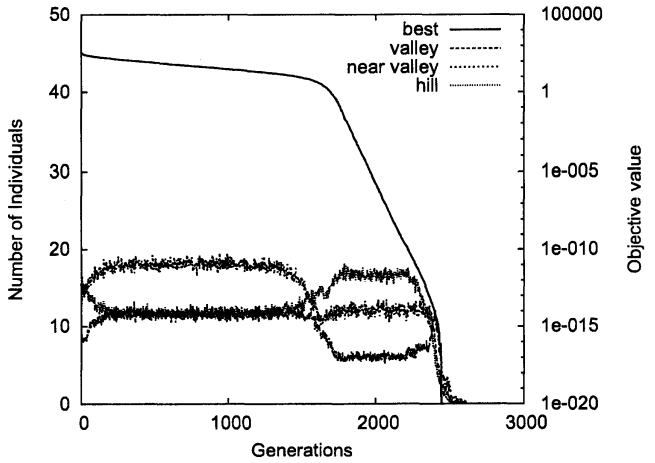

図 17: f_4 における山谷判定 (NRDE/CHR)

6 おわりに

本研究では，集団的最適化アルゴリズムにおいて，探索効率を向上する方法として提案した，巣形成と役割分担に基づく最適化アルゴリズムの改良を行った．巣形成と役割分担に基づく最適化アルゴリズムは，探索点による近接グラフに基づく山谷構造の推定と山谷判定を行い，探索点の近傍における山谷構造に基づいて探索モードを変更することで，探索効率を向上する方法である．しかし，最適化アルゴリズムの探索性能は，近接グラフの辺の数すなわちグラフの疎密によって異なり，しかも適切な辺の数は目的関数の形状によって異なることが確認されている．本研究では，探索アルゴリズムとして DE を採用した巣形成と役割分担に基づく最適化アルゴリズム NRDE に対して，生成する近接グラフの辺の数を制御できる競合ヘブ則により生成した近接グラフを用いた NRDE/CHR を提案した．NRDE/CHR では，競合ヘブ則により生成した近接グラフによる山谷判定で探索点を谷点 (女王蟻)，山点 (帰巣蟻)，谷近傍点 (雄蟻)，その他の点 (探索蟻) の 4 種類に分類する．女王蟻 (谷点) と隣接する雄蟻 (谷近傍点) が巣を形成し近傍探索を行

い，帰巣蟻は最良の女王蟻のもとへ帰巣する中域探索を行い，探索蟻は通常の大域探索を行う．競合ヘブ則により生成した近接グラフによる NRDE/CHR，Gabriel グラフによる NRDE/GG，相対近傍グラフによる NRDE/RNG 及び標準的 DE/rand について性能を比較することにより，NRDE/CHR がいずれの問題に対しても安定的かつ効率的に探索を行う最適化アルゴリズムであることが示された．

今後は，競合ヘブ則によるグラフにおいて問題によって適切な入力パラメータ数 M を選択する方法について検討を行うこと，局所探索と大域探索の実現方法を検討すること，self-adaptive のようにパラメータを動的に制御する手法と性能比較を行うこと，PSO などの集団的最適化アルゴリズムに適用することなどを予定している．

謝辞 この研究の一部は，日本学術振興会科学研究費補助金 基盤研究 (c) (No. 22510166, 24500177) および広島市立大学特定研究費（一般研究） の援助を受けた．

参考文献

[1] R. Storn and K. Price: "Minimizing the real functions of the ICEC'96 contest by differential evolution", Proc. of the International Conference on Evolutionary Computation, pp. 842–844 (1996).

[2] R. Storn and K. Price: "Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces", Journal of Global Optimization, **11**, pp. 341–359 (1997).

[3] 高濱, 阪井, 原："低精度の近似モデルを用いた比較推定法による differential evolution における関数評価回数の削減", 電子情報通信学会論文誌, **J91-D**, 5, pp. 1275–1285 (2008).

[4] T. Takahama and S. Sakai: "Reducing function evaluations in differential evolution using rough approximation-based comparison", Proc. of the 2008 IEEE Congress on Evolutionary Computation, pp. 2307–2314 (2008).

[5] 高濱, 阪井："低精度近似モデルを利用した ε 制約 differential evolution による効率的な制約付き最適化", 人工知能学会論文誌, **24**, 1, pp. 34–45 (2009).

[6] 阪井, 高濱："多次元空間における近傍構造を利用した最適化アルゴリズムに関する一検討", 不確実・不確定性下での意思決定過程, 数理解析研究所講究録 1682, pp. 184–192 (2010).

[7] 阪井, 高濱："巣形成と役割分担を用いた最適化手法に関する一考察", 確率的環境下での意思決定解析, 平成 24 年度 RIMS 研究集会 (2012).

[8] 高濱, 阪井："競合ヘブ則によるグラフ生成に基づく種分化型 differential evolution の提案", 第 22 回インテリジェント・システム・シンポジウム講演論文集 (2012).

[9] T. Takahama and S. Sakai: "Differential evolution with graph-based speciation by competitive hebbian rules", Proc. of the Sixth International Conference on Genetic and Evolutionary Computing (ICGEC2012), pp. 445–448 (2012).

[10] K. R. Gabriel and R. R. Sokal: "A new statistical approach to geographic variation analysis", Systematic Zoology, **18**, pp. 259–270 (1969).

[11] G. T. Toussaint: "The relative neighborhood graph of a finite planar set", Pattern Recognition, **12**, 4, pp. 261–268 (1980).

[12] D. G. Kirkpatrick and J. D. Radke: "A framework for computational morphology", Computational geometry (Ed. by G. Toussaint), North-Holland, pp. 217–248 (1985).

[13] T. Martinetz and K. Schulten: "Topology representing networks", Neural Networks, **7**, 3, pp. 507–522 (1994).

[14] 田中, 土谷, 佐久間, 小野, 小林："Saving MGG: 実数値 GA/MGG における適応度評価回数の削減", 人工知能学会論文誌, **21**, 6, pp. 547–555 (2006).