卒業研究 I

ランダム・ヘテロジニアスなパラメータ設定を用いた島型遺伝的アルゴリズムの性能評価

東京大学教養学部広域科学科広域システム分科

氏名　猪鼻 真裕

学籍番号　08-110401

指導教員　福永 Alex

2012 年 12 月 14 日

目次

1 はじめに

1.1 序論

遺伝的アルゴリズム (Genetic Algorithm; GA) は，1960〜70 年代に J. H. Holland によって提案された近似解法の 1 つで，生物進化に着想を得たアルゴリズムである [6]．GA では，初期に生成された多数の解候補を個体と呼び，個体に進化オペレータを適用して新個体を生成し，自然淘汰の原理を適用して母集団を更新することによって探索を進める．GA は，従来難しいとされていた非線形・微分不可能な関数に対しても適用可能であり，提案以来，様々な改良が加えられてきた．新幹線の先頭部の形状設計やカメラのレンズ系設計における多目的最適化など，工学的な応用にも広く用いられている．

評価関数が具体的な数式として表現できず，多大な実行時間を必要とするシミュレーションにより解の評価を行わなければならない場合などは，実行を並列化し計算速度高める工夫が必要になる [4]．GA は並列化に適したアルゴリズムであり，並列化遺伝的アルゴリズムについて多くの研究が行われてきた．並列化 GA は大きく 4 つの種類に分類される：(1) マスター・スレイブモデル，(2) セルラーモデル (cellar model, fine-grained model)，(3) 島型モデル (Islands model, Distributed model, course-grained model)，(4) 階層モデルである [1]．

並列化を行い，2 つ以上のアルゴリズムを組み合わせて探索を行う手法を，一般にポートフォリオ戦略と呼ぶ．ポートフォリオはもともとは経済学における用語で，手持ちの資源を用いて投資を行う際，資源を複数の投資先に分配することでリスクヘッジを図る手法である．ポートフォリオの概念を探索アルゴリズムに応用する手法が，1997 年に提案された [7]．島型モデルも同様に，探索におけるリスクヘッジ及び並列化による効率化を目指し，90 年代に提案された．島型モデルでは，探索を行う母集団を複数の小集団に分割し，1 つの小集団を島と呼ぶ．進化オペレータは島ごとに独立に適用され，探索は独立して進む．ある一定の周期がきたら，島間で個体情報を交換する移住 (migration) と呼ばれる操作を行う．これは，ポートフォリオ戦略におけるアルゴリズム間通信に対応し [7]，探索性能を向上させると言われている [12][8]．

近年，クラウドコンピューティングや P2P[2]，GPU を用いた計算 [11] などの並列化環境が広く普及しており，大規模並列化が容易になっている．島型モデルはこの環境で並列化するのに特に適したモデルであり，最近，このモデルに関して多くの研究が行われている．

大規模並列化に際して，どのようなポートフォリオを組むかというのは重要な問題である．単一のアルゴリズムを各島で実行するものを Homogeneous 環境，複数のアルゴリズムを組み合わせて並列化したものを Heterogeneous 環境と呼ぶ．当初は，全ての島で同じパラメータ設定を用いる，Homogeneous 島型 GA が多く研究されてきた．しかし，一般には，複数のアルゴリズムを組み合わせるポートフォリオ戦略において，挙動の異なるアルゴリズム同士を組み合わせることによって更なる性能向上を図ることができると報告されている [7]．また，No Free Lunch 定理 [15] によれば，あり得る全ての問題を等価に考えると，他のアルゴリズムよりも平均性能がよいアルゴリズムは存在しない．すなわち，それぞれのアルゴリズムに得意・不得意な問題が考えられるため，複数の異なるアルゴリズムを組み合わせることがポートフォリオ戦略において重要である．

1.2 背景

島型モデルにおけるポートフォリオ戦略において，ポートフォリオの要素となるアルゴリズム，すなわち，島ごとにどのようなアルゴリズムを割り当てるかを決定する方法には，あらかじめ選んできて固定するものや [10][16]，Hyper-Heuristics によって決めるものなどが考えられ [2]，様々な研究が行われてきた．

Fei Peng らは実数関数値最適化問題に対して，DE, PSO, G3PCX, CMA-ES（それぞれ進化的計算手法の一種）を組み合わせたアルゴリズム PAP (Population-based Algorithm Portofolios) を提案し，Heterogeneous 環境の有効性を実証した [10]．E. L. Yu らは多峰性の関数最適化問題に対して，複数のニッチングアルゴリズムを組み合わせたポートフォリオ戦略を作成し，Heterogeneous 環境の有効性を実証した [16]．これらの研究は Heterogeneous 環境の有効性に焦点を絞ったため，島の数 = 要素のアルゴリズム数であり，大規模並列化による影響は調べられていない．Biazzini らはポートフォリオ作成の際に，動的にアルゴリズムを割り当てるもの，適応的にアルゴリズムを割り当てるものなど様々な Hyper-Heuristics を試して，その挙動について調べた [2]．また，P2P 環境における大規模並列化を前提としており，島の数を N とおくと $N = 2^{16}$ までの並列化が試された．どの HHs が効果的であるかは問題に依って変わるが，単純な静的割り当てが許容的だという結果を実証した．

これらの研究から，Heterogeneous なポートフォリオ戦略の有効性が実証されてきたが，その要素として選んでくるアルゴリズムには恣意性がある．実数値関数最適化に対しては，DE や PSO，ニッチングアルゴリズムなどのそれに特化したアルゴリズムやパラメータ設定を選べばよいが，一般に，扱う問題がブラックボックスである実世界の問題などでは，どのアルゴリズムを要素として選べばよいかが明確ではない．すなわち，事前の知識が全くない状況では，これらの手法を適用するのは難しい．探索中にオンラインでパラメータを調節する適応的な GA も多く研究されているが，適応に用いるパラメータ値などの設定は試行錯誤しなければならず，本質的な煩雑さは解消されていない．

それに対して，シンプルな GA の枠組みのみを用いた場合の Heterogeneous 環境の有効性を調べる研究も行われてきた．Miki らは，島型 GA において，まず突然変異率や交叉率の最適値が，集団サイズや扱う問題，島の数などによって異なることを示した [9]．そして，それら複数のパラメータ設定を組み合わせた島モデルの有効性を実証した．しかし，パラメータ値の組み合わせは離散的であり，最適なチューニングがなされているとは言えない．さらに，扱った問題も限定的であり，並列化数も $N = 9$ であった．Gong と Fukunaga は，組み合わせるパラメータ値を連続的かつランダムに選ぶランダム・ヘテロジニアス GA を提案し，$N = 100$ までの並列化を試して，その有効性を実証した [5]．これは，要素となるアルゴリズムを選択するポートフォリオ戦略とは対極にある考え方であり，ポートフォリオ作成の手間と探索性能のトレードオフを目指した枠組みである．さらに重要なことは，これらのアルゴリズムはパラメータフリーなアルゴリズムであることである．一般に，あるアルゴリズムを採用する場合，そのパラメータは適切な値に設定されなければ探索の効率を下げる可能性があるが，上記の枠組みはその煩雑さの本質的な解消を目的としている．

しかし，これらのアルゴリズムに関して，その挙動が十分に理解されているとは言い難い．例えば，島型 GA の探索性能に重要な影響を与えるものに migration が挙げられるが，その影響は十分に調べられていない．一般に，migration は探索性能を大幅にに向上させると考えられているが [3][8]，一概にそうとはいえない．Skolicki と De Jong は，島型 GA における効果的な資源割り当てを，島内

の多様性・島間の多様性という 2 つの観点から考察し，そのことによって migration の有効性を浮き彫りにした [12]．島の数を増やすと migration によって島同士の情報交換が多くなされるが，問題に依ってはそれが負の効果をもたらすことがあることを，人工的な関数を用いて実証した．池田らは，問題の構造として UV 構造を提案し，同様に migration によって探索性能が落ちる現象を指摘している [18]．

1.3 本研究の目的

本研究では，ランダム・ヘテロジニアスな島型 GA において，migration を行う場合と行わない場合で比較実験を行い，migration の効果が探索性能に与える影響を検証する．さらに，新たに narrow Heterogeneous 環境を提案し，migration の効果を有効に用いる枠組みを検討する．また，$N = 128$ まで並列化した際の挙動を調べ，narrow Heterogeneous GA の大規模並列化に際する挙動の変化を調査する．そして，パラメータフリーのアルゴリズムである narrow Heterogeneous GA のロバスト性を，代表的なパラメータ設定を用いた Homogeneous GA との比較により検証する．

本論文は，以下の通りに構成される．まず 2 章で，一般的な遺伝的アルゴリズムの詳細と，本研究で用いるアルゴリズムについて述べる．次に 3 章において，本研究で最適化の対象として扱う評価関数について述べる．4 章では，提案手法の比較対象である best Homogeneous GA のパラメータ設定の決定方法と，その結果を述べる．5 章で比較実験の結果を示し，6 章で特に migration の効果について詳細に解析する．7 章では，大規模並列化に際して，並列化数が探索アルゴリズムに与える影響について考察する．8 章では，提案手法が同等の条件の下での比較手法に比べて十分なロバスト性を持つことの検証実験を行う．9 章に，本研究で行った実験から得られる知見をまとめる．

2　GA の設定

2.1　遺伝子表現 (genotype，phenotype)

実数値関数最適化問題に GA を適用する場合，遺伝子表現には大きく分けて 2 種類が考えられる．1 つは伝統的なバイナリ列を遺伝子として用いて，そこから実数に変換する方法で，もう 1 つは実数値を直接遺伝子表現として用いる方法である．前者をビット列 GA (Bit-string GA)，後者を実数値 GA (Real-coded GA) と呼ぶ．実数値関数最適化問題に対しては，実数値 GA の方が優れた探索性能を持つことが知られている．しかし，それは後述する交叉方法や世代交代モデルに複雑なものを用いて，実数値関数最適化に特化しているからであり，実装も複雑になる．本研究の目的は，特定の評価関数を仮定せずに，シンプルな GA の枠組みで Homogeneous 環境と Heterogeneous 環境の比較を行うことにあるため，本研究ではビット列 GA を採用した．

ただし，Homogeneous，Heterogeneous の概念はパラメータ値を持つアルゴリズムに共通に適用できるので，実数値 GA を用いた場合の研究は今後の課題である．

本研究では，遺伝子表現には 1 次元につき 10 ビットのバイナリ列を用い，最上位ビットを符号表現部，下位 9 ビットを 10 進数整数表現部とした．変換には，グレイ符号を用いるものや IEEE 規格に基づく実数変換法なども考えられるが，本研究では先述したようにシンプルに 2 進符号を用い，変換した数字を 100 で割った．すなわち，各次元の実行可能領域は $-5.11 \leq x_i \leq 5.11$ となる．

2.2　突然変異 (mutation)

突然変異は，ある 1 つの個体から新個体を生成する進化オペレータである．個体のビット列を先頭から順に見ていき，あらかじめ設定した突然変異率 (mutation rate) の下で各ビットを反転させ，新しい個体を生成する．イメージを図 2.1 に示す．例えば，個体の遺伝子長が 10 ビットで突然変異率が 0.1 の場合，1 つのビットが反転した新個体が生成されることが期待できる．

2.3　交叉 (crossover)

交叉は，選んできた 2 つの親個体のビット列を組み合わせて新しい個体を生成する進化オペレータである．

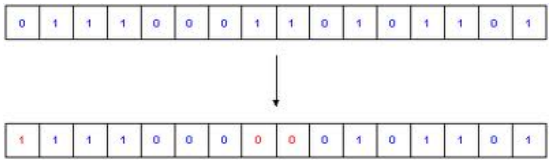


図 2.1　突然変異

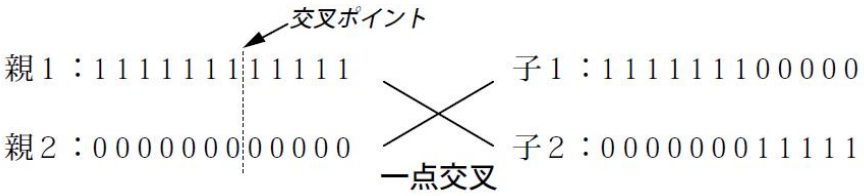


図 2.2　一点交叉

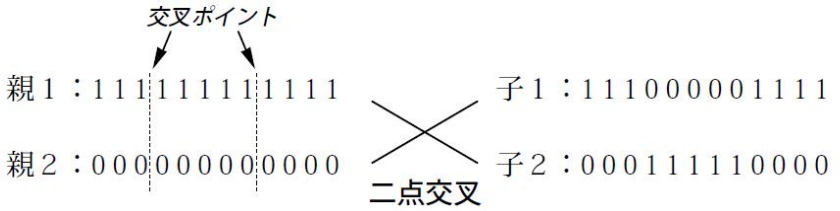


図 2.3　二点交叉

図 2.4　一様交叉

図 2.2 のように組み合わせるものを一点交叉と呼ぶ．2 点以上の複数点で交叉を行うものを，多点交叉と呼ぶ．図 2.3 は二点交叉の例である．図 2.4 のように，各ビットごとに交叉を行うものを一様交叉と呼ぶ．

予備実験において，一点交叉よりも多点交叉のほうが効果的である，という結果が得られたが，先行研究 [5] との比較の観点や，シンプルな枠組みを用いるという方針を一貫させるため，本研究では一点交叉を用いた．

よって本研究では，親 2 個体を選んできて，あらかじめ設定した交叉率 (crossover rate) の下で一点交叉を行い，新個体を 2 個体生成することとした．

2.4　親選択 (selection)

母集団から，次世代の子供を生成する親個体を選ぶ方法にも，いくつかの種類がある．

- エリート選択
  適応度の 1 番高いもの，2 番目に高いもの，…といったように，上から順番に選んでいく方法．

- ルーレット選択
  適応度の高い個体から，その適応度に応じた重み付けのもとで，確率的に選ぶ方法.
- トーナメント選択
  母集団から決められた数の個体をランダムに抽出し，そのグループの中で最も適応度が高い個体を選ぶ方法.
- ランダム選択
  適応度には関係なく，ランダムに個体を選ぶ方法.

本研究では，先行研究 [5] でも用いられ，一般的にも多く用いられるルーレット選択を採用し，実験を行った.

2.5 世代交代 (reproduction)

次世代の生成法には，大きく分けて 2 通りの方法がある.

- Generational GA (G-GA)
  上記の，進化オペレータの適用 → 子供個体の生成という過程を，子供の数が集団サイズに達するまで行い，新しく出来た集団を前の世代の集団と全て取り替えることで，世代交代を行う方法.
- Steady State GA (SS-GA)
  新しく生成された個体を母集団に戻すことで新母集団を生成し，その新母集団から何らかの選択方法を用いて集団サイズ数の個体を選択することで，世代交代を行う方法. G-GA では，古い個体は必ず取り除かれるが，SS-GA では世代交代が連続的なため，個体情報の変化も連続的であるという特徴がある.

本研究では，新しく生成された子供個体を母集団中に戻し，その集団にエリート選択を適用して母集団を更新する，SS-GA の枠組みを採用し実験を行った．これによって，並列環境を逐次環境でシミュレートする際に，各島に計算資源を均等に分配する，という条件を容易に達成することができる.

2.6 移住 (migration)

移住 (migration) は，島モデル GA において，独立に探索を進める島同士で個体情報を交換する操作である．なお，進化計算分野では migration（移住）という用語を用いるが，一般的に，個体は島Ａから島Ｂに移住するのではなく，個体のコピーが島Ａから島Ｂに送信される．migration にも，大きく分けて 2 通りの方法がある.

- 最良個体更新時移住
  各島で，それまでに発見されている最良の個体よりも適応度が高い個体が発見されたタイミングで，その個体を他の島にコピーすることにより情報交換を行う方法.
- 周期的移住
  あらかじめ migration を起こす周期を決めておき，その周期が来たら一斉に個体の交換を行う方法.

また，migration で個体をどの島に送るのかも，様々な設定が考えられる．これを，移住グラフのトポロジと呼ぶ.

- リング型
  島をリング状に連結させ，隣接しあう島に個体を送る方法.
- 完全グラフ型
  ある島からの個体を他の全ての島に送る方法.
- ランダム型
  どの島に個体を送るかは，初期化の際にランダムに決める方法.

本研究では，先行研究に習い [5]，migration のタイミングは最良個体更新時とした．こうすることで，実際に並列化した際に各プロセッサの探索を同期するという困難点を回避することができ，独立に探索を進める島モデルの特徴を生かすことができる．また，最良個体更新時に migration を行う設定は，migration の頻度が周期的に migration を起こすものに比べて少ないことが報告されており [5]，通信コストを削減することができる.

移住グラフのトポロジには，リング型の一種で，隣り合う 2 つの島に個体を送るバイリング型を採用した．これは，他のモデルに比べて migration をローカルに行うモデルであり，大域的な多様性の維持につながるという報告がある [17]．しかし，移住グラフのトポロジの効果についての一般的な知見は得られているとはいえない状況であり，今後の研究課題といえる.

2.7 ランダム・ヘテロジニアス GA

最後に，本研究におけるアルゴリズムの設定をもう一度まとめ直す．先行研究 [5] にならって，ランダム・ヘテロジニアス GA の実装を行った．トップレベルの擬似コードを Algorithm1 に示す.

Algorithm 1 Random Heterogeneous GA

```
int N // the number of islands
Initialize N islands
while termination criteria are false do
  for i = 1 to N do
    Evolve_1step(island[i])
  end for
  for i = 1 to N do
    if island[i] updated local best then
      Migrate local-elite to neighbor of island[i]
    end if
  end for
  Check for terminal criteria
end while
```

実数値関数最適化問題に対しては，実数値を遺伝型として用いる実数値 GA[14] も多く用いられているが，最もシンプルな実装としてバイナリコーディングを用いた．突然変異は，各ビットに対して突然変異率 μ でビット反転を行うものとし，交叉は，交叉率 c で 2 個体を用いた一点交叉を行うものとした．親の選択にはルーレット選択，淘汰にはエリート選択を用いた．世代交代モデルは steady-state とした．これにより，各島に均等に資源を分配し，探索を並列に行う環境をシミュレートできるようにした．この部分の擬似コードを Algorithm2 に示す．

Algorithm 2 Evolve_1step

```
for i = 1 to 2 do
  parent[j] ⇐ choose_parent(island[i]) // using roulette selection
end for
if Random[0,1] ≤ Prob_crossover then
  (child[1], child[2]) ⇐ Cross(parent[1], parent[2])
end if
for i = 1 to 2 do
  if Random[0,1] ≤ Prob_mutation then
    Mutate(child[j])
  end if
end for
Evaluate children
Reproduce(island[i])
```

migration は先行研究 [5] と同様に最良個体が更新されたときに行うものとした．グラフのトポロジには，双方向リングを用いた．以上の設定によって，決定すべきパラメータは集団サイズ p, 突然変異率 μ，交叉率 c の 3 つのみとなる．migration のタイミングを上記の設定にすることで，migration の周期パラメータを削減している．

2.7.1 broad Heterogeneous GA (**先行研究**)

ランダム・ヘテロジニアス GA では，島の初期生成の際に，パラメータをランダムに設定する．先行研究では，集団サイズ，突然変異率，交叉率のそれぞれのパラメータを，$p \in [10, 100]$, $\mu \in [0.0, 1.0]$, $c \in [0.0, 1.0]$ から一様に選ぶ設定であった．しかし，一般的に有効と考えられている突然変異率や交叉率と比べて，この設定では非効率的な探索性能を持つ島が生成される確率が高いと考えられる．特に，突然変異率の推奨値はビット長 L に対して $\mu = 1/L$ といわれており，今回の設定では後述するように $1/L = 0.01$ となるので，最適ではない突然変異率を持った島が生成される確率が高い．以上の枠組みを，本研究中では broad Heterogeneous GA と呼んで，次に提案する手法と区別する．

表 1　broad Heterogeneous GA (先行研究) と narrow Heterogeneous GA (提案手法) の初期設定時にランダムサンプリングするパラメータ空間の違い

	broad Heterogeneous GA	narrow Heterogeneous GA
集団サイズ	$[10, 100]$	$[10, 120]$
突然変異率	$[0.0, 1.0]$	$[0.0, 0.02]$
交叉率	$[0.0, 1.0]$	$[0.3, 1.0]$

2.7.2　narrow Heterogeneous GA (**提案手法**)

前節での考察の結果，本研究では，集団サイズ，突然変異率，交叉率のそれぞれのパラメータを，$p \in [10, 120]$，$\mu \in [0.0, 0.02]$ ($= [0.0, 2/L]$)，$c \in [0.3, 1.0]$ から一様に選ぶ設定を提案する．これは，一般的なパラメータ設定の知識をもとに，妥当なパラメータ値付近にはばを設け，その範囲にランダムサンプリングするパラメータ空間を制限したものである．こちらを，narrow Heterogeneous GA と呼ぶこととする．表 1 に，それぞれのパラメータ設定をまとめる．これを見ると，特に突然変異率において，大幅な制限を加えてあることが分かる．

以上の 2 つのランダム・ヘテロジニアス GA を，各問題に対して最適なパラメータ値を用いたもの (best Homogeneous GA) と比較することによって，探索性能を評価する．

3　本研究で使用する評価関数

本研究で最適化する評価関数には，多くの研究でよく用いられるベンチマーク問題 6 題を用いた．先行研究 [5] で用いられていた Griewank 関数，Rastrigin 関数，Schwefel 関数に加え，2005 年の CEC competition[13] で定義されたベンチマーク問題の要素となる関数から，Sphere 関数，Rosenbrock 関数，Schwefel's problem 1.2 関数を選び，計 6 題とした．

最適化の対象となる関数には，探索が容易であるか困難であるかを判別する，いくつかの特徴が知られている．

- 変数間依存性
  ある評価関数について，以下の特徴がある場合に，その関数は変数間依存性がない (separable) という．

$$\arg\min_{(x_1,\cdots,x_n)} f(x_1,\cdots,x_n) = \left(\arg\min_{x_1} f(x_1,\cdots),\cdots,\arg\min_{x_n} f(\cdots,x_n)\right)$$

  すなわち，評価関数の最適化が，各次元ごとの最適化で達成される場合に，その関数は separable だといわれる．逆に，複数の変数間に非線形性がある場合には変数間依存性がある (non separable) といわれる．一般に，non separable な関数の方が最適化するのが難しい．

- 単峰性・多峰性
  ある評価関数が，局所解を 1 つしか含まない場合に，その関数は単峰性 (unimodal) といわれる．逆に，局所解を多数含む場合に，その関数は多峰性 (multimodal) といわれる．一般に，多峰性の関数の方が，非最適解にトラップされる確率が増えるため，探索するのが難しい．

各関数の特徴を及び探索打ち切りの基準値を，表 2 にまとめる．

本研究では，全て 10 次元の問題として扱った．よって探索領域は $[-5.11, 5.11]^{10}$ となり，探索空間のサイズは $(2^{10})^{10} = 2^{100} \approx 10^{30}$ となる．各次元 10 ビットのバイナリコーディングなので，全ビット長 $L = 100$ であり，突然変異率の推奨値は $1/L = 0.01$ となる．具体的な関数形とそのグラフ

表 2　本研究で使用する評価関数とその特徴及び探索打ち切りの基準値

function	Name	features	tolerance(optimal)
f_1	Sphere	separable, unimodal	1.0 (0.0)
f_2	Griewank	separable, multimodal	0.1 (0.0)
f_3	Rastrigin	separable, multimodal	10.0 (0.0)
f_4	Schwefel	separable, multimodal	0.01 (0.0)
f_5	Rosenbrock	non-separable, multimodal	10.0 (0.0)
f_6	Schwefel-1.2	non-separable, unimodal	1.0 (0.0)

については，付録 A にまとめる．

各関数が探索を打ち切る基準値は，予備実験にて求めた．なお，今回用いた評価関数は全て最適解の評価値に 0.0 という値を持つため，表 2 の基準値は解の近似精度と同等である．

予備実験では，broad Heterogeneou GA の設定を用いて，特定の基準値を設定せずに 100000 世代の探索を 100 回を行い，最終的な評価関数の値を参考に決定した．なお，ここでの 1 世代とは，steady-state GA の各ステップにおいて生成される新個体の数と等しいものとする．また，本研究と同様にビット列 GA を用いた関連研究 [16] での設定値も参考とした．評価関数自体は異なるものを用いているが，1 次元あたりどの程度の精度が許容的なのかを参考に，本実験での基準値を定めた．ビット列 GA では，実数値 GA や DE などのアルゴリズムに比べると解の精度は劣るが，本研究では絶対的な最適解の探索を目指すのではなく，相対的な枠組みの比較研究を目的としたため，表 2 の設定で問題ないといえる．

4 Homogeneous 環境における最良設定

4.1 実験設定

Homogeneous 環境と Heterogeneous 環境の比較を行うために，それぞれのパラメータ値を決定する必要がある．Heterogeneous 環境のほうは，先述したランダム・ヘテロジニアスな設定を用い，Homogeneous 環境のほうは，各評価関数ごとの最良なパラメータ設定を用いることとした．この比較を行うことによって，ランダム・ヘテロジニアス GA が，最も望ましい探索性能と比べてどの程度の探索性能を示すのかを評価できる．

そこで本章では，各問題に対する最良パラメータ設定を求める実験の結果を示す．各問題に対して，評価関数の値に 3 章で設定した基準を設け，その基準に達したら探索成功とみなし，探索を終了する．終了までに評価関数を評価した回数 (NFE; Number of Fitness Evaluation) で探索性能を評価する．NFE が小さいほど，速く最小値まで辿り着いているため，よりよい探索性能をもつとみなす．

実験は，$p \in [10, 25, 50, 75, 100, 120]$，$\mu \in [0.005, 0.01, 0.02, 0.04, 0.1, 0.3, 0.6, 0.8, 1.0]$，$c \in [0.1, 0.3, 0.6, 0.7, 0.8, 1.0]$ のそれぞれの組み合わせから全通りを試した．各設定の下で 10 回ずつ実験を行い，その NFE の平均で最良のパラメータ設定を求めた．

4.2 実験結果

実験結果のうち，上位 2 つのパラメータ設定を表 3 にまとめる．各関数において，表 3 の 1 段目の設定を最良パラメータ設定とし，以降の実験で用いることとする．

先行研究 [5] でも best Homogeneous GA との比較を行っているが，200 回のランダム設定のうち，最もよいものを最良パラメータ設定としていた．今回は，各関数ごとに $6 \times 9 \times 6 \times 10 = 3240$ 回の実験を行って最良パラメータ値を求めたので，より高精度で最良パラメータ設定が求まっていると考えられる．

結果を見ると，突然変異率と交叉率については一定の傾向が見られる．今回の設定の GA では，どの関数においても，$\mu = 0.02$，$c = 1.0$ 付近に最良パラメータ値が存在する．逆に集団サイズについては，関数ごとにばらつきが見られる．Sphere 関数 (f_1) のようになめらかで局所解が 1 つしかない簡単な関数では，集団サイズを小さくして集中的に探索を行ったほうがよいことが分かる．一方，Griewank 関数 (f_2) や Rastrigin 関数 (f_3) のような多峰性の関数では，集団サイズが大きいほうが初期サンプリングを大域的に行えるため，有利であると考えられる．以上のことは既存の研究結果で得られていた知識だが，今回再確認することができ，高精度で最適なパラメータ設定を求めることができた．

最適なパラメータ設定のためには，今回のように多くの実験を行わなければならない．本実験では，6 つの関数それぞれについて最適なパラメータ値を求める実験を行ったため，総計で $3240 \times 6 = 19440$ 回の実験を行った．一般に，島の数を変えると最適なパラメータ値は変化するため [9]，並列化環境が変わるたびに同様の実験を行わなければならない．以上のように，パラメータチューニングには煩雑な作業が伴うということは，再度強調されるべきである．

表 3　実験的に求めた各関数における Homogeneous GA の最良パラメータ設定

function	pop-size	mut-rate	cross-rate	ave-NFE	rank
f_1	10	0.02	1.0	343.6	best
	10	0.04	1.0	348.6	second
f_2	100	0.02	1.0	1512.8	best
	50	0.02	0.7	1523.4	second
f_3	120	0.01	0.8	2983	best
	50	0.04	1.0	3248.8	second
f_4	25	0.01	1.0	2303.3	best
	25	0.005	1.0	2378.2	second
f_5	50	0.04	1.0	3554.4	best
	120	0.02	1.0	5069.4	second
f_6	120	0.02	1.0	3051.2	best
	120	0.04	1.0	3329.8	second

5　島数を変えた場合における各手法の比較実験

5.1　実験設定

2 章で述べた設定及び前節で得られた設定を用いて 6 つの関数の最適化問題に取り組み，best Homogeneous GA，broad Heterogeneous GA，そして今回の提案手法である narrow Heterogeneous GA の性能比較を行った．migration はある場合とない場合の両方を実行し，各 100 回の実験を行ってその平均で評価した．島の数は $N = 16, 64, 128$ の場合を試し，逐次環境で並列環境をシミュレートした．各実行において $MAX\_NFE = 100000$ としたが，逐次環境で並列環境をシミュレートするため実際は $MAX\_NFE = 100000 \times N$ とした．そして，性能評価に用いる NFE は 実際の NFE/N として，並列環境を擬似的に再現した．並列化 GA の研究において，総 NFE(= 全体の集団サイズ) を分配すべき資源として考える研究もあるが [12]，今回は単に並列化資源をどのように分配するかのみに注目した．よって島ごとの集団サイズは異なり，その方が実用に近い環境だと考えられる．

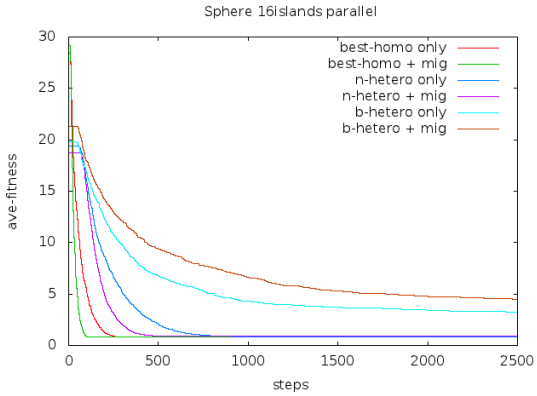
5.2　実験結果

結果を図 5.1，5.2，5.3 に示す．図を見ると，その挙動は 2 つのパターンに分類される．

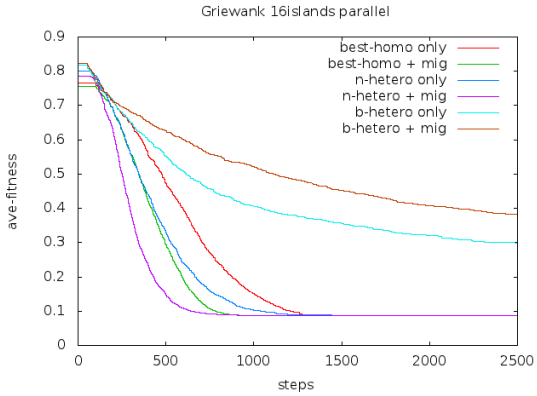
1 つは Sphere 関数に代表されるもので，best Homogeneous GA が一番よい性能を示し，続いて narrow Hetegeneous GA，broad Heterogeneous GA の順に性能が落ちていく (図 5.1(a)，5.1(b)，5.1(c))．他にも，Schwefel 関数 (図 5.2(d)，5.2(e)，5.2(f))，Rosenbrock 関数 (図 5.3(a)，5.3(b)，5.3(c))，Schwefel-1.2 関数 (図 5.3(d)，5.3(e)，5.3(f)) が同様の挙動を示した．

もう 1 つは Griewank 関数に代表されるもので，narrow Heterogeneous GA が一番よい性能を示し，続いて best Homogeneous GA，broad Heterogeneous GA の順に性能が落ちていく (図 5.1(d)，5.1(e)，5.1(f))．他に，Rastrigin 関数が同様の挙動を示した (図 5.2(a)，5.2(b)，5.2(c))．

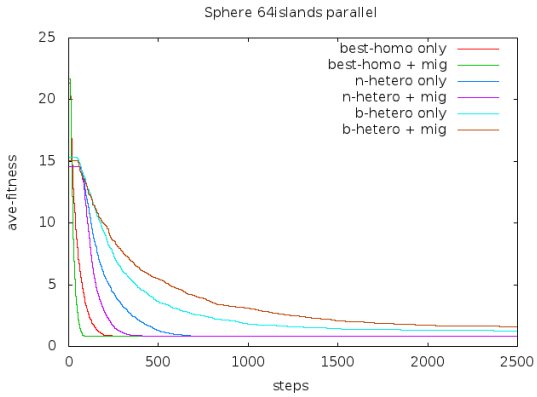
先行研究の枠組みである broad Heterogeneous GA が性能の面では一番劣っている．非効率的な島が多く生成されているという直感に反しない結果となった．



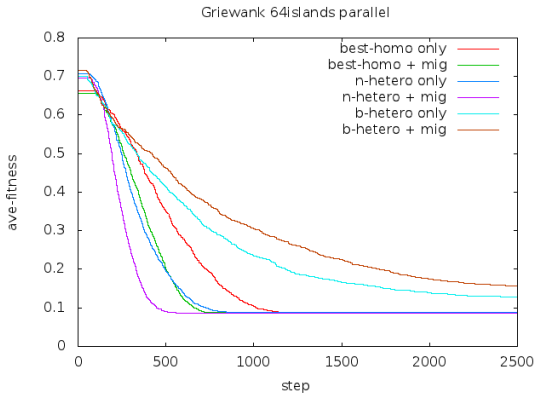
(a) Sphere, $N = 16$



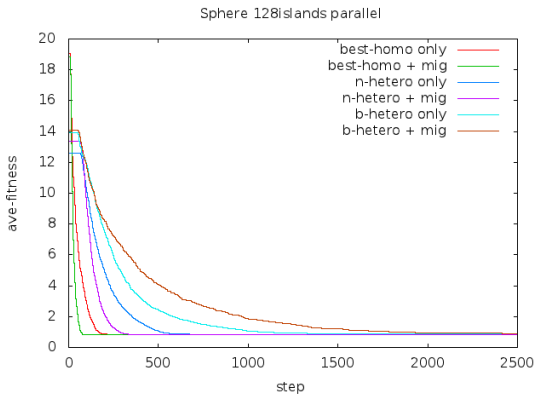
(d) Griewank, $N = 16$



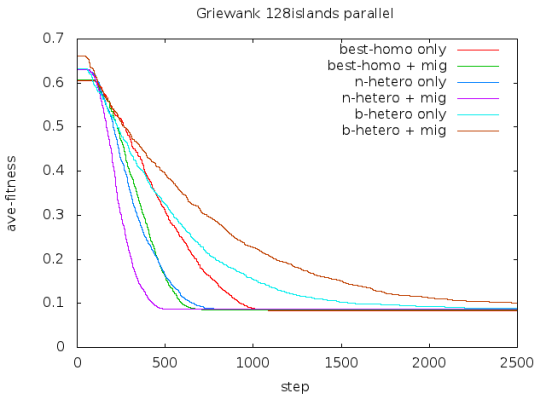
(b) Sphere, $N = 64$



(e) Griewank, $N = 64$

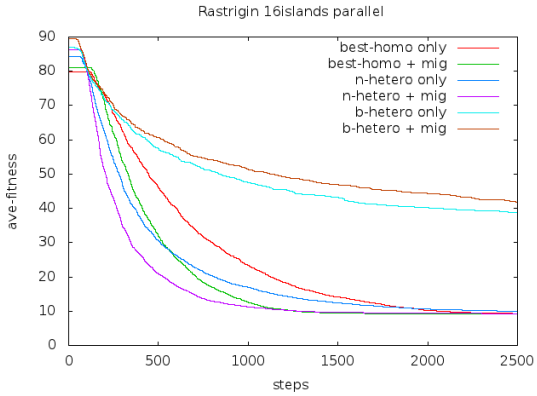


(c) Sphere, $N = 128$

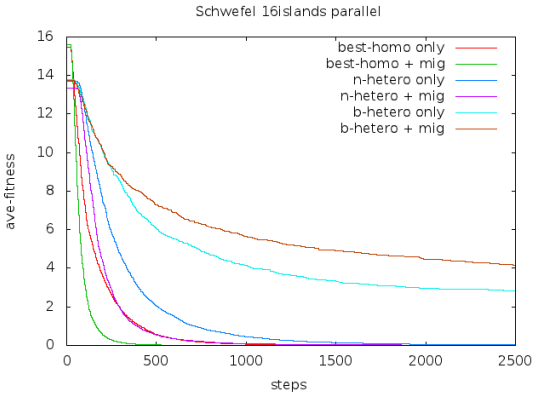


(f) Griewank, $N = 128$

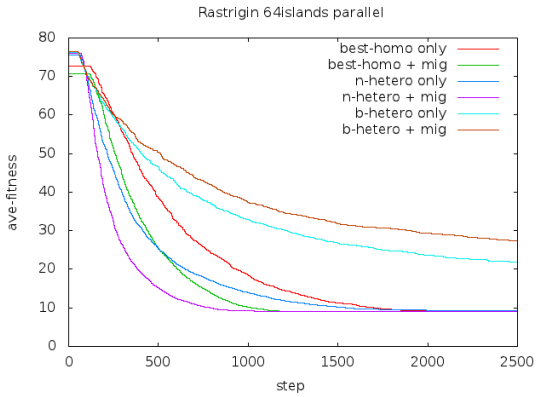
図 5.1　$N = 16, 64, 128$ と変えた場合における評価回数の経過に対する最良評価値の推移 (Sphere 関数 (f_1), Griewank 関数 (f_2))



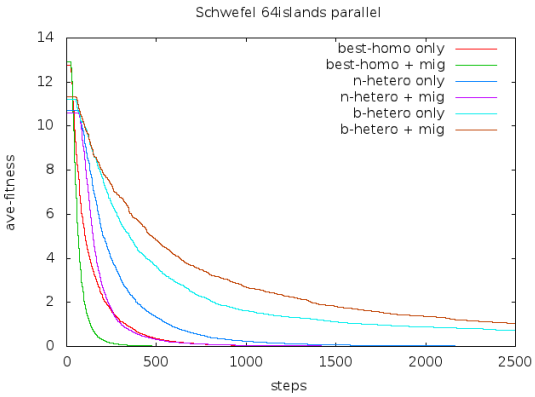
(a) Rastrigin, $N = 16$



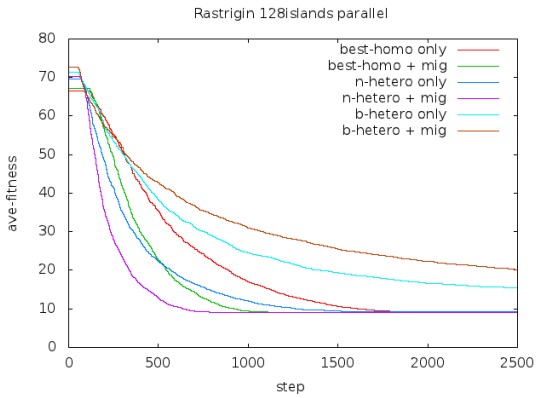
(d) Schwefel, $N = 16$



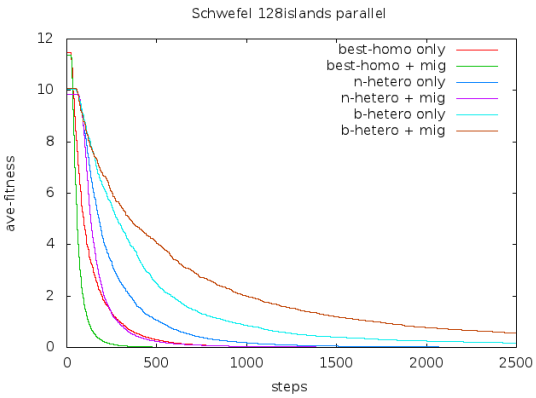
(b) Rastrigin, $N = 64$



(e) Schwefel, $N = 64$

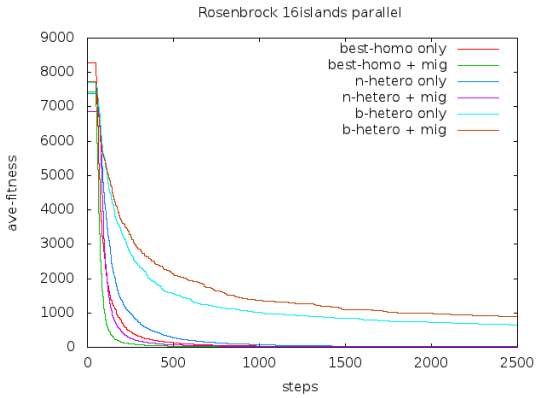


(c) Rastrigin, $N = 128$

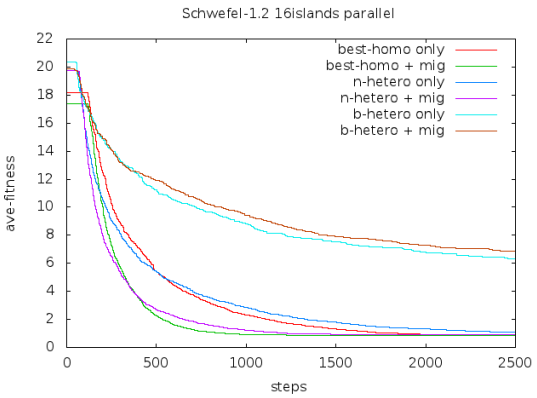


(f) Schwefel, $N = 128$

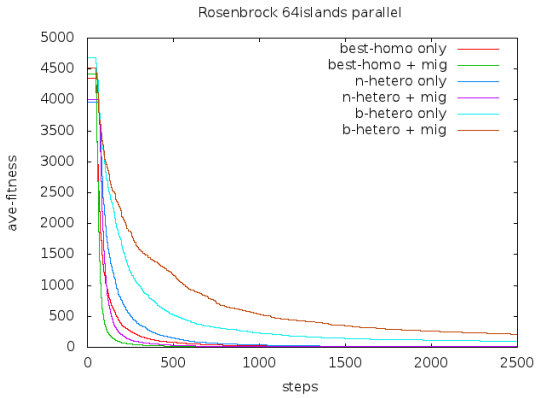
図 5.2　$N = 16, 64, 128$ と変えた場合における評価回数の経過に対する最良評価値の推移 (Rastrigin 関数 (f_3), Schwefel 関数 (f_4))



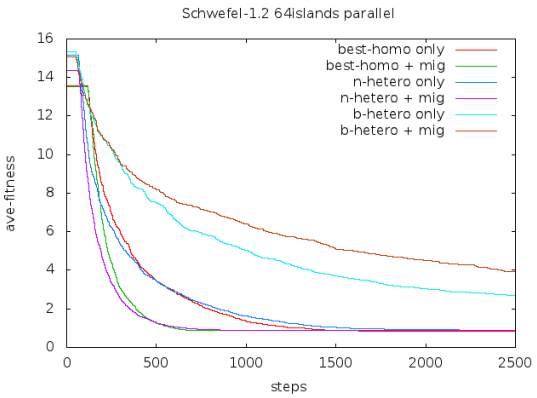
(a) Rosenbrock, $N = 16$



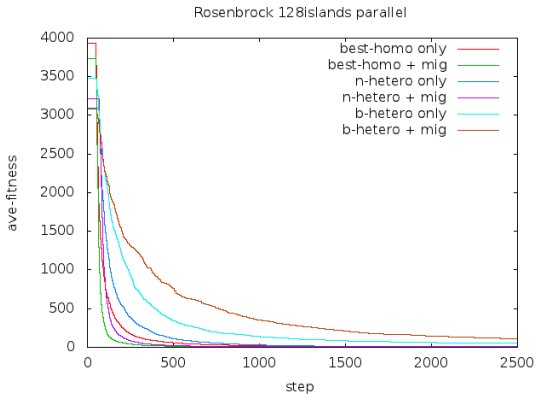
(d) Schwefel-1.2, $N = 16$



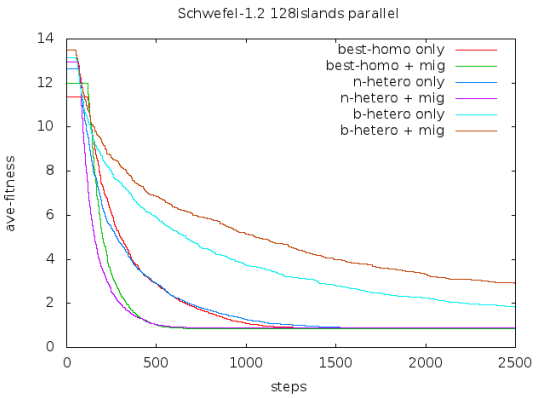
(b) Rosenbrock, $N = 64$



(e) Schwefel-1.2, $N = 64$



(c) Rosenbrock, $N = 128$



(f) Schwefel-1.2, $N = 128$

図 5.3　$N = 16, 64, 128$ と変えた場合における評価回数の経過に対する最良評価値の推移 (Rosenbrock 関数 (f_5), Schwefel-1.2 関数 (f_6))

6 Migration の効果の検証

6.1 各手法における migration 効果の比較検証

migration の効果について，narrow Heterogeneous GA と best Homogeneous GA では migration を行ったほうが探索性能は向上しているが，broad Heterogeneous GA では migraition を行ったほうが探索性能が落ちている様子が，前章のグラフから確認できる．パラメータ設定が不十分である場合には，migration を行った結果全体の多様性が下がることで，初期収束に陥ったと考えられる．

以上の結果から，migration の効果をはっきりさせるため，migration あり・なしの間の有意差検定を行った．今回，NFE について正規性を仮定できる理由はなかったため，ノンパラメトリックな検定である Wilcoxon の順位和検定を行った．有意水準は 5% と定めた．特に $N = 128$ の場合における結果を表 4 にまとめる．

表 4 の m は，有意差があった場合に，migration あり・なしのどちらが性能がよいかを表す変数である．$m =$ ‘ − ’ では migration は行わないほうが性能が良く，$m =$ ‘ + ’ では migration を行ったほうが性能が良くなることを示している．有意水準 5% の検定において，narrow Heterogeneous GA, best Homogeneous GA では migration を行った方が有意に性能がよい．これは，よいパラメータ設定や問題条件の下では migration は探索効率を高めるという従来の知見と一致する [12][8]．一方，broad Heterogeneous GA においては migration を行うと有意に性能が落ちる．これは，パラメータ設定が非効率的である場合は migration の効果が負に働く，ということを示している．

表 4　各手法における migration の有無に関する NFE の有意差検定 ($N = 128$)：表中の m は優意差があった場合の migration の効果を示す変数．$m =$‘+’： migration はプラスの効果，$m =$‘−’： migration はマイナスの効果を表す．

function	b-Heterogeneous		n-Heterogeneous		b-Homogeneous	
	p-Value	m	p-Value	m	p-Value	m
f_1	< 2.2e-16	-	< 2.2e-16	+	< 2.2e-16	+
f_2	2.279e-8	-	< 2.2e-16	+	< 2.2e-16	+
f_3	2.962e-5	-	< 2.2e-16	+	< 2.2e-16	+
f_4	1.746e-8	-	< 2.2e-16	+	< 2.2e-16	+
f_5	4.24e-4	-	< 2.2e-16	+	< 2.2e-16	+
f_6	1.258e-4	-	< 2.2e-16	+	< 2.2e-16	+

6.2 非効率的なパラメータ設定における migration 効果の検証

前節の結果を受けて，島の数を少なくし，より単純化した設定の下で，migration の効果を見る実験を追加で行った．

単純化のために，パラメータ設定は Homogeneous 環境，評価関数には Sphere 関数を用いて実験を行う．島の数は $N = 2, 4, 8, 12, 16$ を試したが，その効果が分かりやすい例として，$N = 4, 8$ の場合を以下に示す．

前節の結果より，best Homogeneous 環境では migration の効果はプラスであることは分かった．しかし，broad Heterogeneous 環境，すなわち，非効率的なパラメータ設定が含まれる環境では，migration の効果はマイナスになっている．

そのため本実験では，非効率的なパラメータ設定で固定した Homogeneous 環境を用意して，migration の効果を検証した．非効率的な場合のパラメータ値は，ランダムにパラメータ空間からサンプリングしてきた際に，挙動が悪化する境界付近から，予備実験を行うことにより設定した．予備実験の結果，$p = 100$，$\mu = 0.04$，$c = 0.7$ の設定を用いることとし，migration あり・なしのそれぞれの場合の探索性能を比較した．

境界付近を調べることで，さらに非効率的な設定の場合には今回以上に性能が落ちることが予測される．探索結果とその統計量を表 5 に示す．

表 5 を見ると，確かに migration ありの方が探索性能が劣っている．しかし，NEF の best や median に注目すると，best や median は migration の有無であまり違いがない．大きく違うのは worst の値で，migration を行ったときは探索が失敗する現象が見られる．この，migration を行うと探索が失敗するときがあるために，平均性能も劣っているのだと考えられる．

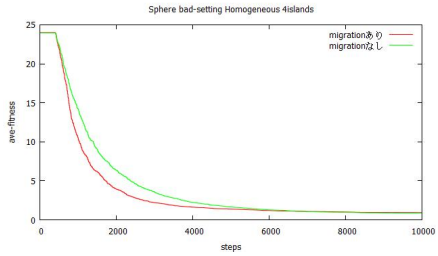
探索過程がどのように進むかを見るために，図 6.1 にその結果を示す．

図 6.1 を見ると，migration ありの方が収束が早まっていることが分かる．よって非効率的なパラメータ設定の下で migration を行った場合，基準に達していない，その時点での最良個体が，探索の初期段階で全ての島全体に広がり卓越することで，システム全体の多様性がなくなり，探索が停滞していることが考えられる．よって基準に達しない最良個体が改良されないまま評価回数が伸び，migration ありの場合は大きな分散値を取っているのではないかと考えられる．

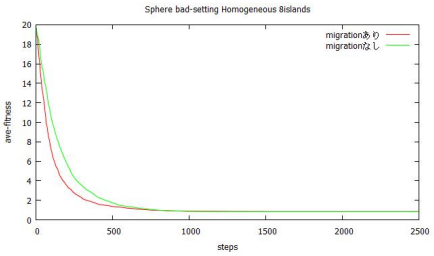
Sphere 関数という単峰性の問題であってもこのような初期収束の現象が確認されたため，broad

表 5　非効率的なパラメータ設定における探索結果：$p = 100$，$\mu = 0.04$，$c = 0.7$

島の数	migration	ave-NFE	std.	best	median	worst
4	なし	1779.1	495.7	382	1792	3154
	あり	3443.88	5969.8	430.25	1870	100000
8	なし	3148.92	803.4	1556	3020	5584
	あり	3358.64	3438.7	800	2732	100000



(a) bad Homogeneous $N = 4$



(b) bad Homogeneous $N = 8$

図 6.1　非効率的なパラメータ設定における探索過程 (Sphere 関数)

Heterogeneous で migration を行うことはリスクが高く，前節のような結果が出たものだと考えられる．本実験によって，broad Heterogeneous GA では migration の効果がマイナスに働くことを裏付けることができた．

6.3　Migration 効果のまとめ

以上の結果より，同じ Heterogeneous 環境の枠組みでも，ランダムサンプリングするパラメータ空間の取り方によって migration の有効性が異なることを示せた．narrow Heterogeneous GA では，migration による効果をプラスにし，性能の向上を達成することができた．一方，broad Heterogeneous GA では，基準値に達しない個体の拡散を migration が加速させるために初期収束が起こり，migration の効果が負に働くことを実証した．

今回は移住トポロジに双方向リングを用いたが，他に 2 次元グリッドや完全グラフなどの移住トポロジも考えられる．様々な移住トポロジを用いた場合における影響の調査は今後の課題であるが，broad Heterogeneous GA において，最も情報伝播が遅いリング状の移住トポロジで負の効果があるということは，更に情報伝播が速い場合には更なる負の影響が予測される．

7　島数が島モデル GA の探索に与える影響

3 つの枠組みのうち broad Heterogeneous GA の性能が相対的に劣っていることは分かったが，narrow Heterogeneous GA と best Homogeneous GA の間にどの程度の違いがあるのか，Wilcoxon の順位和検定により調べた．前章の結果より，migration ありの場合について比較を行えばよい．特に $N = 128$ の場合の結果を，表 6 にまとめる．表 6 中の SR (Success Rate) は，100 回の実験のうち，基準値まで探索できた回数を表す．NEF の欄には，100 回の実験の平均値を示す．

表 6 において，NFE に有意差があった場合には，性能が良い方を太字で示した．Schwefel-1.2 関数 (f_6) では，2 つの間に有意差は出なかったが，その他の関数については全て有意な差が出た．ベンチマーク問題のうち，半分に関しては best Homogeneous GA が優れているが，Griewank 関数 (f_2), Rastrigin 関数 (f_3) に関しては narrow Heterogeneous GA の方が優れている．これは，最良設定として用いたパラメータの組み合わせは島 1 つの場合の最良設定であり，島が複数に増えた場合の最良パラメータ設定とは異なることによるものと考えられる [9].

表 6 の結果を見ると，narrow Heterogeneous GA は十分考慮に値する枠組みだと思われる．本実験で比較対象とした best Homogeneous GA の設定には，異なる問題，異なる並列環境それぞれの場合で多大な労力が必要であり，実用上では，時間コスト・金銭コストなどの観点から，最良パラメータ設定を問題ごとに探していくのは現実的ではない．一方，narrow Heterogeneous GA は，探索性能は best Homogeneous GA に劣るものの探索成功率は同等であり，特に事前の労力を全く必要としない．本質的にパラメータフリーな narrow Heterogeneous GA が表 6 の性能を示すことは，narrow Heterogeneous GA が有用な手法であることを示していると考えられる．

さらに，図 5.1, 5.2, 5.3 において $N = 16, 64, 128$ の結果を順に見比べると，N を大きくするにつれて各アルゴリズム間の差が縮まっている様子が確認できる．図 7.1 に，narrow Heterogeneous GA が劣っていた 4 つの関数について，縦軸に best Homogeneous GA との NFE の差，横軸に島の数をとったものを示す．

図 7.1 を見ると，$N = 16 \rightarrow 64$ のときに差が大きく縮まり，$N = 64 \rightarrow 128$ においてもさらに差が

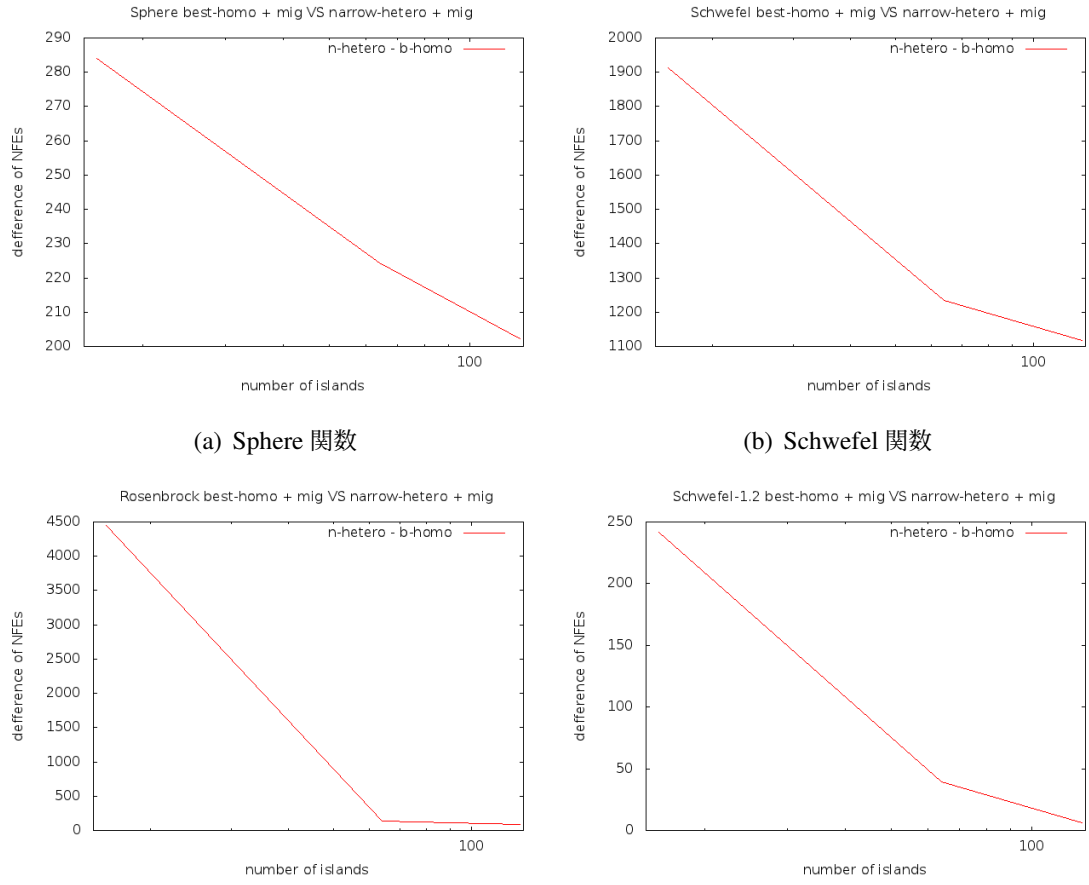
表 6　narrow Heterogeneous GA VS best Homogeneous GA

function	n-Heterogeneous		b-Homogeneous		p-Value
	SR	NFE	SR	NFE	
f_1	100	**270.59**	100	68.34	<2.2e-16
f_2	100	398.17	100	**571.34**	<2.2e-16
f_3	100	604.73	100	**872.86**	<2.2e-16
f_4	100	**1685.27**	100	567.42	<2.2e-16
f_5	100	**828.60**	100	745.40	2.01e-4
f_6	100	463.13	100	456.82	0.8649

縮まっていることが分かる．このまま N を大きくしていくと，徐々に best Homogeneous GA の性能に近づいていき，いずれは best Homogeneous GA の性能と同程度の性能が得られることが予想される．この現象は Gong と Fukunaga にも言及されていたが [5]，島の数が増えるにしたがって，どこかの島は最良パラメータ設定に近い設定を得て，探索を進めるからだと思われる．理論的にも，N を十分大きく増やせば，ランダム・ヘテロジニアス GA の性能が best Homogeneous GA の性能に近づくことが示せる可能性はあるが，それは今後の研究課題である．

この現象は，narrow Heterogeneous GA が，用意されたパラメータ空間から一様にパラメータをサンプリングするために起こっていると考えられる．すなわち，本実験のように連続的なパラメータ設定を用いた場合に期待できる現象であり，先行研究 [9] のように離散的なパラメータ設定を用いる場合には，この効果は期待できないように思われる．

特に Biazzini ら [2] のような大規模並列化の環境 ($N = 2^{16}$) では，narrow Heterogeneous GA の更なる性能向上が期待できる．



(a) Sphere 関数　(b) Schwefel 関数

(c) Rosenbrock 関数　(d) Schwefel-1.2 関数

図 7.1　narrow Heterogeneous GA が劣っていた関数における best Homogeneous GA との NFE の差

8 Heterogeneous 環境のロバスト性の検証

8.1 実験設定

本章では，最良パラメータ設定に関する知識や学習によるパラメータ調整などを仮定しない場合における narrow Heterogeneous GA の有効性を調べるため，一般によいとされているパラメータ値を用いた Homogeneous GA との比較を行った．一方，前章までの結果は，各問題に対する最適パラメータ設定を用いた best Homogeneous GA と，narrow Heterogeneous GA の比較であった．直感的には，推奨値を用いる Homogeneous GA よりも narrow Heterogeneous GA の方がロバストであると考えられる．Homogeneous GA のパラメータ設定は $p = 50$, $\mu = 0.01(= 1/L)$, $c = 0.6$ とし，この設定を standard Homogeneous と呼ぶ．Homogeneous GA のパラメータ値を変えた以外は全く同じ環境の下で，同様のベンチマーク問題を解き，Wilcoxon の順位和検定を行った．

8.2 実験結果

$N = 128$ の場合の結果を，表 7 に示す．

Sphere 関数 (f_1)，Griewank 関数 (f_2) では standard Homogeneous GA の方が有意に性能がよい．特に Griewank 関数は best Homogeneous GA よりも standard Homogeneous GA の方が性能がよいが，これは前述したように，N を大きくしたことにより最良パラメータ設定が変化した影響だと考えられる．

また，特に standard Homogeneous GA の Schwefel 関数 (f_4) における挙動は特徴的である．standard Homogeneous GA では，f_4 に関して探索成功率が極端に低くなっており，これはある一定のパラメータ設定を用いる場合の限界を示している．すなわち，Homogeneous 環境では，確率的な探索とはいえ，どの島もある程度似通った探索性能しか示すことができず，ある固定された設定ではどうしても解けない苦手な問題に当たってしまった場合，全く対応出来ないことを示している．

表 7　narrow Heterogeneous GA VS standard Homogeneous GA

function	n-Heterogeneous		s-Homogeneous		p-Value
	SR	NFE	SR	NFE	
f_1	100	270.59	100	**232.30**	9.582e-9
f_2	100	398.17	100	**374.12**	1.987e-3
f_3	100	**604.73**	100	672.80	3.632e-4
f_4	100	**1685.27**	0	100000	< 2.2e-16
f_5	100	**828.60**	100	1099.04	1.989e-15
f_6	100	463.13	100	455.26	0.687

また，NFL 定理 (No Free Lunch Theorem) より，そのような問題は理論的には必ず存在するため，Homogeneous 環境を用いることは，リスクが高い選択だといえる．

一方，narrow Heterogeneous GA は，どの関数も今回設定した基準で探索に成功している．すなわち，探索性能はときに standard Homogeneous GA に劣ることもあるが，パラメータ設定が原因で探索に失敗するリスクは低いと考えられる．よって，7 章の結果と合わせて，narrow Heterogeneous GA は十分ロバストな枠組みだと考えられる．

9 おわりに

本研究では，ランダム・ヘテロジニアス GA においてランダムサンプリングするパラメータ空間を無制限に取った場合，すなわち $\mu = [0.0, 1.0]$, $c = [0.0, 1.0]$ としたときに，migration の効果が負に働くことを実験的に示した．それを踏まえて narrow Heterogeneous GA の枠組みを提案し，migration による正の効果を得て，性能を向上させることに成功した．実際，narrow Heterogeneous GA のパラメータサンプリングの範囲は実験的に求めたのではなく，標準的に用いられているパラメータ付近に幅を持たせたものであることに注意して欲しい．突然変異率の推奨値は $\mu = 1/L$ であるため，$\mu \in [0.0, 2/L]$ とし，交叉率の推奨値は 0.6,0.7 などであるため，$c \in [0.3, 1.0]$ とした．これには，パラメータの推奨値という情報を除いて，事前の知識を全く仮定していない．また，このパラメータ値の推奨値という情報のみを持っていた場合における探索法として，narrow Heterogeneous GA と standard Homogeneous GA の枠組みの比較を行った．その結果，standard Homogeneous GA では探索が失敗するケースが見られたのに対して，narrow Heterogeneous GA は十分な性能を示した．これにより，narrow Heterogeneous GA は standard Homogeneous GA に比べて十分ロバストな枠組みだと考えられる．よって，少なくともまずシンプルな GA を試す場合には，Homogeneous GA ではなく narrow Heterogeneous GA を試すことが有効な手段であるといえる．

さらに，島の数を増やすにつれ narrow Heterogeneous GA と best Homogeneous GA の性能差が縮まっていく様子が観察された．これはランダムサンプリングにより，島の数が増大するほどよいパラメータ設定の島が生成される確率が高くなり，島全体のうち一部でもよいパラメータ設定の島があれば，有効に探索が進むからだと考えられる．更に島の数を増大させれば，narrow Heterogeneous GA の性能は best Homogeneous GA の性能に近づいていくと考えられる．そのため今後の課題としては，$N = 1000$ 以上の更なる並列化が挙げられる．今回の比較対象とした best Homogeneous GA の設定は，事前に数千回の実験を行って求めたものであり，多大なコストを費やさなければならない．例えば，個体の評価値を計算するためのシミュレーション 1 回の実行に 1 時間かかる場合，100000 個体を評価する GA を 1 回実行するのに 100000 時間かかる．これは，10000 コアの超並列計算機上で 10 時間かけてやっと実行できる計算である．このような状況では，事前に数千回の実験を行って最適パラメータ設定を求めることは，現実的ではない．一方 narrow Heterogeneous GA は，事前の努力をせずにその性能に近づいていくことのできる枠組みであり，実用的かつ十分に有効な戦略だといえる．

本研究では GA を用いて narrow Heterogeneous 環境の枠組みを試したが，島の数が増えるにつれ有効な島が得られ，最良パラメータ設定と同程度の性能が得られる現象は，制御パラメータを有するアルゴリズムに共通ではないかと考えられる．本研究で得られた narrow Heterogeneous GA における考察が GA 以外のアルゴリズムにも適用可能かどうかは，今後の研究課題である．

10　謝辞

同じ学科の友人や先輩方には，中間発表など様々な折に，有益なアドバイスや議論をしていただきました.

指導教員である福永先生には，研究のアイディアや技術的な面はもちろんのこと，それに留まらず研究者としての姿勢や気構えといった，研究生活のあらゆる面において、本当にたくさんのことを学ばせて頂きました。深く，感謝の念を申し上げたいと思います。

参考文献

[1] Enrique Alba and José M. Troya. A survey of parallel distributed genetic algorithms. *Complexity*, Vol. 4, No. 4, pp. 31–52, 1999.

[2] Marco Biazzini, Balazs Banhelyi, Alberto Montresor, and Mark Jelasity. Distributed hyper-heuristics for real parameter optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 1339–1346, 2009.

[3] Erick Cantu-Paz and David E. Goldberg. Efficient parallel genetic algorithms: Theory and practice. In *Computer Methods in Applied Mechanics and Engineering*, p. 2000. press, 2000.

[4] Alex Fukunaga, Hideru Hiruma, Kazuki Komiya, and Hitoshi Iba. Evolving controllers for high-level applications on a service robot: a case study with exhibition visitor flow control. *Genetic Programming and Evolvable Machines*, Vol. 13, No. 2, pp. 239–263, 2012.

[5] Yiyuan Gong and Alex Fukunaga. Distributed island-model genetic algorithms using heterogeneous parameter settings. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 820–827, 2011.

[6] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[7] Bernardo A. Huberman, Rajan M. Lukose, and Tad Hogg. An economics approach to hard computational problems. *Science*, Vol. 275, No. 5296, pp. 51–54, January 1997.

[8] Jörg Lässig and Dirk Sudholt. The benefit of migration in parallel evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 1105–1112, 2010.

[9] M. Miki, T. Hiroyasu, M. Kaneko, and K. Hatanaka. A parallel genetic algorithm with distributed environment scheme. In 1999 IEEE International Conference, editor, *Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings.*, Vol. 1, pp. 695–700, 1999.

[10] Fei Peng, Ke Tang, Guoliang Chen, and Xin Yao. Population-Based Algorithm Portfolios for Numerical Optimization. *IEEE Transactions on Evolutionary Computation*, Vol. 14, No. 5, pp. 782–800, October 2010.

[11] Petr Pospichal, Jiri Jaros, and Josef Schwarz. Parallel genetic algorithm on the cuda architecture. In *Proceedings of the 2010 international conference on Applications of Evolutionary Computation*, EvoApplicatons'10, pp. 442–451, Berlin, Heidelberg, 2010. Springer-Verlag.

[12] Z. Skolicki and K. De Jong. The importance of a two-level perspective for island model design. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 4623–4630, 2007.

[13] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. Technical report, Nanyang Technological University, Singapore, 2005.

[14] Shigeyoshi Tsutsui, Masayuki Yamamura, and Takahide Higuchi. Multi-parent recombination

with simplex crossover in real coded genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, Vol. 1, pp. 657–664, 13-17 July 1999.

[15] David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp. 67–82, 1997.

[16] E. L. Yu and P. N. Suganthan. Ensemble of niching algorithms. *Information Sciences*, Vol. 180, No. 15, pp. 2815–2833, August 2010.

[17] Li Feng Zhang and Chen Xi Zhou. Self organized parallel genetic algorithm to automatically realize diversified convergence. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–9, 2012.

[18] 池田心, 小林重信. GA の探索における UV 現象と UV 構造仮説. 人工知能学会論文誌, Vol. 17, pp. 239–246, nov 2002.

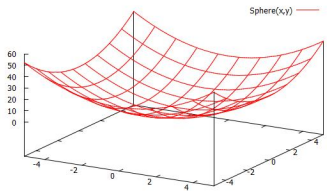
付録 A　評価関数

以下に，本研究で用いた評価関数の定義式と，その 2 次元の場合の図を示す．Rosenbrock 関数は 2 次元では単峰性の関数であるが，多次元になると多峰性の関数になる．その他の関数は，次元によってその特徴が変わることはない．

A.1　Sphere 関数

$$Sphere(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$$

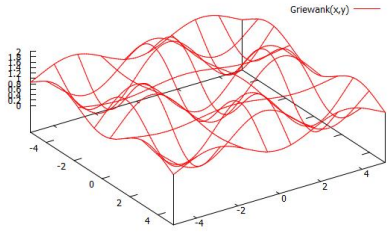
$$(-5.11 \leq x_i \leq 5.11)$$



A.2　Griewank 関数

$$Griewank(\mathbf{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \left(cos(\frac{x_i}{\sqrt{i}}) \right)$$

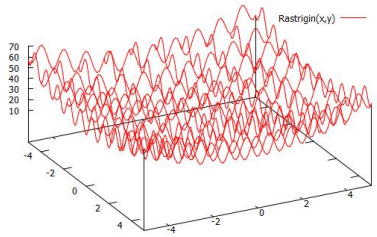
$$(-5.11 \leq x_i \leq 5.11)$$



A.3 Rastrigin 関数

$$Rastrigin(\mathbf{x}) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10cos(2\pi x_i) \right)$$

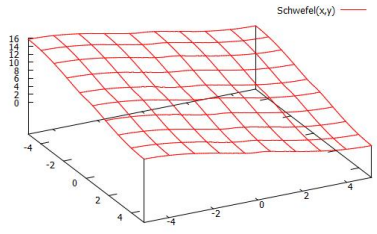
$$(-5.11 \leq x_i \leq 5.11)$$



A.4 Schwefel 関数

$$Schwefel(\mathbf{x}) = 3.9424320212n - \sum_{i=1}^{n} x_i sin\left(\sqrt{|x_i|}\right)$$

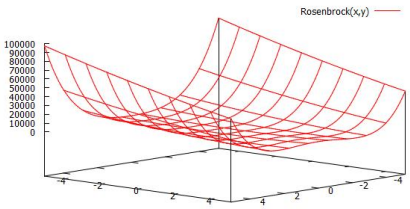
$$(-5.11 \leq x_i \leq 5.11)$$



A.5 Rosenbrock 関数

$$Rosenbrock(\mathbf{x}) = \sum_{i=1}^{n-1} \left(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right)$$

$$(-5.11 \leq x_i \leq 5.11)$$



A.6 Schwefel's Problem 1.2

$$Schwefel_{1.2}(\mathbf{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$$

$$(-5.11 \leq x_i \leq 5.11)$$