最適化問題における
免疫アルゴリズムの研究

Studies on Immune Algorithms
in Optimization Problems

2008 年 3月

森山 賀文

目次

概要

本論文は，免疫システム型遺伝的アルゴリズム（Genetic Algorithm with Immune System: GAIS）と免疫アルゴリズム（Immune Algorithm: IA）に関する研究の結果をまとめたものである．本研究では，経済の発展にともない大規模・複雑化した最適化問題のロバストな解を得るための一手法として唯一の大局的な最適解を含めた複数の局所的に優れた解を探索し，その解探索精度の向上と高速化を目的とする．生体の免疫システムに着想を得たGAISとIAは，その多様性のある解候補の生成機構により，探索空間に存在する複数の局所的な最適解を得ることができる．本研究では，特にIAによる最適化問題の複数解探索に関する実験を行い，その効果を考察した．

第 1 章

序論

1.1 研究の背景

最適化問題（Optimization Problems）とは，ある制約条件のもとで目的関数を最大化または最小化にする最適解を求める問題であり，実世界の多くの場面で見られ，工学，経済学を始めとする様々な分野で研究が行われている．この問題の厳密な最適解を求めるためには，問題の性質上，基本的に全探索を行う必要があり，問題の規模が拡大するにつれて評価すべき解の総数が爆発的に増加し，膨大な計算時間が必要となる．このように，現実的な時間内に最適解を求めることが困難となる問題の近似的な最適解（準最適解）を可能な限り高速に求める近似解法として山登り法（Hill Climbing Method）や焼き鈍し法（Simulated Annealing）[Kirkpatrick 83]，タブー探索（Tabu Search），遺伝的アルゴリズム（Genetic Algorithm: GA）などの確率的探索手法が考案されている．これらの手法は，単峰性の最適解探索に適しており，唯一の最適解またはその準最適解を得ることができる．

しかしながら，近年，計算機の処理速度の高速化や高性能化にともない，最適化の対象となる問題も大規模化，複雑化してきており，特に目的関数が複数存在する多目的最適化問題（Multi-objective Optimization Problem）では，複数の目的関数間にトレードオフの関係があるため各目的関数の重み付けが困難となり，厳密な最適解が設計者の意図するものではない場合がある．そのため，予め設計された問題の最適解（以下，大局的最適解とも呼ぶ）を求めるだけでは利用者側にとって不十分であり，利用者が複数の代替案の中から解を選択できるよう，複数の実現可能な優れた解（以下，局所的最適解とも呼ぶ）を探索できることが望まれる [森 97, 廣谷 06, 本間 05, 姫野 02]．例えば，建築構造の形態を求める問題では，力学的に最も優れた構造形態が必ずしも実用的な形状であるとは限らず，力学的に優れた大局的最適解だけではなく，デザインを考慮した実現可能な評価の優れた局所的最適解も重要となる場合がある [本間 05]．このように，実世界の最適化問題では厳密な最適解だけが要求されることは少なく，実行可能な複数の最適解が要求されることが多い．

一方で，脳神経系，遺伝適応系と並ぶ第三の生体システムとして，免疫システムが注目されている [森 93, 松村 99, 石黒 97]．免疫システムは，生体内に侵入する多種多様な抗原に対して特異的に反応する抗体を産生し排除する，という防御機構に着想を得た解探索手法であり，抗

原（問題）に適した抗体（解候補）を効果的に産生することにより解を得る [北野 95, 三宮 98]．特に，この免疫システムに着想を得た最適化問題の近似解法である，免疫システム型遺伝的アルゴリズム（Genetic Algorithm with Immune System: GAIS）[斉藤 02] や免疫アルゴリズム（Immune Algorithm: IA）[森 97] は，母集団内の多様性を維持しつつ解を探索することで，大局的に優れた唯一の最適解だけでなく局所的に優れた複数の解を探索することができる．

GAIS は，従来の GA の骨組みに免疫システムのアルゴリズムを一部導入した手法であり，解候補となる個体の新たな評価値として免疫度を導入している．免疫度とは，母集団内の各個体を相対的に評価する指標であり，局所的最適解に近い個体ほど高い評価を得る．この免疫度に基づいて母集団を進化させることで，大局的に優れた個体だけではなく複数の局所的に優れた個体が次世代の母集団として選択される可能性が高くなり，母集団の多様性を維持した複数の局所的最適解探索が可能となる．斉藤によると，この GAIS を画像探索に適用することで，対象画像内からテンプレート画像と一致性の高い複数の部分画像領域を同時に探索できることが示されている [斉藤 02]．しかしながら，この GAIS も，GA の持つ強い収束性質のために，対象とする問題や進化させる世代数によっては唯一の大局的最適解に収束してしまうという現象が起こり得ることを実験により確認した．GAIS は，これらの点に改善の余地がある．

一方で，IA は，未知の抗原に感染することにより新たな免疫が備わるという獲得免疫を工学的にモデル化した手法である．IA には，多種多様な抗原（問題）に特異的に反応する抗体（解候補）を産生する抗体産生機構と，解探索に有利な抗体を適切な量だけ生成する自己調節機構，さらに優れた抗体を記憶細胞（解の集合）として記憶する免疫記憶がある．これらのシステムにより，IA は探索空間内から複数の局所的に優れた解を探索することができる．この IA を多峰性関数に適用し，複数の局所的最適解探索が可能であることが示されている [森 97]．また，複数画像領域探索問題に適用し，GAIS よりも効率的に複数の部分画像領域を探索できることが報告されている [飯村 05]．しかし，次世代の抗体を生成する際に適用される交叉や突然変異といった遺伝的操作は，GA 同様，遺伝子型空間において適用されるため，遺伝子型空間と表現型空間との位相構造の差が大きく異なる場合，最適解の近傍を効率的に探索できない．また，量子化の精度によっては十分な精度の解を得られない場合や，冗長に探索空間を拡大してしまう場合がある．これらの点に検討の余地がある．

近年，量子重ね合わせ状態や量子干渉効果，量子もつれ状態などの量子力学的原理を利用した量子コンピュータが注目されている [Nielsen 00, 中山 04]．量子重ね合わせ状態を利用することで量子並列処理が可能となり，一度に大量の情報処理や計算を行うことができる．また，量子干渉効果によ効率的な解探索が可能となり，量子もつれ状態は量子非局所性[*1]の性質を持ち，量子転送に利用されている．量子もつれ状態は，従来の古典的コンピュータ上で実現しにくいものであるが，量子重ね合わせ状態や量子干渉効果は既存のアルゴリズムでも容易に取り入れることができる考え方であり，進化的アルゴリズムと融合した確率的な組

[*1] 非局所性（non-locality）とは，空間的，あるいは時間的に離れている粒子同士が，ある条件下では深い因果的関連性を持つという量子力学の性質である．

合せ探索アルゴリズムが提案されている．その一つとして，量子干渉効果を模擬した干渉交叉（Interference Crossover）[Narayanan 96] が提案されており，この干渉交叉を従来の GA や IA に導入し，巡回セールスマン問題（Traveling Salesman Problem: TSP）において探索性能の改善が可能であることが示されている [Narayanan 96, 中山 05, 中山 06a]．また，干渉交叉を DNA の螺旋構造に由来する螺旋交叉（Helical Crossover）として解釈する見方が中山らによって示されている [中山 06b]．しかしながら，これまでの GA や IA における螺旋交叉の実験では TSP のみを対象としており，TSP 以外の問題に対する螺旋交叉の効果について考察を行う必要があると考えられる．

これまで示してきたように，IA は最適化問題の有効な解探索手法であると考えられるが，IA を実装する場合，選択や交叉，突然変異などの遺伝的操作に関する知識や免疫システムに関する知識が必要であり，それらをシミュレートするための煩雑なコーディング作業をプログラム開発者が行う必要がある．コーディングの煩雑さを軽減する策としては，IA 用のクラスを開発しそれらをプログラム開発者に提供する，または IA 専用の DDL (Dynamic Link Library) を開発し提供するなどが容易に考えられるが，Java 言語が詳しくないと，IA 用のクラスタの使用や専用の DDL さえも IA が複雑なために難しいと思われる．そのため，IA に関する深い知識を持たない一般ユーザであっても IA を最適化問題の解法手段として容易に利用できる環境構築が必要であると考えられる．

1.2　研究の目的

以上のことを踏まえ，本論文では，問題の変化に対応可能な安定した解（ロバスト解）を得るための一手段として複数の局所的最適解を探索し，特に IA の最適化問題における複数解探索の高速化と高性能化を図る．また，一般ユーザに対しても IA の容易な利用を可能とする，IA の実装支援システムの開発を目的とする．

従来の GAIS は，免疫度に従った母集団の選択と淘汰により複数の局所的最適解を探索する手法であるが，問題の規模や進化させる世代数によっては唯一の大局的最適解へ収束するという現象が生じる．この母集団の多様化を目的として導入された免疫度は，目的関数の絶対的な評価値である適応度をもとに算出されるために適応度の評価が高い個体ほど免疫度による評価も高くなる可能性が高い．つまり，基本的な GAIS では，免疫度を基準とした個体の選択と淘汰により一時的に母集団の過剰収束を回避できても，GA の持つ強い収束性質により最終的には唯一の大局的最適解しか得られないと考えられる．そのため，適応度に帰着しない母集団の早期収束を抑制するシステムが必要であると考えられる．本論文では，母集団の多様性を長期的に維持することを目的として，獲得免疫の自己調節機構の一つであるサプレッサー T 細胞の働きを模倣し，過剰な類似個体の増殖を抑える抑制機構を導入した GAIS を提案する．この抑制機構は，一部の探索領域内に過剰な個体が存在する場合，解探索に有効な個体のみを残し他を排除することで母集団の多様性を維持する．抑制機構と，早期の局所的最適解探索を目的とした局所探索を GAIS に導入し，対象画像内から 2 次元的な姿勢自由度を持つ複数の部分画像領域を探索する複数画像領域探索問題に適用することで，

提案手法の有効性を明らかにする．

次に，IA は，複数画像領域探索問題において，GAIS と比較してより少ない抗体（個体）数で効率的に解探索が可能であることが示されている [飯村 05]．しかし，飯村らにより提案された複数画像領域探索手法では，抗体の遺伝子表現としてバイナリ型を採用しており，抗体の遺伝子型の空間と実際に探索を行う表現型の空間との位相構造が大きく異なるため，探索空間の連続性を考慮した交叉手法等の実装が困難であり，遺伝的操作による効率的な解探索ができない．また，従来の複数画像領域探索手法では，1 種類のテンプレート画像探索を目的としており，複数種類のテンプレート画像を探索するためにはテンプレート画像ごとに独立して探索を行う必要がある．そこで，抗体の遺伝子表現に整数型を用い，複数種類のテンプレート画像を識別するための遺伝子を新たに組込んだ IA を用いた複数画像領域探索手法を提案する．提案する手法は，遺伝子型の空間と表現型の空間を一致させることにより遺伝的操作による効率的な解探索が可能であり，またテンプレート画像を識別する遺伝子を用いて複数種類のテンプレート画像を同時に探索することで探索の無駄を省くことができる．提案手法を，書架画像の中から複数種類の巻，号画像を探索することで書籍を特定する問題に応用することで，その有効性を明らかにする．

ところで，Narayanan らにより量子系の干渉効果を模擬した干渉交叉が考案されており，干渉交叉が組込まれた GA を 9 都市の TSP に適用した結果，干渉交叉が探索世代数削減の観点で有効であることが示されている [Narayanan 96]．また，この干渉交叉を IA に導入することで，都市数の多い 101 都市の TSP に適用した結果，最適解の探索能力向上の観点からもその有効性が示されている [中山 05]．本論文では，文献 [中山 06b] の解釈のもと，干渉交叉を螺旋交叉と呼ぶことにし，TSP に対してその有効性が確認されている螺旋交叉法をジョブショップスケジューリング問題（Job-shop Scheduling Problem: JSP）に適用し，IA における螺旋交叉法が TSP に特化したものになっていないことを計算機実験を通して明らかにする．

次に，IA は生体の獲得免疫を模倣したアルゴリズムであるという性質上，免疫システムや遺伝的操作に関する知識が必須であり，また IA の実装には煩雑なコーディング作業が必要となる．そこで本論文では，そのコーディングの煩雑さを軽減し，一般ユーザでも IA を容易に利用できる環境構築を目的として，IA のための Immune 言語を提案する．Immune 言語を用いてプログラムを作成することで，免疫システム特有の処理に関するメソッドとだけでなく，GUI（Graphical User Interface）による視覚的な結果表示を行うメソッドも自動的に補完されるため，コーディングの煩雑さを軽減でき，また探索の結果を把握しやすい．Immunge 言語を用いて記述したプログラムと Java 言語を用いて記述したプログラムのファイルサイズとステップ数を比較し，またアンケートによる主観評価を行うことで，Immune 言語の有効性を明らかにする．

1.3　本論文の構成

本論文の概要を以下に示す．

第 2 章では，まず GAIS や IA の基礎となる GA と免疫システムについて概説し，基本的な GAIS，IA について述べる．

第 3 章では，従来の GAIS に免疫システムの抑制機構と優れた個体近傍の局所探索とを導入した GAIS について説明し，その有効性を明らかにするために行った実験およびその結果について述べる．

第 4 章では，実際に探索を行う空間と同じ整数型で遺伝子をコーディングし，また複数種類のテンプレート画像を識別するための遺伝子を新たに組込んだ IA について述べ，実験結果をもとに提案する方式の有効性について述べる．

第 5 章では，IA における干渉交叉法とその螺旋交叉的解釈について述べ，JSP への適用実験を通して IA における螺旋交叉法の探索性能を評価する．

第 6 章では，IA を実装する際のコーディングの煩雑さを軽減することを目的とした Immune 言語を提案し，その仕様とコーディング事例について述べ，アンケートによる主観評価を交えて Immune 言語の有効性について述べる．

第 7 章は結論である．本論文で得た知見をまとめる．

第 2 章

遺伝的アルゴリズムと免疫アルゴリズム

2.1 はじめに

近年，確率探索や学習，最適化の手法として，生物の情報処理に着想を得た情報システムの研究が盛んに行われている．遺伝的アルゴリズム（Genetic Algorithm: GA）や遺伝的プログラミング（Genetic Programing）[Koza 92] などの進化的計算（Evolutionary Computation）は生物が進化する過程をモデル化した手法であり，関数最適化や組合せ最適化など様々な問題に適用され，その有効性が示されている．また，脳神経系からヒントを得たニューラルネットワーク（Neural Network）[熊沢 98] はパターン認識等の問題へ応用されている．一方，遺伝適応系，脳神経系に並ぶ第三の生体システムとして免疫系が注目されている．免疫系の獲得免疫やネットワークをモデル化した幾つかの手法が提案されており，負荷割り当て問題 [森 93] や多峰性関数最適化問題 [森 97]，自律移動ロボット [石黒 97]，交渉支援 [松村 99]，パターン認識 [田島 00] など種々の問題に応用されている．本論文では，多様性のある母集団を形成することにより，探索空間内に存在する複数の局所的最適解を探索可能な免疫システム型遺伝的アルゴリズム（Genetic Algorithm with Immune System: GAIS）と免疫アルゴリズム（Immune Algorithm: IA）に着目する．

本章では，まず，GAIS と IA の基礎となる，GA と免疫システムについて述べる．その後，免疫システムの仕組みに着想を得た免疫システム型遺伝的アルゴリズムと免疫アルゴリズムについて述べる．

2.2 遺伝的アルゴリズム

2.2.1 遺伝的アルゴリズムの概要

遺伝的アルゴリズム（Genetic Algorithm: GA）とは，自然界における環境の変化に適応するための生物の進化のメカニズムを計算機上で模倣した手法である．GA は，1960 年代後半に Michigan 大学の Holland らによる適応システムの研究が基礎となっている [Holland 75]．1970 年代に入って DeJong により関数最適化問題を対象とした計算機実験が試みられ，その

後，1989 年に Goldberg によってアルゴリズムの枠組みが整理された [Goldberg 89]．以降，最適化や適応，学習のための方法論として急速に注目されはじめ，多方面に応用されるようになった．

GA では，最適化問題における解候補を個体（individual）と呼ぶ．各個体は遺伝子（gene）の集まりである染色体（chromosome）として表現される．染色体上の遺伝子を格納する場所のことを遺伝子座（locus）と呼び，また，各遺伝子の取り得る値の候補を対立遺伝子（allele）と呼ぶ．対立遺伝子には整数や実数，あるいは単なる記号などを用い，最も簡単なものは “0” と “1” の 2 値からなる．各個体が問題となる環境にどの程度適応しているかを表す評価値として適応度（fitness）を設定し，環境に適した個体ほど高い適応度を得る．各世代（generation）における個体の集団を母集団（population）と呼び，世代の更新は，母集団内で環境（問題）に対する適応度の高い個体の選択および適応度の低い個体の淘汰（selection）と，残された適応度の高い個体を親とした交叉（crossover）や突然変異（mutation）などの遺伝的操作（genetic operations）による子の生殖（reproduction）とによって行われる．

2.2.2　遺伝子型と表現型

GA や後述する免疫アルゴリズムで扱う情報は，遺伝子型（Genotype: Gtype）と表現型（Phenotype: Ptype）の二層構造からなる．図 2.1 に遺伝子型と表現型の概念図を示す．遺伝子型は染色体の構造にあたり，遺伝的操作を適用する対象となる．表現型は個体の形質や特性として実際に発現したものであり，環境内での構造を表す．この表現型から環境に応じた適応度が決定される．表現型から遺伝子型へ写像することをコーディング（coding）といい，逆に遺伝子型から表現型へ逆写像することをデコーディング（decoding）という．

個体の適応度による評価や選択，淘汰は実際に探索を行う表現型の空間において作用されるが，交叉や突然変異による生殖は遺伝子型の空間において作用される．そのため，遺伝子型の空間と表現型の空間の位相構造が大きく異なる場合，表現型の空間において互いに近い 2 つの親に対して交叉を適用しても子は両親の近傍に生成されるとは限らない．

2.2.3　遺伝的アルゴリズムの処理手順

GA の処理手順を図 2.2 に示し，各処理の詳細について述べる．

Step 1〔初期化〕

ランダムに遺伝子が決定された N 個の個体を生成し，初期世代（$t = 0$）の母集団 $P(0)$ を生成する．個体の遺伝子には目的関数の設計変数を決められた対立遺伝子の型でコーディングする．

Step 2〔評価〕

各個体の遺伝子から目的関数の設計変数をデコーディングし，現在の母集団 $P(t)$ における各個体 u の適応度 $fitness(u)$ を計算する．適応度は目的関数に対する評価値であ

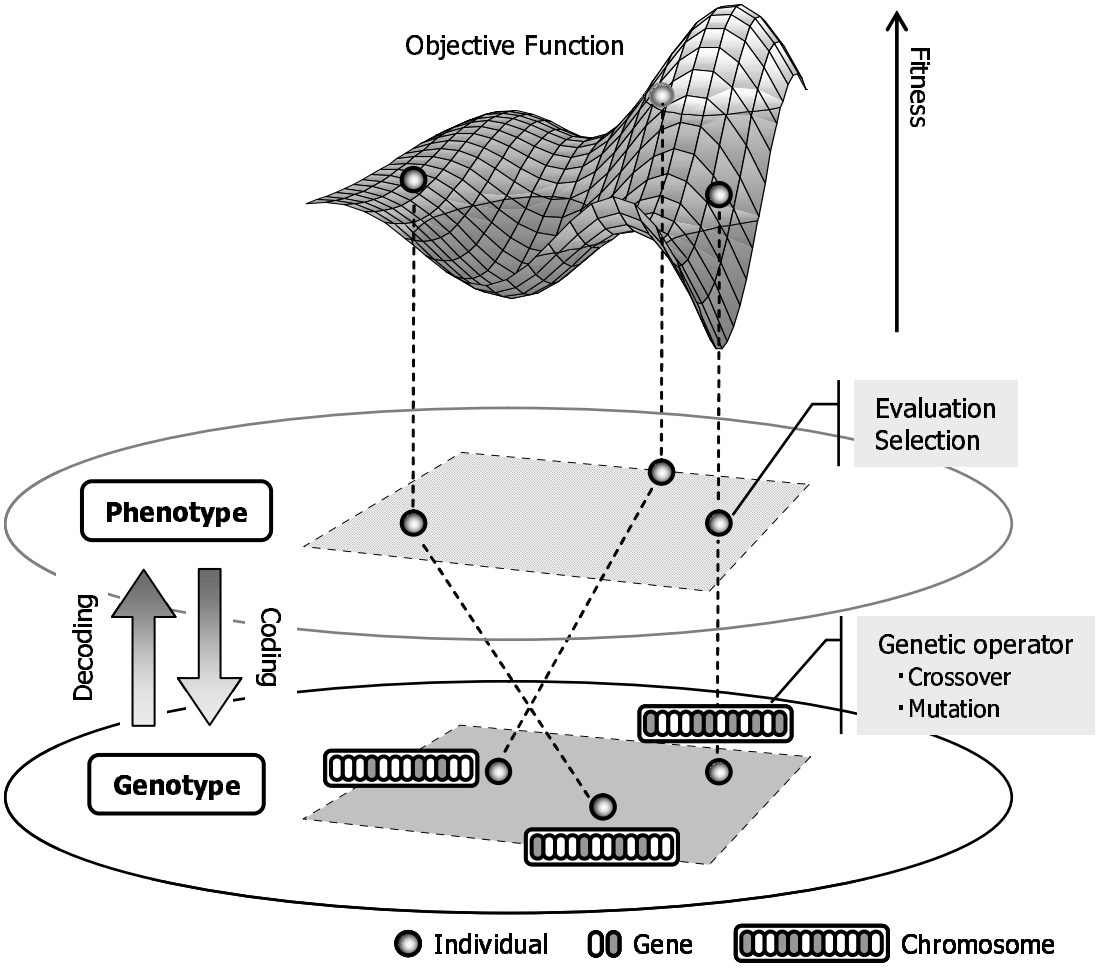


図 2.1 遺伝子型と表現型

り，適応度の高い個体ほど次世代に残り増殖される確率が高くなる．ただし，本論文では最適解に近い個体ほど高い適応度を与えるものとする．

Step 3〔終了判定〕

予め指定された進化の終了条件を満たしていれば Step 6〔進化終了〕へ，そうでなければ Step 4〔選択〕へ進む．

Step 4〔選択〕

$fitness(u)$ に応じて $P(t)$ から親個体群 $\dot{P}(t)$ を選択し，選択されなかった個体は淘汰される．

選択手法には，$fitness(u)$ に比例した確率で選択されるルーレット選択や，$P(t)$ からランダムに一部の個体群を抽出し，その中で最も $fitness(u)$ の高い個体を選択するトーナメント選択，$fitness(u)$ の順に個体をソートし順位に応じた生存確率を設定するランキング選択，$P(t)$ の中で $fitness(u)$ の高い個体を無条件で次世代に残すエリート戦略などがある．

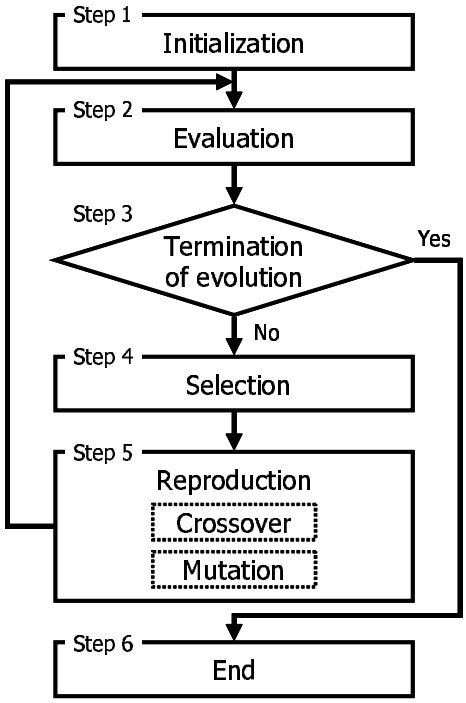


図 2.2 遺伝的アルゴリズムの処理手順

Step 5〔生殖〕

選び出された $\dot{P}(t)$ に対して交叉，突然変異を作用させ，次世代の母集団 $P(t+1)$ を生成する．交叉は生物の有性生殖を模倣した操作であり，基本的に 2 つの親にコーディングされた染色体の一部を組換えることで子を生成する．つまり，交叉を用いて生成される子は両親からの特徴を引き継ぐこととなる．その後，Step 2〔評価〕へ戻る．

Step 6〔進化終了〕

予め指定された終了条件を満足したため，進化を終了する．

2.2.4　遺伝的アルゴリズムの特徴

GA は，適応度の高い個体の選択と，それをもとにした増殖を繰返すことで，最適解が存在する可能性が高い領域へと次第に収束し，その領域を重点的に探索することで最適解またはその準最適解を得る．探索空間内に局所的に優れた解が複数存在する場合，解探索の序盤では複数の局所的な最適解付近に存在するが，母集団の進化とともに唯一の最適解付近へと収束する．一方で，初期の段階で母集団の多様性が失われると，探索の途中である一つの局所的な最適解に陥る場合があり，最適解を得ることが困難となる．そのため，GA では探索空間内に複数の局所的に優れた解が存在する場合であっても，唯一の大局的最適解またはある一つの局所的最適解しか得られない．母集団の多様性を維持する手法としては，母集団を複数のサブ母集団に分割し，各サブ母集団ごとの遺伝的操作と，一定世代間隔での移住を行う

島モデルによる分散化 [Tanese 89] などの手法がある[*1]．また，GA を用いて複数の局所的最適解を得るためには，母集団の多様性を維持するシェアリング [Horn 94] 操作が必要となる．

2.3　免疫システム

2.3.1　免疫システムの概要

前節で述べた GA は大局的な最適解の探索を目的としているため，複数の局所的最適解を得るためには種々の工夫が必要となる．それに対して，生体内に侵入してきた抗原に対応する免疫系の防御機構を工学的にモデル化した免疫システムは，多様な抗体の産生機構により複数の局所的に優れた解候補が得られるという性質があり，免疫系の獲得免疫やネットワークのモデル化 [森 93, 森 97, 石黒 97, 松村 99, 田島 00] や GA と免疫システムを融合する試みがなされている [Forrest 93, 北野 95, 三宮 98, 伊庭 99, 斉藤 02] ．以下では，まず生体の免疫系の仕組みについて説明し，その後，2.5 節で述べる免疫アルゴリズムや第 3 章で提案する抑制機構のモデルとなる獲得免疫について説明する．

免疫（immunity）[中島 97, 三井 04, 大内 03] とは，身の周りに存在する微生物や粉塵，花粉などによる外からの攻撃や，癌などによる内からの破壊から個体を守る生体の仕組みである．この免疫系の原点は，アメーバーのような単細胞が持つ，微生物の侵入を防ぐ細胞膜（守りの免疫）と微生物を取り込んで消化する食作用（攻めの免疫）である．人など様々な細胞が集団を作る多細胞の個体では，皮膚や粘膜が単細胞の膜の働きを引き継ぎ，食作用は消化管と食細胞（phagocyte）が担う．また，これらの細胞の働きを調節するリンパ球（lymphocyte）を併せて免疫系と呼ぶ．

自然免疫

病原微生物等の異物の侵入を防ぐ第一線のバリアとして皮膚や粘膜の物理的，化学的バリアがある．皮膚は何重もの細胞層（重層扁平上皮）で覆われ，病原体の侵入を防ぐ．また皮脂腺からの脂肪酸や汗の中の乳酸，あるいは塩酸により pH が低く抑えられた胃液などには殺菌作用がある．これらの物理的，化学的バリアを乗り越えて組織内に侵入してきた病原微生物に対しては，まず補体に代表される液性因子が働く．その後，細胞性因子である好中球やマクロファージなどの食細胞が，異物の特徴を認識するレセプターや補体などの液性因子の助けを借りて微生物を認識し，活性化され，貪食，排除を行う．これらの生体に先天的に備わり，迅速かつ非特異的に反応する防御機構を自然免疫と呼ぶ．

早期誘導免疫

ナチュラルキラー（natural killer） 細胞や $\gamma\delta$ 型 T 細胞，NK マーカーを持った T 細胞（NKT 細胞），CD5 陽性 B 細胞は感染の早期に反応してサイトカインや抗体を産生し，キ

[*1] 島モデルによる分散化を IA に適用した実験結果を付録の第 A 章に示す．

ラー細胞として感染細胞を傷害するとともに，続いて誘導される獲得免疫のタイプを決定する橋渡し的役的役割を有する集団が存在する．これらを早期誘導免疫と呼ぶ．

獲得免疫

大部分の微生物は，これまで説明してきた食細胞を中心とした非特異的に反応する防御機構によって排除され，感染症等の病気を引き起こすことは稀であるが，強い病原性を持つ微生物はこれらの防御機構に抵抗するエスケープ機構を有しており，組織への定着と増殖を始め，さらに血行性やリンパ行性に全身に広がる．このような微生物に対しては特異的に反応する防御機構が不可欠となる．T 細胞（T リンパ球）および B 細胞（B リンパ球）は微生物の抗原に対応する抗原レセプターを持ったものだけが活性化され，直接的に細胞傷害活性で感染細胞を排除したり，サイトカインや抗体を産生して自然免疫と共同で働いて，効率よく微生物の排除を行う．一部のリンパ球は記憶細胞となり，再び同じ病原体に感染した際にはこの記憶細胞が迅速に働き排除する．このように，未知の抗原に特異的に反応し，また新たな免疫として獲得する免疫機構を獲得免疫と呼ぶ．

2.5 節で述べる免疫アルゴリズムや第 3 章で提案する抑制機構は，この獲得免疫のシステム全体または一部を模倣した手法である．以下，獲得免疫の仕組みについて述べる．

2.3.2　獲得免疫の仕組み

獲得免疫には，あらゆる抗原に対応するため多様な抗体を産生する抗体産生機構と，適切な量の抗体を産生するために抗体産生の促進と抑制を行う自己調節機構，さらに抗原の排除に有効な抗体を記憶し再感染に備える免疫記憶というシステムがある．これらのシステムにより，未知の抗原に感染することで後天的に新たな免疫が備わる．「麻疹」や「おたふく風邪」などに一度かかると二度かからない，もしくはかかったとしても軽度で済むのはこの獲得免疫によるものである．図 2.3 に獲得免疫の概念図を示し，以下では獲得免疫の仕組みについて述べる．

免疫に関わるリンパ球やその他の血液細胞は全て血液幹細胞（hematopoietic stem cell）が分化したものである．ここでは特にリンパ球である B 細胞（B cell）と T 細胞（T cell）について説明する．血液幹細胞から分化したリンパ球幹細胞（lymphoid precursors）からは，骨髄（bone marrow）で B 細胞の前駆細胞が現れ，プレ B 細胞から B 細胞へと段階的に分化する．リンパ球幹細胞から前駆細胞へ分化する段階で抗体遺伝子の再編成が行われる．一方でリンパ球幹細胞から分化した胸腺リンパ球の前駆細胞は胸腺（thymus）へ移動し T 細胞となる．T 細胞はさらに，機能に応じてヘルパー T 細胞（helper T cell），サプレッサー T 細胞（suppressor T cell），細胞障害性 T 細胞（cytotoxic T cell，killer T cell）に分化する．

B 細胞は，抗原を認識したヘルパー T 細胞からのサイトカインと呼ばれる生理活性分子により活性化され，活発な細胞増殖と体細胞の突然変異を起こす．体細胞の突然変異を起こした B 細胞のうち，抗原と結合できる親和性（affinity）の高いものだけが生き残り，他は死滅する．ほとんどの B 細胞は骨髄に移住し抗体を産生するが，一部の B 細胞は抗原の侵入部

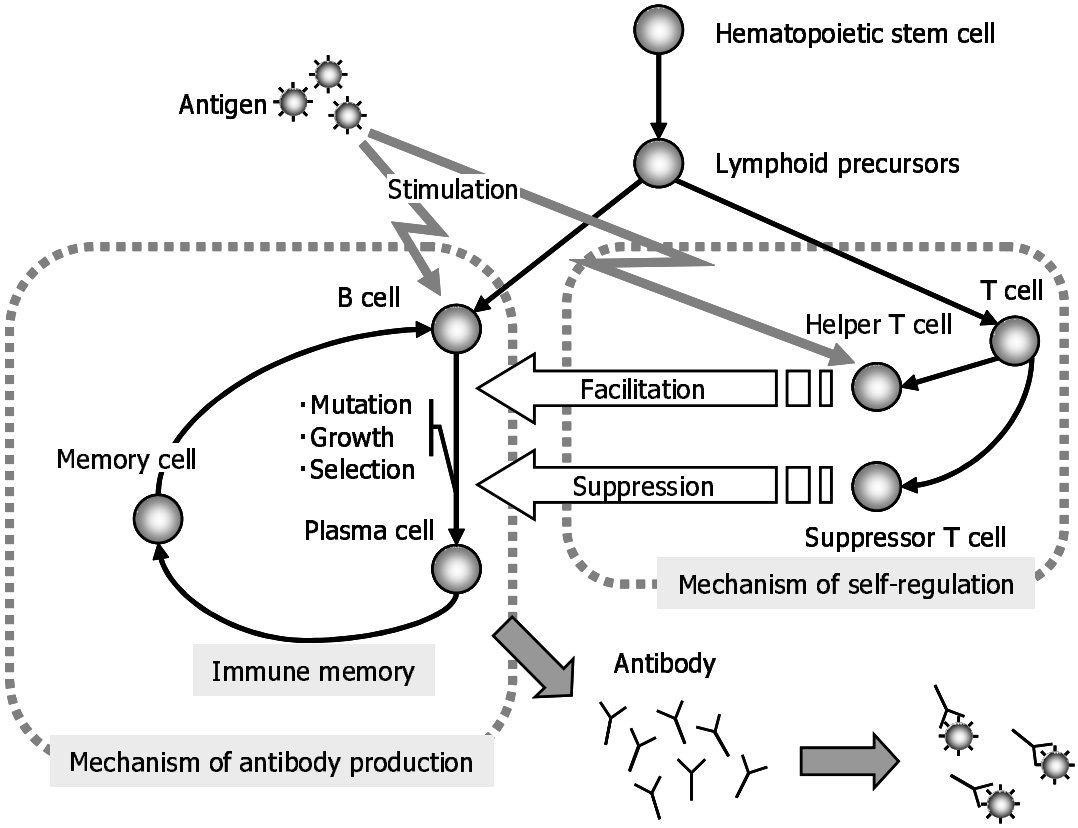


図 2.3 獲得免疫の概念モデル図

位へ遊走し，そこでさらに抗原による刺激とヘルパー T 細胞による活性化を受けて形質細胞（plasma cell）にまで分化し抗体を産生する．活性化し増殖された B 細胞や T 細胞の一部は長寿命の記憶細胞（memory cell）として保持される．このように，抗原を排除する有益な抗体を記憶することで，再感染に対して迅速に応答する．これを免疫記憶と呼び，獲得免疫の最も重要な特徴である．

抗原の刺激によって引き起こされる免疫応答は，活動が極大となった後，時間とともに減弱していく．これは，抗体などの免疫応答に関わる細胞の量が過剰となったり，質が異常となったりすることで，自己を標的として作用しないよう制御されるためである．この制御機構の一つに，免疫の後期に現れるサプレッサー T 細胞による抑制がある[*2]．ヘルパー T 細胞や B 細胞の働きを抑え，過剰な応答の発現を抑制する．

獲得免疫では，B 細胞を中心とした一連の分化と増殖，および遺伝子の再構成により様々な抗原に対応する多様な抗体を生成し（抗体産生機構），また T 細胞を中心とした抗体産生の促進と抑制により制御される（自己調節機構）．さらに抗原を排除する有益な抗体の一部は記憶細胞として記憶され，再感染した場合でも迅速に対応できる（免疫記憶）．

[*2]現在では，サプレッサー活性の多くはヘルパー T 細胞や細胞障害性 T 細胞の機能表現の一つであると考えられている（参考文献 [中島 97] 第 3 章 D 2 項参照）．

2.4　免疫システム型遺伝的アルゴリズム

2.4.1　免疫システム型遺伝的アルゴリズムの概要

GA は生物進化の過程に着想を得た大局的最適解を得る手法であり，大局的な解探索能力に優れている特徴を有する．近時，この GA は最適化手法の観点から盛んに研究されており，工学的課題などに広く適用されつつある．一方，免疫システムは，生体内に侵入する抗原に対応するために細胞遺伝子の再構成を行って抗原に対応する抗体を産生し抗原を排除する，という防御機構に着想を得た解探索手法であり，抗原（問題）に適した抗体（解候補）を産生することにより最適化問題の解を得る [北野 95, 三宮 98]．解の探索空間に多くの局所的最適解が含まれるような場合，一般的な GA では唯一の大局的最適解しか得られず，複数の局所的最適解を得るためには種々の工夫が必要となる．しかしながら，免疫システムには，多様な抗体の産生機構により解候補となる複数の優れた解が得られるという特徴がある．GA と免疫システムは，その起源は異なるものの，計算機に実装した場合，基本的な処理過程は類似する部分が多いため，双方の親和性は高いと言われている．そこで，免疫システムの優れた性質を GA に取り込むことを目的として，近年，GA に免疫システムを融合する試みが行われており，その一つとして GAIS[斉藤 02, 森山 05] がある．

この GAIS は，従来の GA の骨組みの中に，複数の局所的最適解探索に適した免疫システムのアルゴリズムを一部導入したものである．以下に GAIS の特徴を示す．

免疫度の導入

GA において個体を評価する指標は適応度のみであったが，GAIS では新たな評価値として免疫度（immunity）を導入する．適応度は目的関数に対する各個体の絶対的な優秀性を表す指標であり，突然変異などの遺伝的操作を適用しない限り変化しないのに対して，免疫度は母集団内における相対的な優秀性を表す指標であり，各世代の母集団の状況によって変化する．免疫度は，一部の個体群のランダム抽出によるエリート個体への適応度の累積処理により評価される．

免疫度による個体の選択

複数の局所的最適解を探索する場合，母集団の中で特に優れた個体だけでなく，多様性のある優れた個体群を次世代へと残す必要があるため，GAIS では免疫度を用いて個体の選択を行う．免疫度が高い個体を選択することで，適応度を用いた選択を行う一般的な GA と比較して，各世代の母集団の中で相対的に優れた多くの個体が次世代へ残る可能性が高くなり母集団の多様性を維持した解探索が可能となる．

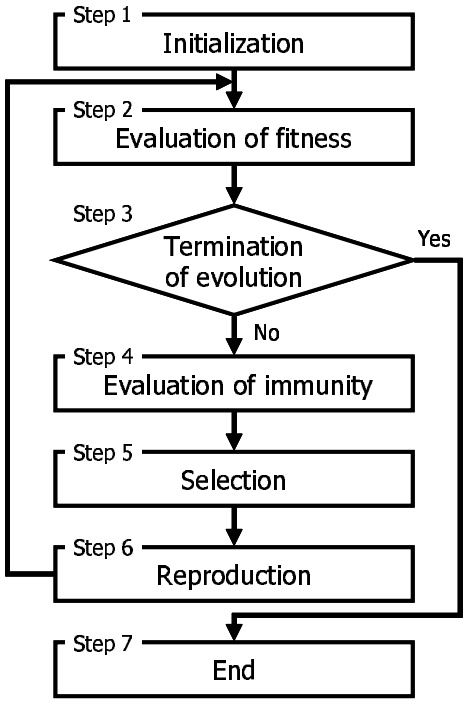


図 2.4 GAIS の処理の流れ

突然変異による個体の生殖

複数の局所的な最適解探索を探索する場合，異なる局所的最適解付近に存在する個体同士の交叉は解探索に有益でないと考えられるため，GAIS では GA における象徴的な遺伝的操作である交叉を行わず，基本的に突然変異によって淘汰された個体を補充する．GA での適応度の優れた個体間の交叉は，適応度の高い個体ほど最適解の近傍に存在すると考えられるため交叉によってより優れた個体が生成されることが期待できる．しかしながら，母集団の多様性を維持しながら探索を行う GAIS では，免疫度の高い個体は複数の局所的最適解に分散していると考えられるため，異なる局所的最適解付近に位置する個体同士の交叉を行った場合，両親とは掛け離れた位置に子が生成される可能性が高くなる．従って，GAIS では突然変異を主とした遺伝的操作とする．

2.4.2　免疫システム型遺伝的アルゴリズムの処理手順

図 2.4 に GAIS の処理手順を示し，各処理の詳細を述べる．

Step 1〔初期化〕

初期世代（$t = 0$）の母集団 $P(0)$ として，N 個の個体群をランダムに生成する．GA 同様，各個体の遺伝子は目的関数の設計変数を決められた遺伝子表現の型でコーディングされる．

Step 2〔適応度の評価〕

目的関数に対する各個体の優秀性を表す適応度による評価を行う．適応度は各個体の

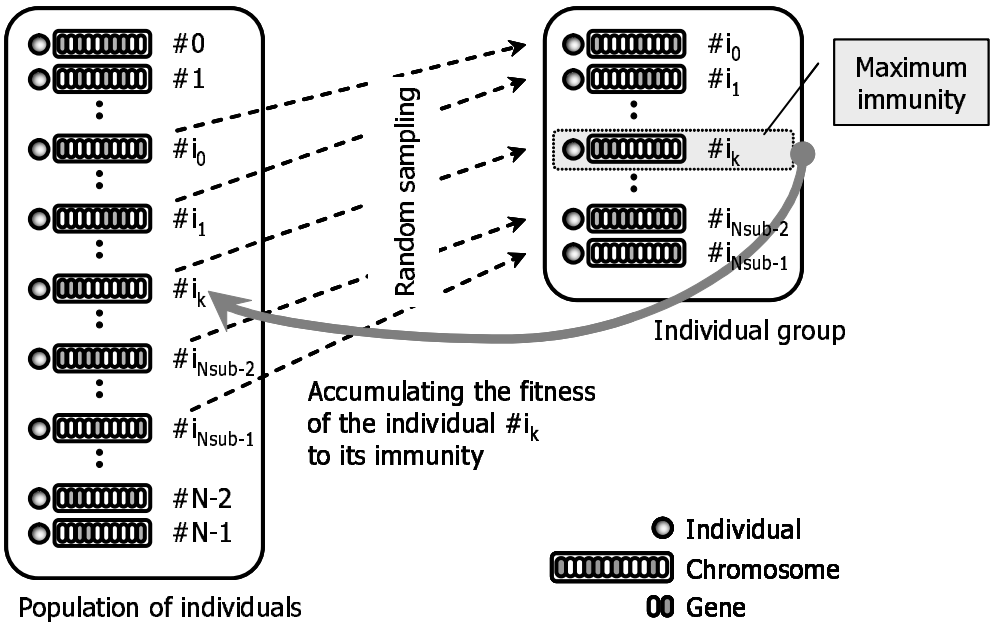


図 2.5 個体の免疫度評価

遺伝子をデコーディングすることにより得られる設計変数を用いて計算される．この適応度は各個体の免疫度の計算に用いられる．

Step 3〔終了判定〕

予め指定された進化の終了条件を満たしていれば Step 6〔進化終了〕へ，そうでなければ Step 4〔選択〕へ進む．

Step 4〔免疫度の評価〕

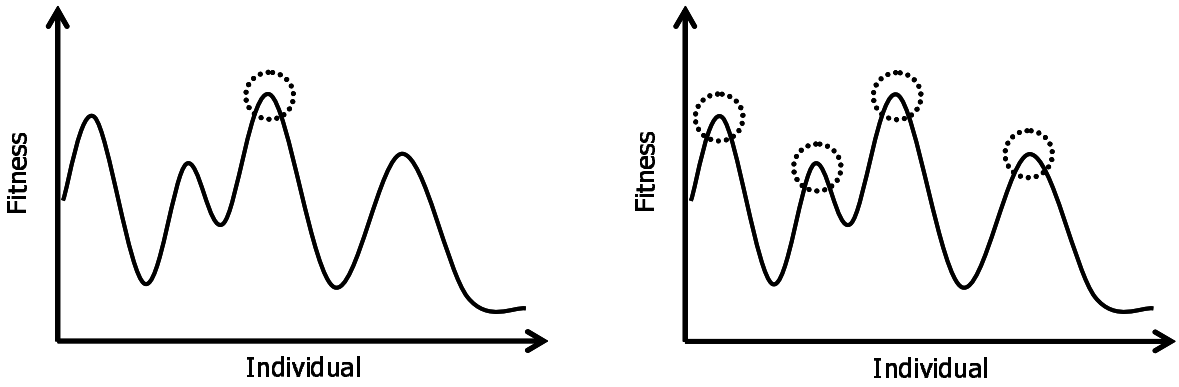
現在の母集団 $P(t)$ 内の各個体 u に対して免疫度 $immunity(u)$ を計算する．図 2.5 は，免疫度の累積処理を示したものである．$immunity(u)$ の初期値にはその世代における $fitness(u)$ を設定する．N 個の全個体の中からランダムに $N_{sub}(N_{sub} < N)$ 個の個体を選出し，その部分的な個体群 $P(t)'$ の中で最も高い免疫度を持つ個体 u' は，全集団の中で他の個体よりも相対的に優秀な個体である可能性が高いため，$immunity(u')$ に $fitness(u')$ を累積する．この処理を 1 世代の中で n_r 回繰返すことによって，集団内で相対的に優れた個体は徐々に高い免疫度を持つようになる．だだし，ここでの部分的な個体群はランダムに決定される．

Step 5〔選択〕

$P(t)$ から次世代に残す $immunity(u)$ の高い個体の選択と，$immunity(u)$ の低い個体の淘汰を行う．GA では $fitness(u)$ に応じて優れた個体の選択を行ったが，GAIS では $immunity(u)$ に応じて優れた個体の選択を行うことで，母集団の多様性を維持する．

Step 6〔生殖〕

淘汰されずに残った個体の中から遺伝的操作を適用させる親個体を選び出す．このとき，子の増殖は GA の特徴的な遺伝的操作である交叉を用いず，選び出された親個体



(a) Fitness　　(b) Immunity

図 2.6 適応度および免疫度における優秀性

に対して突然変異のみ作用させて次世代の母集団 $P(t+1)$ を生成する．その後，Step 2〔評価〕へ戻る．

Step 7〔進化終了〕

予め指定された終了条件を満足したため，進化を終了する．

2.4.3　免疫度による評価

各個体の免疫度は，母集団内の一部の個体群の抽出とその個体群の中で最も免疫度の高い個体に対して適応度を累積する処理を 1 世代の中で n_r 回繰返すことによって得られ，集団内で相対的に優れた個体は徐々に高い免疫度を持つようになる．これは，例を挙げて説明すると，学校のクラスの中で成績の総合点だけで判定するとトップのエリートしか選ばれないが，クラスの中には英語や数学に強いグループのトップや，喧嘩に強いグループのトップ，昆虫に詳しいグループのトップに免疫度を与えると，いろいろな難問に対処しやすいエリート集団ができると考えられる．ただし，ここでのグループ分けは，ランダムな選択に寄っている．

ここでは，個体の優秀性を評価する尺度として，GA における適応度に加えて免疫度を用いている．適応度が個体の集団内における各個体の絶対的な優秀性を表す尺度であるのに対し，免疫度は各世代における各個体グループの相対的な優秀性を表す尺度である．それらの関係は，図 2.6 で表される．図 2.6(a) は適応度を尺度として評価した場合である．適応度は個体集団内における各個体の絶対的な優秀性を表す尺度であるため，このような大局的最適解（唯一の点線の丸印）を探索する際に有効な尺度である．一方，図 2.6(b) は，免疫度を尺度とした場合である．免疫度は各個体グループの相対的な優秀性を表す尺度であるため，このような複数の局所的最適解（複数の点の丸印）を探索する際に有効な尺度である．

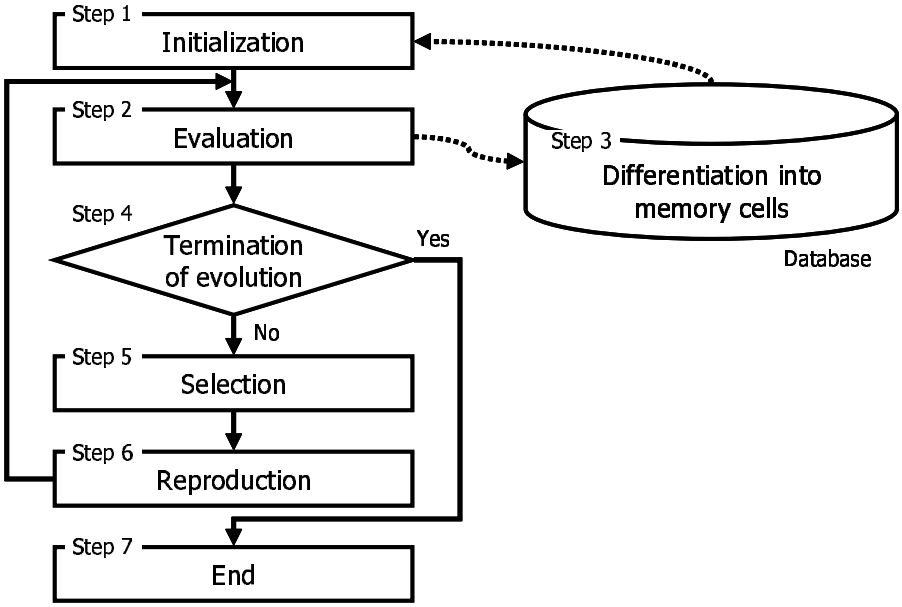


図 2.7 免疫的アルゴリズムの処理手順

2.5　免疫アルゴリズム

2.5.1　免疫アルゴリズムの概要

免疫システムの中でも未知の抗原に感染することにより後天的に得られる獲得免疫の仕組みを模倣した最適化問題の近似解法として，免疫的アルゴリズム [森 93] とその免疫的アルゴリズムを改良した免疫アルゴリズム（Immune Algorithm: IA）が考案されている [森 97]. 図 2.7 に示すように，免疫的アルゴリズムは，過去に獲得した，抗原の排除に有効な記憶細胞を初期母集団の生成に用いることで迅速に解を探索でき，また多様性のある母集団を生成することで局所的最適解へと陥らずに唯一の大局的最適解を探索できる．それに対して IA は，自己調節機構の一つであるサプレッサー T 細胞による過剰に増殖した抗体の産生を抑制する機構を導入することでさらに母集団の多様性を高め，記憶細胞を探索時間の短縮を目的とした優秀な解候補のデータベースとしてではなく，解探索により得られた解の集合と見ることで大局的最適解を含む複数の局所的最適解を得ることができる．

これまでに，IA を多峰性関数の最適化問題に適用し，大局的最適解を含む複数の局所的最適解をすべて探索する場合，GA よりも IA の方が効率的にすべての解を探索できることが報告されている [廣谷 06]．また，トラス構造の形状最適化問題に適用し，多様な構造の解を得ようとする場合，GA より IA の方が有効であることが示されており，それに加えて，構造物の形状をそのまま遺伝子として表現した実数型 IA の方がバイナリ型 IA よりも有効であることが示されている [本間 05]．さらに GA と免疫系を融合させ複数の解を発見できる GAIS[斉藤 02, 森山 05] と比較すると，交叉を主な遺伝的操作とする IA の方がより少ない抗体数で効率的に解を探索できることが示されている [飯村 05].

IA では，最適化問題における目的関数や前提条件を抗原（antigen），問題に対する解候補を抗体（antibody）とする．抗体は GA における個体（individual）に相当する．また，解の

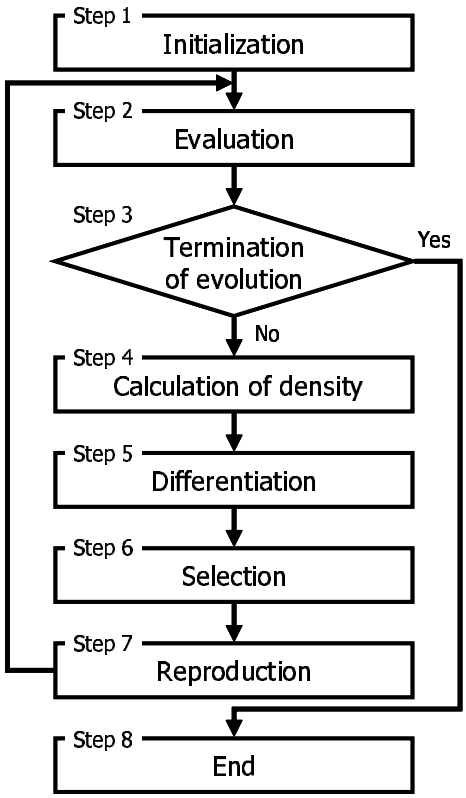


図 2.8 免疫アルゴリズムの処理手順

集合を記憶細胞（memory cell）とし，過剰に増殖した抗体を排除する細胞をサプレッサー T 細胞（suppressor T cell）とする．抗体と記憶細胞，サプレッサー T 細胞は，GA と同様に遺伝子（gene）で表現される．抗体群が形成する集合を母集団（population）と呼ぶ．

抗原と抗体の親和性，つまり目的関数に対する解候補の評価値は親和度（affinity）で表され，より解に近い抗体ほど高い親和度が与えられる．親和度は GA における適応度（fitness）に相当する．また，2 つの抗体間における遺伝子の類似性を類似度（similarity）で表す．各抗体の活性化の度合を示す指標として濃度（density）を用い，母集団内に類似した抗体が多く存在する抗体ほどより高い濃度が与えられる．次世代の抗体の増殖を促進，あるいは抑制するための指標として期待値（expectation）を用い，親和度が高く濃度が低い抗体，つまり，最適解の近傍に存在し母集団内に類似した抗体数が少ないほど高い期待値が与えられ増殖が促進される．

2.5.2　免疫アルゴリズムの処理手順

IA の処理手順を 図 2.8 に示す．

まず，遺伝子をランダムに決定した抗体を N_v 個生成し初期母集団を形成する．次に，各抗体の指標である親和度，類似度，濃度をそれぞれ計算する．濃度の高い抗体は記憶細胞とサプレッサー T 細胞へ分化し，抗原を排除する有効な抗体の記憶と過剰に収束した抗体の排除を行う．その後，期待値の計算と母集団内で親和度の低い抗体の淘汰を行い，交叉，突然

変異により淘汰された抗体の補充を行う．これらの処理が終了条件を満すまで繰返さることにより，大局的最適解を含む複数の局所的最適解を探索することができる．

以下に，各処理の詳細について述べる．

Step 1〔初期化〕

ランダムに抗体の遺伝子を決定し，初期世代（$t = 0$）の母集団 $P(0)$ を生成する．このとき，第 t 世代の抗体群が形成する母集団を $P(t)$ とする．

Step 2〔評価〕

各抗体 v について抗原に対する有効性を示す親和度 Φ_v を計算する．GA における適応度と同様，最適解に近い抗体ほどより高い適応度を得る．

Step 3〔終了判定〕

予め指定された進化の終了条件を満たしていれば Step 8〔進化終了〕へ，そうでなければ Step 4〔濃度計算〕へ進む．

Step 4〔濃度計算〕

まず，各抗体 v について他の抗体 w との類似度 Ψ_{vw} を計算する．抗体 v，w 間の類似度は，式（2.1）で計算し，類似した遺伝子を持つ抗体 v，w 間ほど高い類似度を得る．

$$\Psi_{vw} = 1 - D_{vw}. \tag{2.1}$$

D_{vw} は抗体 v，w 間の距離を表す．D_{vw} には，例えばバイナリ型のビット列を遺伝子とした場合，各遺伝子座について遺伝子（ビット）を比較し，異なるビット数（ハミング距離）を用いる．

次に，類似度 Ψ_{vw} が閾値 $T_{\pi 1}$ 以上の抗体 v，w を類似抗体とみなし，式（2.2）で定義される各抗体 v の濃度 Θ_v を求める．但し，N_v は総抗体数であり，π_{vw} は式（2.3）で定義される単位ステップ関数である．

$$\Theta_v = \frac{1}{N_v} \sum_{w=1}^{N_v} \pi_{vw}, \tag{2.2}$$

$$\pi_{vw} = \begin{cases} 1 & (\Psi_{vw} \geq T_{\pi 1}) \\ 0 & (\text{otherwise}). \end{cases} \tag{2.3}$$

Step 5〔分化〕

濃度 Θ_v が閾値 T_{Π} 以上の抗体 v を記憶細胞候補 v^* とする．すでに分化した記憶細胞数が上限数 N_m に達していなければ v^* をそのまま記憶細胞に分化し，N_m に達していれば記憶細胞候補 v^* との類似度 Ψ_{v^*m} が最も高い記憶細胞 m を記憶細胞候補 v^* で更新する．

また，記憶細胞候補 v^* をサプレッサー T 細胞 s へ分化させ，s との類似度 Ψ_{vs} が閾値 T_s 以上の抗体を母集団 $P(t)$ から消滅させる．消滅させた抗体と同数の抗体をランダムに生成し，総数が N_v となるよう補充する．

Step 6〔選択〕

母集団 $P(t)$ 内で親和度 Φ_v が低い $N_v/2$ 個の抗体を淘汰し，残った抗体について式（2.4）で定義される期待値 E_v を計算する．但し，$T_{\pi 2}$ は v と s との類似度 Ψ_{vs} により，期待値 E_v を規定する閾値である．N_s はサプレッサー T 細胞の総数であり，Ξ_{vs} は，式（2.5）で定義されるステップ関数である．

$$E_v = \frac{\Phi_v}{\Theta_v \sum_{w=1}^{N_v} \Phi_w} \prod_{s=1}^{N_s} (1 - \Xi_{vs}), \tag{2.4}$$

$$\Xi_{vs} = \begin{cases} \Psi_{vs} & (\Psi_{vs} \geq T_{\pi 2}) \\ 0 & (\text{otherwise}). \end{cases} \tag{2.5}$$

Step 7〔生殖〕

各抗体の期待値 E_v に従ったルーレット選択により $N_v/4$ 組の両親となる 2 つの抗体を決定する．決定された両親から交叉により $N_v/2$ 個の抗体を生成し，突然変異を適用することで，次世代の母集団 $P(t+1)$ を生成する．

Step 8〔進化終了〕

予め指定された条件を満足したため，進化を終了する．

2.6　まとめ

本章では，まず GA と免疫システムについて述べ，その後，免疫システムに着想を得た最適化問題の近似解法である GAIS と IA ついて述べた．GA は唯一の大局的最適解を目的とした手法であり，複数の局所的な最適解を得るためにはシェアリング等の処理を行う必要があるが，GAIS や IA などの免疫システムを模倣した手法は，母集団の多様性を維持しながら解探索を行うという優れた性質を有するため，特殊な処理を用いることなく複数の局所的最適解を得ることができる．GAIS は，各個体の母集団内における相対的な優秀性を表す免疫度による個体の選択を行うことで，早期収束を回避できる．IA は，多様性のある抗体産生機構と適切な量だけ抗体を産生できる自己調節機構により母集団の多様性を維持した解探索が可能である．また，優れた抗体を記憶細胞として記憶する免疫記憶により，抗体数が局所的最適解の数より多くなければならないという制約を受けない．

GAIS における免疫度は，局所的に優れた個体に対する適応度の累積処理により算出され，また局所的に優れた個体はランダムに選ばれた部分集合の中から決定されるため，解探索の序盤から中盤にかけては複数の局所的に優れた解を得ることができる．しかしながら，基本的に適応度の高い個体ほど高い免疫度を得る可能性が高いため，問題の規模や進化させる世代の長さによっては，唯一の大局的最適解に収束する場合がある [森山 03]．また，免疫度の評価には，より複数の局所的に優れた解に対して高い免疫度を与えるために，例えば 10,000 回という非常に多くの局所的に優れた個体の選択と親和度の累積処理を要し，さらに，突然変異を主たる遺伝的操作としているため，個体が収束するまでに時間がかかる．

前者の問題に関しては，親和度に帰着しない，類似個体の過剰な増殖を抑制する機構が必要であると考える．そこで，第 3 章において，獲得免疫の自己調節機構の一つであるサプレッサー T 細胞による抑制機構を組込んだ GAIS を提案する．また，後者の問題に関しては，前節で説明した IA が，複数の部分画像領域探索問題において，GAIS よりも少ない抗体（個体）数で解探索を行うことにより，約 1/10 の探索時間で同等以上の解探索率が得られることが示されている [飯村 05]．そこで，第 4 章以降では，IA の最適化問題における複数解探索の高速化と高性能化を目的としたいくつかの手法を提案し，数値実験によりその有効性を確認する．

第 3 章

抑制機構を有する免疫システム型遺伝的アルゴリズムによる画像探索法

3.1 はじめに

2.4 節で述べた GAIS は，GA の骨組みの中に複数の局所的最適解探索に適した免疫システムのアルゴリズムを一部導入し，個体の相対的な評価値である免疫度に応じて淘汰した個体を増殖することによって複数の局所的最適解を探索することができる．しかしながら，GA がもつ強い収束性質のため，進化の世代の長さや問題の規模によっては唯一の大局的最適解に収束してしまう現象が起こる場合がある．

そこで，本章では，獲得免疫の自己調節機構の一つである，サプレッサー T 細胞による B 細胞の過剰な応答を抑えるシステムを模倣した抑制機構を GAIS に導入し，対象画像内から 2 次元的な姿勢自由度を持つ複数の部分画像領域を探索する手法を提案する．抑制機構は，脊椎動物の持つ獲得免疫の仕組みを参考にした最適化問題を解くアルゴリズムである IA において一般的に用いられているが，ここではその抑制機構を，従来の GA に免疫度という評価指標を加えた GAIS に導入し，その効果を検証する．提案手法における抑制機構が個体の多様性を維持し，長い世代進化を続けても唯一の大局的最適解に収束することなく，複数の局所的最適解を探索できる．

以下，提案する抑制機構を有する GAIS の詳細と，その有効性を明らかにするために行った実験とその結果について述べる．

3.2 抑制機構を有する GAIS による画像探索法

提案手法では，従来の GAIS に，サプレッサー T 細胞による個体の産生を抑制する機構と局所探索を導入することにする．図 3.1 は，提案手法の処理の流れである．従来の GAIS との差異は，類似した個体の過剰な増加を抑制する処理が「個体の増殖」の後に追加されている点と,「優れた個体の選択」の段階で局所探索が加味されている点である．以下に，抑制機構を有する GAIS を用いた複数の画像領域を探索する手法について詳しく述べる．

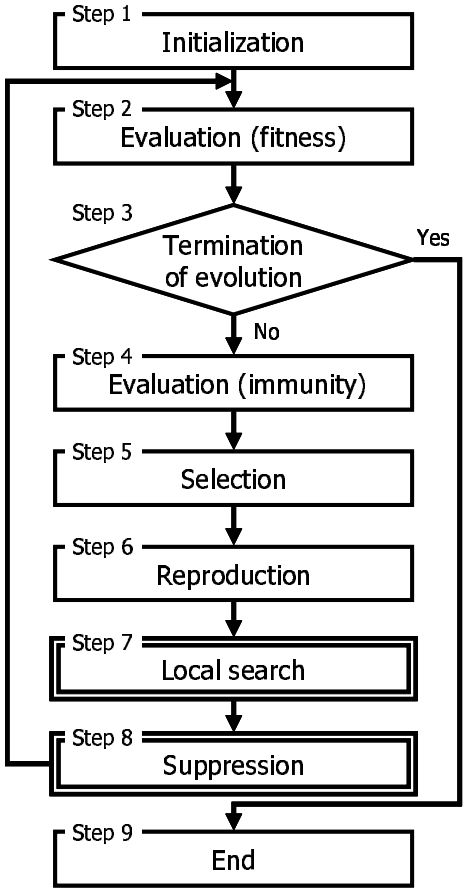


図 3.1 抑制機構を有する GAIS の処理の流れ

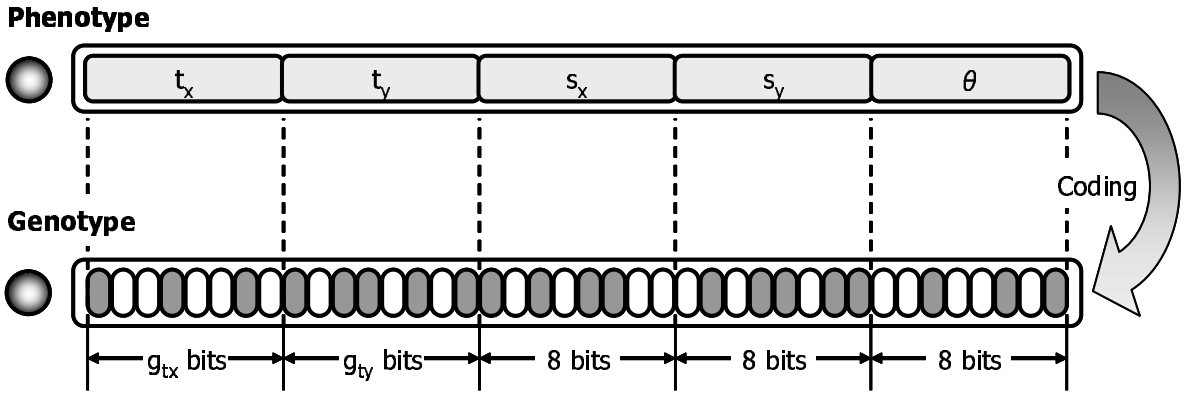


図 3.2 染色体のコーディング

3.2.1　初期個体群の生成

GAIS を画像探索へ適用するために，まず，各個体のコーディング手法について説明する．個体の染色体には，探索すべき部分画像 $\mathcal{I}'$ の姿勢パラメータをコーディングする．図 3.2 に示すように，部分画像 $\mathcal{I}'$ を囲む矩形の位置 (t_x, t_y)，縮尺 (s_x, s_y)（s_x は x 軸方向の縮尺，s_y は y 軸方向の縮尺），傾き θ を決められたビット数で表現し，これら 5 種類の姿勢パラメータ $(t_x, t_y, s_x, s_y, \theta)$ を 1 次元的に並べた配列を染色体とする．ただし，部分画像 $\mathcal{I}'$ を囲む矩

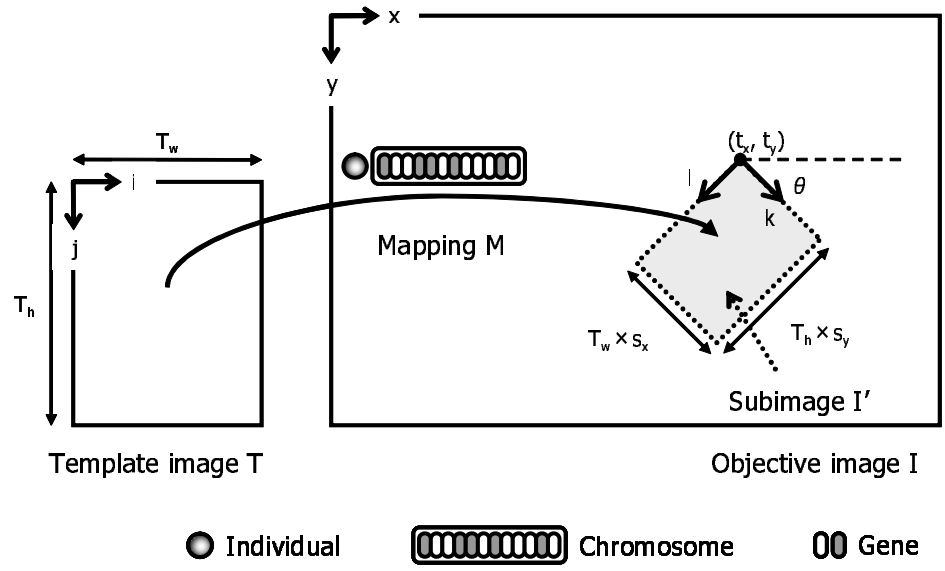


図 3.3 個体の適応度評価

形の位置を表す t_x と t_y のビット数 g_{t_x} と g_{t_y} は，探索対象画像 $\mathcal{I}$ の大きさにより決定される．また，初期個体群はランダムに N 個生成するものとする．

3.2.2　適応度の評価

図 3.3 に示すように，各個体の姿勢パラメータ $(t_x,\, t_y,\, s_x,\, s_y,\, \theta)$ に従い，テンプレート画像 $\mathcal{T}$ を探索対象画像 $\mathcal{I}$ にマッピングする．マッピングされた領域の部分画像 $\mathcal{I}'$ とテンプレート画像 $\mathcal{T}$ を比較し，全ピクセルについて RGB 成分ごとにカラー値の差 δ_C を計算し，それら全ピクセル分の合計を求める．このとき，$|\delta_C|$ が小さいものほど解に近い個体であると考え，この個体の適応度が高くなるように RGB 各成分の最大値である $C_{\max} = 255$ から $|\delta_C|$ を引いた値を用い，適応度を評価する．

$\mathcal{T}_w \times \mathcal{T}_h$ 画素からなるテンプレート画像 $\mathcal{T}$ と，そのテンプレート画像 $\mathcal{T}$ が姿勢パラメータ $(t_x,\, t_y,\, s_x,\, s_y,\, \theta)$ によって探索対象画像 $\mathcal{I}$ 内にマッピングされた領域である部分画像 $\mathcal{I}'$ との適応度 *fitness* は，式（3.1）で与えられる．

$$fitness = \frac{\displaystyle\sum_{i=1}^{\mathcal{T}_w}\sum_{j=1}^{\mathcal{T}_h}\sum_{C=\{R,G,B\}} \mathcal{W}(i,j)\cdot(C_{\max} - |\delta_C(i,j)|)}{\displaystyle\sum_{i=1}^{\mathcal{T}_w}\sum_{j=1}^{\mathcal{T}_h} \mathcal{W}(i,j)\cdot 3\cdot C_{\max}}, \tag{3.1}$$

ただし，

$$\begin{aligned}\delta_C(i,j) &= \mathcal{I'}_c(k,l) - \mathcal{I}_c(i,j) \\ &= \mathcal{I}_c(x,y) - \mathcal{I}_c(i,j) \\ &= \begin{pmatrix}\mathcal{I}_R(x,y)\\ \mathcal{I}_G(x,y)\\ \mathcal{I}_B(x,y)\end{pmatrix} - \begin{pmatrix}\mathcal{T}_R(i,j)\\ \mathcal{T}_G(i,j)\\ \mathcal{T}_B(i,j)\end{pmatrix},\end{aligned}$$

$$\begin{aligned}\begin{pmatrix}x\\ y\\ 1\end{pmatrix} &= \mathcal{M}\begin{pmatrix}i\\ j\\ 1\end{pmatrix} \\ &= \begin{pmatrix}s_x\cos\theta & -s_y\sin\theta & t_x\\ s_x\sin\theta & s_y\cos\theta & t_y\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}i\\ j\\ 1\end{pmatrix},\end{aligned}$$

である．

ここで，$\mathcal{W}(i,j)$ は重み画像 $\mathcal{W}$ 内の座標 (i,j) における重みである．$\{\mathcal{I}_R(x,y)$，$\mathcal{I}_G(x,y)$，$\mathcal{I}_B(x,y)\}$，$\{\mathcal{I'}_R(k,l)$，$(\mathcal{I'}_G(k,l)$，$\mathcal{I'}_B(k,l)\}$，$\{\mathcal{T}_R(i,j)$，$\mathcal{T}_G(i,j)$，$\mathcal{T}_B(i,j)\}$ はそれぞれ探索対象画像 $\mathcal{I}$，部分画像 $\mathcal{I'}$，テンプレート画像 $\mathcal{T}$ 内の座標 (x,y)，(k,l)，(i,j) における R 値，G 値，B 値である．なお，行列 $\mathcal{M}$ は，姿勢パラメータ $(t_x, t_y, s_x, s_y, \theta)$ より与えられる同次座標（homogeneous cordinate）表現されたアフィン変換である．

次に，重み画像 $\mathcal{W}$[長尾 02] について説明する．図 3.4 に示すように，例えば探索すべき文字「2」の黒い部分がテンプレート画像 $\mathcal{T}$ 中に占める割合が少ない場合を考える．このような場合，安易に考えてしまうと，全画素が白のみの部分画像 $\mathcal{I'}$ に対してもテンプレート画像 $\mathcal{T}$ と一致する画素が多いため，比較的高い適応度を得てしまうことになる．このような不具合を解消するため，ここでは重み画像 $\mathcal{W}$ を導入する．重み画像 $\mathcal{W}$ とは，テンプレート画像 $\mathcal{T}$ 中の画素の重要度を 0.0 から 1.0 までの値として各画素に与えた画像のことである．今回の実験においては，先ほどの例で説明すると画素が黒いほどその重みが高くなるような重

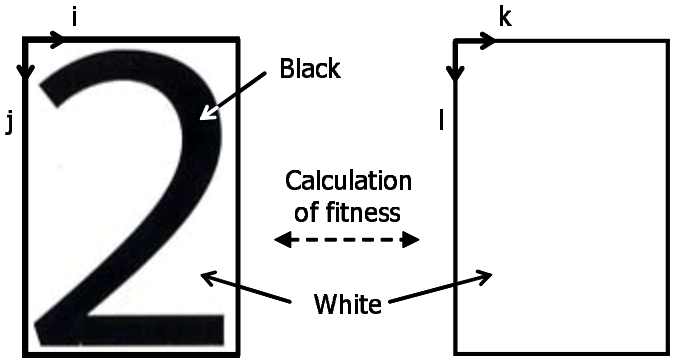


図 3.4 テンプレート画像 $\mathcal{T}$ と部分画像 $\mathcal{I'}$

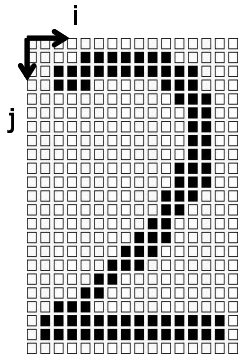


図 3.5 重み画像 $\mathcal{W}$

み画像を作成し用いることとした．図 3.5 は重み画像の一例であり，この場合，例えば黒い画素 (i_b, j_b) は重み $\mathcal{W}(i_b, j_b) = 1.0$，白い画素 (i_w, j_w) は重み $\mathcal{W}(i_w, j_w) = 0.5$ を表している．

3.2.3　優れた個体の抽出と局所探索および個体の増殖

免疫度が上位 $r_l\%$（$r_l \leq 50$）の個体のみを残してその他を淘汰し，残された高い免疫度を持つ個体群の上位半分に対して局所探索を行う．淘汰した個体数分の補充は，$\lfloor N \times r_l/100 \rfloor$ 個分については淘汰されなかった高い免疫度を持つ個体群の突然変異により生成し，残り分については新たにランダム生成することで行う．ただし，$\lfloor z \rfloor$ は z を超えない最大整数を表す．なお，ここでは，突然変異としてランダムに決定した 1 遺伝子座のビット反転を適用し

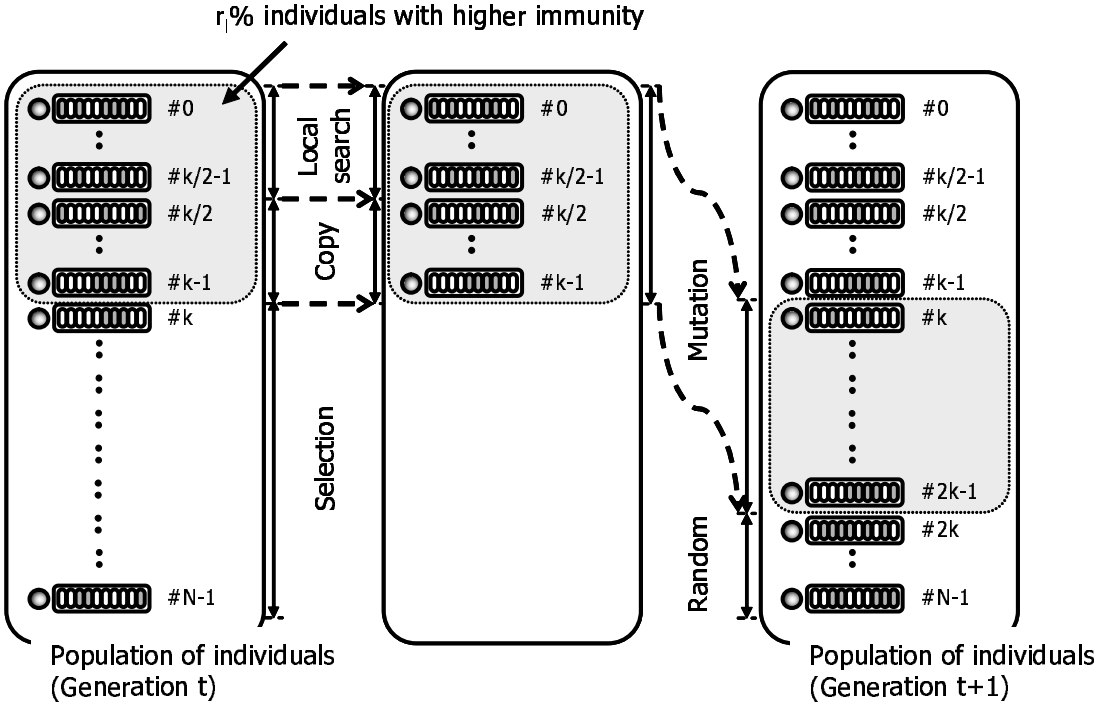
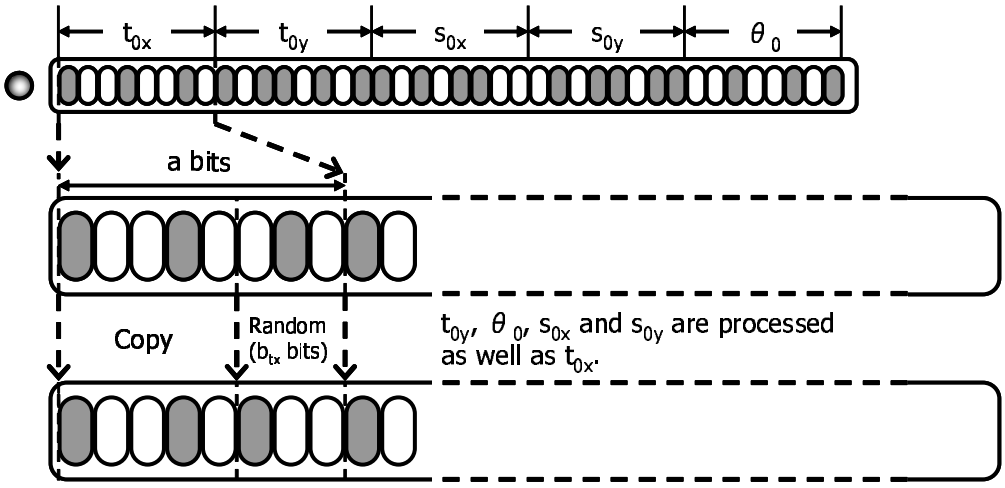


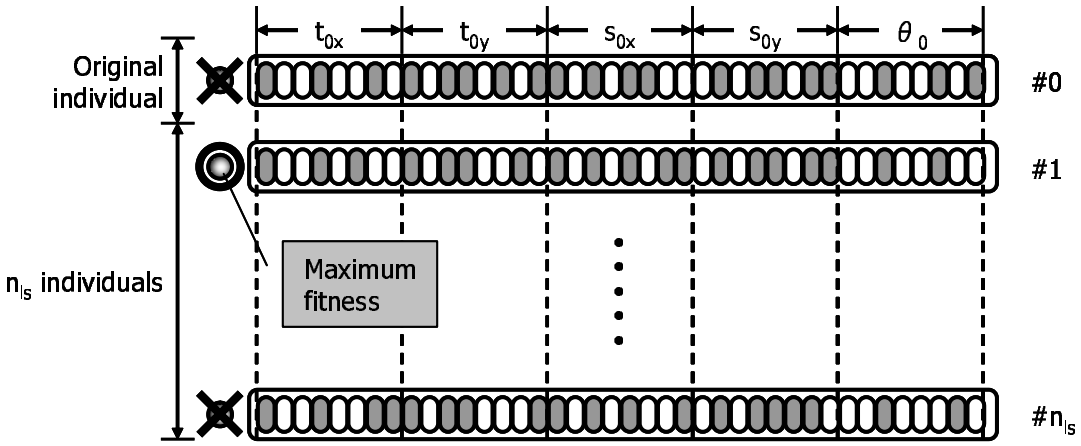
図 3.6 優れた個体の抽出と局所探索および個体の増殖

た．図 3.6 は，これらの処理を図示したものである．

次に，提案手法の 1 つの特徴である優れた個体の近傍を探索する局所探索の詳細について説明する．図 3.6 で示したように，次世代に残る上位 $r_l\%$ の個体のうち，特に優れていると考えられる上位 $r_l/2\%$ の個体については局所探索を行う．局所探索は，個体の姿勢パラメータ $(t_x, t_y, s_x, s_y, \theta)$ に対して，それぞれ下位 b_{t_x}，b_{t_y}，b_{s_x}，b_{s_y}，b_θ ビットをランダムに変化させたものを n_{ls} 個生成することで特に優れた個体の周囲を探索する．図 3.7 はこの処理を図示したものであり，図 3.7(a) は t_x に関する解 b_{t_x} ビットをランダムに変化させたものである．この t_x に関する処理を，その他の要素である t_y，s_x，s_y，θ に対しても同様に行うことで，優れた個体の近傍の新たな固体が生成される．図 3.7(b) は，もとの個体と，各姿勢パラメータの下位ビットをランダムに変化させる処理を n_{ls} 回行って生成された n_{ls} 個の個体のう



(a) Random generation of low order bits on t_x



(b) Original individual and n_{ls} individuals whosw low order bits are generated randomly on each element $(t_x, t_y, s_x, s_y, \theta)$

図 3.7 局所探索

ち，最大適応度を持つエリート個体のみが採用され，その他の個体は淘汰される．

3.2.4　免疫システムの抑制機構を模倣した類似個体の抑制

免疫システムは，生体内に侵入した抗原を認識し，過去に排除した抗原に対しては記憶細胞から直接抗体を作り出し，素早く排除する．また，新しい抗原に対しては，細胞遺伝子の再構成によってそれに対応する抗体産生細胞を作り，これを増殖して対応する抗体を作り出し，排除する．さらに，免疫システムは自己に対しても免疫性を示し，定常状態に戻す機構を備える．この機構によって無限に近い種類の抗原に対応することができる．ここで，生体内に大量に発生した抗体の産生を抑制する働きをするのがサプレッサー T 細胞であり，提案手法の特徴である類似個体の抑制は，このサプレッサー T 細胞の機構を模倣したものである．

類似した個体の抑制の方法は，探索対象画像領域 $\mathcal{R}$ のある部分領域 $\mathcal{R}'$（幅 $\mathcal{R}'_w$，高さ $\mathcal{R}'_h$）に着目し，$\mathcal{R}'$ に存在する個体数が閾値 T_{sp} を超えた場合，そのうちの適応度が上位 $r_s\%$ の個体のみを残し，下位の個体は淘汰して新たにランダム生成する．つまり，大量に発生した類似個体を抑制し，多様性を維持する．$\mathcal{R}$ を満遍なく覆うように $\mathcal{R}'$ を移動させて同様の処理を行うことで，類似した個体の抑制を実現する．すなわち，この処理は，ある近傍に大量に発生した類似個体を抑制し，これにより個体の多様性を維持することによる探索性能の向上を期待するものである．図 3.8 は，この処理の概要を図示したものである．まず，探索対象画像 $\mathcal{I}$ の画像領域 $\mathcal{R}$ の座標 $(0, 0)$ から幅 $\mathcal{R}'_w$，高さ $\mathcal{R}'_h$ の部分画像領域 $\mathcal{R}'$ を設定する．そして，この $\mathcal{R}'$ 内に存在する個体数を数え閾値 T_{sp} と比較する．もし閾値 T_{sp} を超えていた場合，そのうちの適応度が上位 $(r_s)\%$ の個体のみを残し，下位の個体は淘汰して新たにランダ

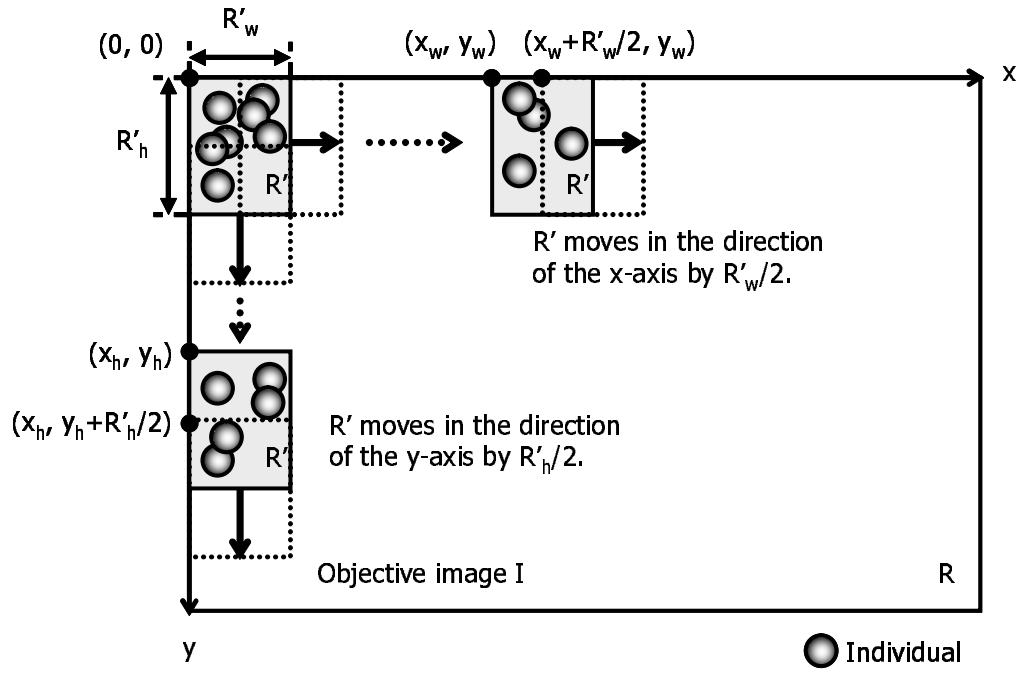


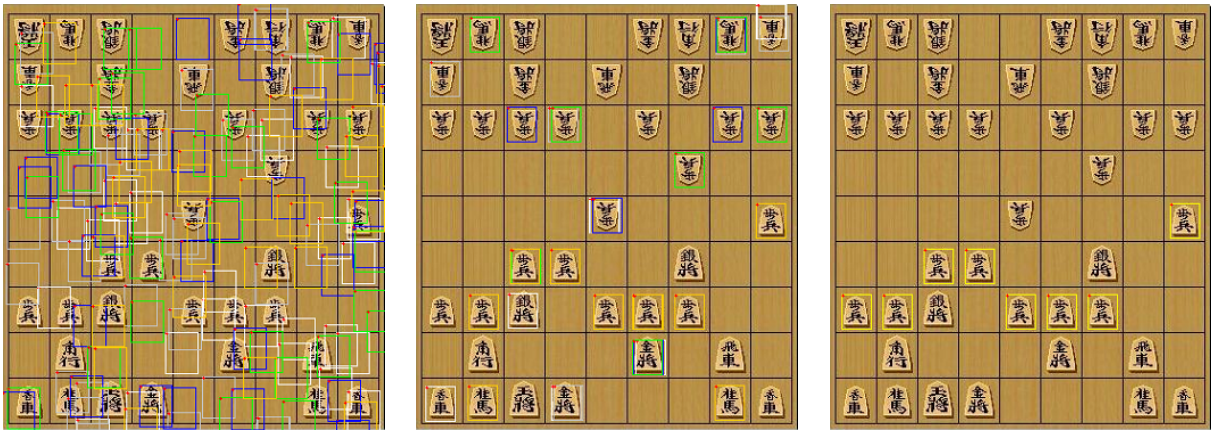
図 3.8 類似個体の抑制機構

ム生成する．閾値 T_{sp} を超えていない場合は，特に何の処理も行わない．$\mathcal{R}$ を満遍なく覆うように，x 軸方向，y 軸方向，それぞれ ${\mathcal{R}'}_w/2$，${\mathcal{R}'}_h/2$ ずつ $\mathcal{R}'$ を移動させて同様の処理を行う．$\mathcal{R}'$ をそれ以上移動させることができなくなった場合，この類似個体の抑制処理は終了となる．

3.3　評価実験の結果と考察

3.3.1　将棋盤画像からの駒「歩兵」の探索（実験 1）

探索対象画像 $\mathcal{I}$ に将棋盤（356×396 ピクセル）を用いて，駒「歩兵」（27×32 ピクセル）のテンプレート画像 $\mathcal{T}$ を探索する実験を行った．実際の将棋対戦における実写真画像での戦況判定を行わせるような画像認識処理システムを想定しているが，今回は予備的な実験としてインターネットでの将棋盤画像を対象としている．図 3.2 では一般的な染色体のコーディングを示したが，ここでは姿勢パラメータとして (t_x, t_y) を採用し，部分画像 $\mathcal{I}'$ は対象画像 $\mathcal{I}$ 内の任意な位置に依存するものとした．なお，図 3.2 における g_{t_x}，g_{t_y} のビット数は探索対象画像 $\mathcal{I}$ のサイズをもとに決定し，それぞれ 9 ビットとした．実験で使用したパラメータは，予備実験を行い決定し，個体数 $N = 500$，免疫度を求める際の個体グループの抽出個体数 $N_{sub} = 10$，個体抽出の繰返し回数 $n_r = 10,000$，優れた個体の選択処理における個体の存続率 $r_l = 50\%$とした．ここでは，染色体にコーディングした姿勢パラメータが t_x と t_y の 2 つであり問題が比較的単純なため，局所探索は用いていない．また，類似個体の抑制処理における個体数の閾値 $T_{sp} = 10$，同処理における個体の存続率 $r_s = 20\%$とした．そして，部分領域 $\mathcal{R}'$ は 5×5 ピクセルの矩形領域とし，その移動は x，y 軸方向共にそれぞれ 3 ピクセル



(a) Initial state

(b) Result of evolution until the 20th generation(Drawing concerning immunity)

(c) Result of evolution until the 3,000th generation(Drawing concerning fitness)

図 3.9 駒「歩兵」探索過程

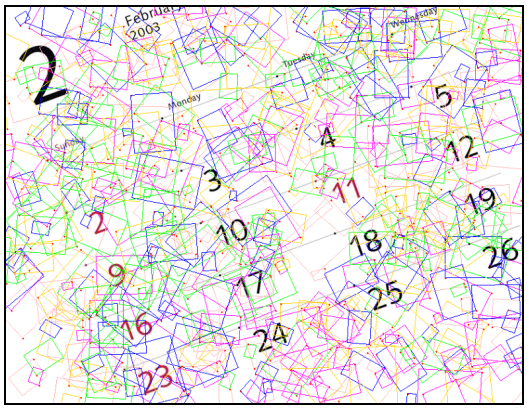
ずつずらしていくものとした．なお，重み画像の作成においては，文字部の重みを 1.0，背景部を 0.1 とした．

上記の実験を 10 回行った結果，80 個（探索対象画像 $\mathcal{I}$ 内の「2」の数 × 実験回数）すべてを探索できたときを 100%とすると，500 世代進化後において 81.3%，1,000 世代進化後において 95.0%，そして 1,300 世代進化後において 100.0%の探索成功率を得ることができた．ただし，ここでは，実験における目視による確認の結果を踏まえて，適応度 *fitness* が 0.861 以上の場合を探索が成功したと判断している．図 3.9 は，探索の過程を示している．世代が進むにつれて，テンプレート画像 $\mathcal{T}$ と類似した複数の部分画像 $\mathcal{I}'$ 付近に個体が集中していく様子がわかる．また，類似個体の抑制機構により，3,000 世代進化後も個体の多様性が保たれ，複数の部分画像領域を探索できることが確認できた．なお，他の駒に対しても実験を行った結果,「歩兵」の場合とほぼ同様の結果を得ることができた．

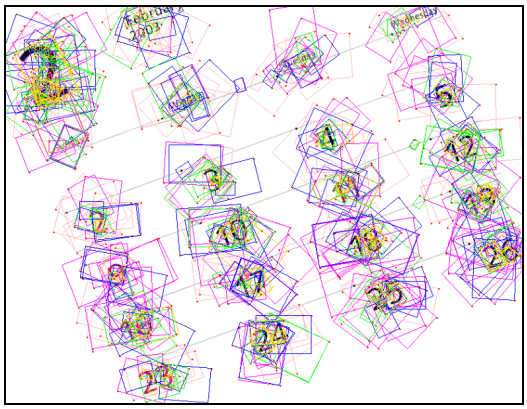
3.3.2　カレンダー画像からの数字「2」の探索（実験 2）

探索対象画像 $\mathcal{I}$ にカレンダー（821 × 630 ピクセル）を用いて，文字「2」（12 × 17 ピクセル）のテンプレート画像 $\mathcal{T}$ を探索する実験を行った．図 3.2 では一般的な染色体のコーディングを示したが，ここでは姿勢パラメータとして (t_x, t_y, s, θ)（ただし，$s = s_x = s_y$）を採用し，部分画像 $\mathcal{I}'$ は対象画像 $\mathcal{I}$ 内の任意な位置に依存して，縮尺 s は 0.8 倍から 6.2 倍，傾き θ は 360 °の範囲で自由度を持つものとした．なお，図 3.2 における g_{t_x}，g_{t_y} のビット数は探索対象画像 $\mathcal{I}$ のサイズをもとに決定しそれぞれ 10 ビットとした．実験で使用したパラメータは，予備実験を行い決定し，個体数 $N = 7,000$，免疫度を求める際の個体グループの抽出個体数 $N_{sub} = 10$，個体抽出の繰返し回数 $n_r = 10,000$，優れた個体の選択処理における個体の存続率 $r_l = 40\%$とした．局所探索に関するパラメータは，$(b_{t_x}, b_{t_y}, b_s, b_\theta) = (4, 4, 3, 3)$，$n_{ls} = 4$ とした．また，類似個体の抑制処理における個体数の閾値 $T_{sp} = 10$，同処理における個体の存続率 $r_s = 20\%$ とした．そして，部分領域 $\mathcal{R}'$ は 20 × 20 ピクセルの矩形領域とし，その移動は x，y 軸方向共にそれぞれ 10 ピクセルずつずらしていくものとした．なお，重み画像の作成においては，文字部の重みを 1.0，背景部を 0.5 とした．

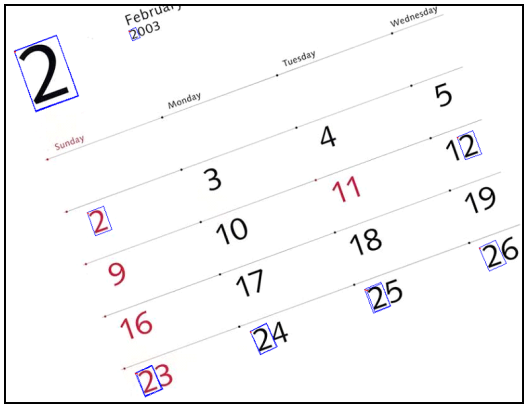
上記の実験を 10 回行った結果，80 個（探索対象画像 $\mathcal{I}$ 内の「2」の数 × 実験回数）すべてを探索できたときを 100%とすると，3,000 世代進化後において 97.5%の探索成功率を得ることができた．ただし，ここでは，実験における目視による確認の結果を踏まえて，適応度 *fitness* が 0.829 以上の場合を探索が成功したと判断している．図 3.10 は，探索の過程を示している．駒「歩兵」の探索の場合と同様，世代が進むにつれて，テンプレート画像 $\mathcal{T}$ と類似した複数の部分画像 $\mathcal{I}'$ 付近に個体が集中していく様子がわかる．また，類似個体の抑制機構により，10,000 世代進化後も個体の多様性が保たれ，複数の部分画像領域を得れることが確認できた．



(a) Initial state



(b) Result of evolution until the 20th generation(Drawing concerning immunity)

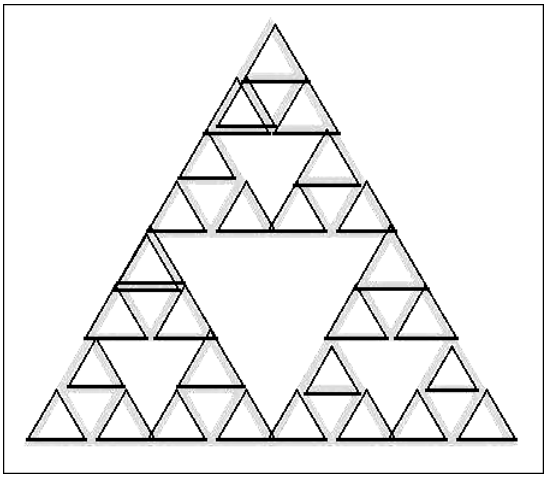


(c) Result of evolution until the 3,000th generation(Drawing concerning fitness)

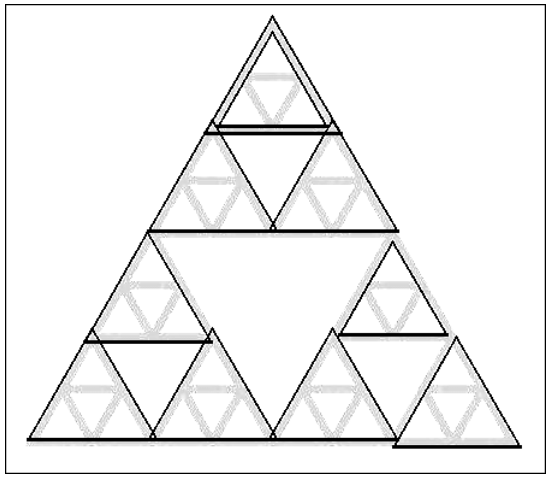
図 3.10 数字「2」の探索過程

3.3.3　フラクタル図形画像からの図形「△」の探索（実験 3）

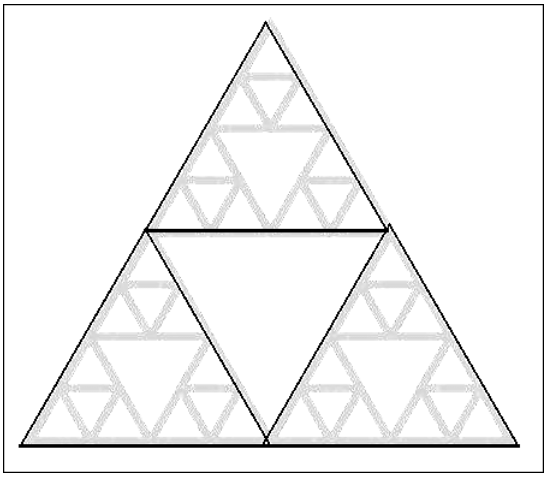
探索対象画像 $\mathcal{I}$ にシルピンスキーのガスケット（Sierpinski's gasket）[本田 02]（450×390 ピクセル）を用いて，図形「△」（53×46 ピクセル）のテンプレート画像 $\mathcal{T}$ を探索する実験を行った．図 3.2 では一般的な染色体のコーディングを示したが，ここでは，姿勢パラメータとして (t_x, t_y, s)（ただし，$s = s_x = s_y$）を採用し，部分画像 $\mathcal{I}'$ は対象画像 $\mathcal{I}$ 内の任意な位置に依存して，縮尺 s は 0.8 倍から 8.2 倍の範囲で自由度を持つものとした．なお，図 3.2 における g_{t_x}，g_{t_y} のビット数は探索対象画像 $\mathcal{I}$ のサイズをもとに決定しそれぞれ 9 ビットとした．実験で使用したパラメータ値は，これまでと同様に予備実験を行い決定し，個体数 $N = 500$，免疫度を求める際の抽出個体数 $N_{sub} = 10$，個体抽出の繰返し回数 $n_r = 10,000$，優れた個体の選択処理における個体の存続率 $r_l = 40\%$とした．局所探索に関するパラメータは，$(b_{t_x}, b_{t_y}, b_s) = (4, 4, 3)$，$n_{ls} = 4$ とした．また，類似個体の抑制処理における個体数の閾



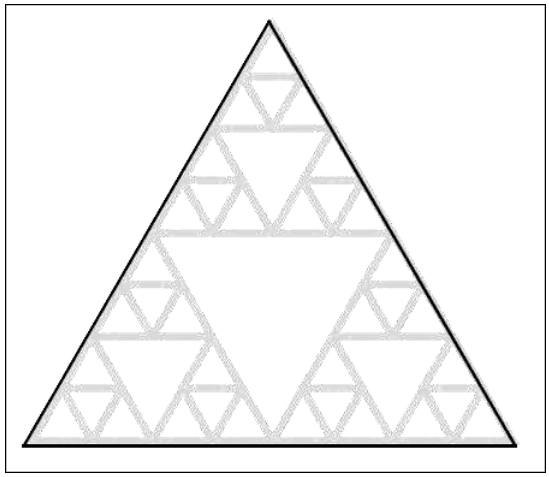
(a) Original size(Drawing concerning fitness)



(b) Twice size(Drawing concerning fitness)



(c) Four times size(Drawing concerning fitness)



(d) Eight times size(Drawing concerning fitness)

図 3.11 図形「△」のサイズ別の探索結果（3,000 世代進化後）

値 $T_{sp} = 5$，同処理における個体の存続率 $r_s = 20\%$とした．そして，部分領域 $\mathcal{R}'$ は 10×10 ピクセルの矩形領域とし，その移動は x，y 方向共にそれぞれ 5 ピクセルずつずらしていくものとした．なお，重み画像の作成においては，図形部の重みを 1.0，背景部を 0.0 とした．

上記の実験を 10 回行った結果，400 個（探索対象画像 $\mathcal{I}$ 内の図形「△」の数 × 実験回数）すべてを探索できたときを 100%とすると，3,000 世代進化後において約 99.8%の探索成功率を得ることができた．ただし，ここでは，実験における目視による確認の結果を踏まえて，適応度 *fitness* が 0.739 以上の場合を探索が成功したと判断している．図 3.11 は，その探索結果を図形「△」のサイズごとに示したものであり，図 3.11(a) はテンプレート画像 $\mathcal{I}$ に対して 1 倍の図形「△」，図 3.11(b) は 2 倍，図 3.11(c) は 4 倍，そして図 3.11(d) は 8 倍の図形「△」の探索結果である．ここでは探索結果を見やすくするために，図形「△」のサイズご

表 3.1 3 つの実験の探索成功率

Object image $\mathcal{I}$	Generation	Search Success rate	
Chessboard	500	81.25 %	(65 / 80)
	1,000	95.00 %	(76 / 80)
	1,300	100.00 %	(80 / 80)
Calendar	3,000	97.50 %	(78 / 80)
Gasket	3,000	99.75 %	(399 / 400)

とにその結果を示したが，これらサイズの異なる「△」は同時に探索される．図 3.11 を見ると，探索された図形「△」が微妙にずれている箇所が見受けられるが，これはテンプレート画像 $\mathcal{T}$ における線の太さが 1 ドットであるのに対して探索対象画像 $\mathcal{I}$ では線の太さが 8 ドットであったために起きたものと考えられる．

なお，これまで述べてきた 3 つの実験の探索成功率を表 3.1 にまとめる．ここで，探索成功率で括弧書きの数字は，分母が探索すべき部分画像 $\mathcal{I}'$ の総数，分子が探索できた解の総数である．

3.3.4　類似個体の抑制機構と局所探索の効果

先に述べた 3 つの実験のうち，最も難しい問題であると考えられる実験 2 を対象として，抑制機構および局所探索の効果を確認する実験を行った結果を図 3.12 に示す．同図は，横軸が世代数，縦軸が探索できた「2」の個数であり，プロットした値は 10 回試行の平均値で

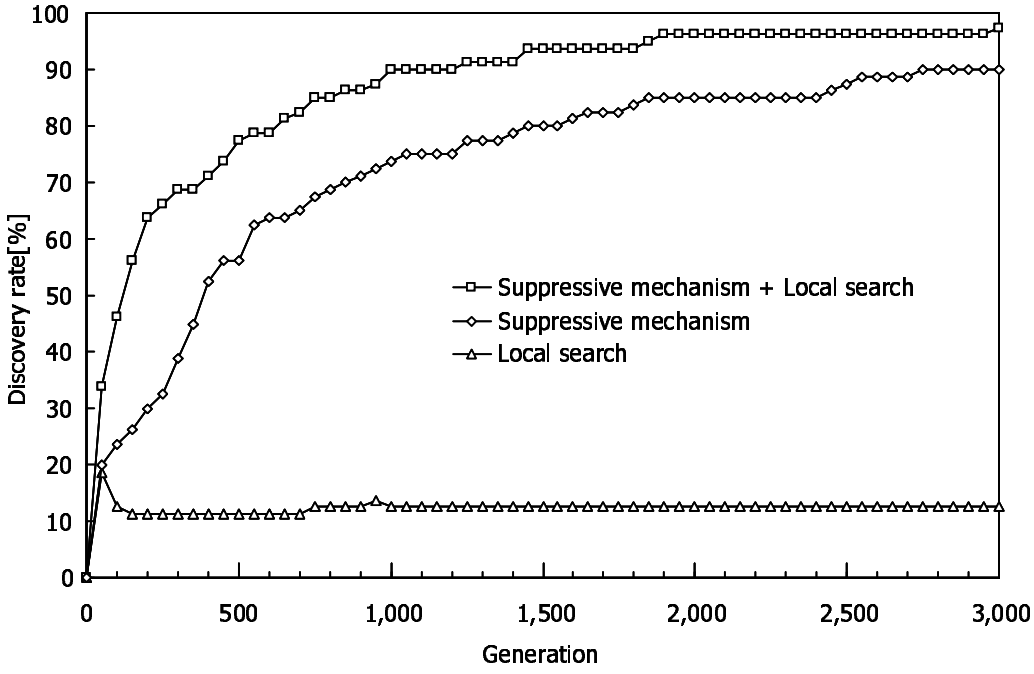


図 3.12 抑制機構と局所探索の効果

ある．

局所探索のみを加味した場合，早い世代で収束が見られ，1,000 世代進化後には 1 つの局所的最適解に収束してしまっている．これは，局所探索の影響に加え，個体の多様性を維持する抑制機構がないことによる多様性維持能力の低下により，唯一の大局的最適解に収束してしまっている効果であると考えられる．一方，抑制機構のみを加味した場合，複数の局所的最適解を得る傾向にあることが確認できる．また，抑制機構と局所探索の両方を加味した場合は，抑制機構のみの場合に比べて，より少ない世代数で多くの局所的最適解を得られることが認められる．

以上のことから，提案手法は，抑制機構により長い世代進化を続けても唯一の大局的最適解に収束することなく，また局所探索を行うことでより早く複数の局所的最適解を得ることができる特徴を有する手法であるといえる．

3.3.5　抑制機構を有する GAIS による書籍特定

提案する抑制機構を有する GAIS を用いて書架画像中に存在する複数の書籍の巻，号画像を探索し，探索された解と書籍境界の情報をもとに書籍を特定する実験を行った．

まず，探索対象とする書架画像に対してエッジ強調 [藤岡 02] とハフ変換 [田村 02] を行い書籍境界を検出し，配架された各書籍の領域を決定する．次に，テンプレート画像に書籍の背表紙に印刷された巻，号画像を用い，書架画像中のすべての巻，号画像を探索する．このとき，複数種類のテンプレート画像はそれぞれ独立して探索され，1 種類のテンプレート画像の探索が終了するたびに母集団は初期化されるものとする．その後，発見された巻，号画像の位置と書籍境界の検出により得られた書籍領域から各書籍を特定する．

探索対象画像 $\mathcal{I}$ に書架画像（611×475 ピクセル）を用いて，巻，号画像のテンプレート画像 $\mathcal{T}$ を探索する実験を行った．各テンプレート画像のサイズを表 3.2 に示す．図 3.2 と同様に，ここでは姿勢パラメータとして $(t_x, t_y, s_x, s_y, \theta)$ を採用し，部分画像 $\mathcal{I}'$ は対象画像 $\mathcal{I}$ 内の任意な位置に依存して，縮尺 s_x，s_y は 0.8 倍から 1.2 倍の範囲で自由度を持つものとし，

表 3.2 各テンプレート画像 $\mathcal{T}$ のサイズと閾値 T_f

Template image $\mathcal{T}$	Image size [pixels]	Threshold T_f
0	15×24	0.604
1	11×25	0.574
2	16×25	0.546
6	15×25	0.604
7	15×24	0.537
8	15×24	0.629
9	15×23	0.631
67	25×18	0.621
68	27×21	0.636
69	27×20	0.633

図 3.13 68 巻 10 号の書籍特定

傾き θ は書籍境界の検出により得られた書籍の傾き情報をもとに制限するものとした．なお，図 3.2 における g_{t_x}，g_{t_y} のビット数は探索対象画像 $\mathcal{I}$ のサイズをもとに決定し，g_{t_x} を 9 ビット，g_{t_y} を 10 ビットとした．実験で使用したパラメータ値は，これまでと同様に予備実験を行い決定し，個体数 $N = 700$，免疫度を求める際の抽出個体数 $N_{sub} = 10$，個体抽出の繰返し回数 $n_r = 10,000$，優れた個体の選択処理における個体の存続率 $r_l = 40\%$とした．局所探索に関するパラメータは，$(b_{t_x}, b_{t_y}, b_{s_x}, bs_y) = (4, 4, 4, 4)$，$n_{ls} = 4$ とした．また，類似個体の抑制処理における個体数の閾値 $T_{sp} = 7$，同処理における個体の存続率 $r_s = 20\%$とした．そして，部分領域 $\mathcal{R}'$ は 10×10 ピクセルの矩形領域とし，その移動は x，y 方向共にそれぞれ 5 ピクセルずつずらしていくものとした．なお，重み画像の作成においては，図形部の重みを 1.0，背景部を 0.25 とした．

上記の実験を 10 回行った結果，330 個（探索対象画像 $\mathcal{I}$ 内の巻，号画像の数 $\times$ 実験回数）すべてを探索できたときを 100%とすると，3,000 世代進化後において約 98.5%の探索成功率を得ることができた．ただし，ここでは，実験における目視による確認の結果を踏まえて，適応度 $fitness$ が閾値 T_f 以上の場合を探索が成功したと判断している．また，本実験では，テンプレート画像 $\mathcal{T}$ として，巻画像には二桁の数字が描かれた画像を用い，号画像には一の位と十の位を分割した一桁ずつの数字が描かれた画像を用いた．各テンプレート画像の閾値 T_f を表 3.2 に示す．

図 3.13 は 68 巻，10 号の書籍を特定した結果である．まず，提案する GAIS を用いて，対象とする書架画像内に存在するすべてのテンプレート画像と類似した部分画像領域を探索する．テンプレート画像には，68 巻の “68”，10 号の十の位の “1” と一の位の “0” を用い，それぞれのテンプレート画像を独立して探索する．次に，エッジ強調とハフ変換により得られた書籍境界をもとに，対象画像 $\mathcal{I}$ 内の各書籍境界領域を決定し，得られたテンプレート画像

$\mathcal{T}$“68”，“1”，“0” と類似した部分画像領域の位置情報をもとに，“68”，“1”，“0” の探索結果がすべて 1 つずつ含まれる書籍領域を決定することで，探索すべき 68 巻，10 号の書籍を特定することができた．

3.4　まとめ

本章では，免疫システムにおけるサプレッサー T 細胞がリンパ球前駆細胞から分化した B 細胞の産生を抑える機構を模倣して類似した個体が過剰に増えないようにする抑制機構と優れた個体の近傍を探索する局所探索を GAIS に導入し，対象画像内から 2 次元的な姿勢自由度を持つ複数の部分画像領域を探索する手法を提案した．提案手法は，類似個体の抑制機構により長い世代進化を続けても唯一の大局的最適解に収束することなく，また局所探索を行うことでより早く複数の局所的最適解解を探索できるという特徴を有する．実験の結果，提案手法は，従来手法の問題点である進化の世代の長さによっては唯一の大局的最適解に収束してしまう現象を効率的に回避し，複数の局所的最適解を探索できることを確認できた．

また，提案する GAIS を書籍特定に応用し，対象とする書架画像内から複数種類の巻，号画像の位置情報を探索し，エッジ強調とハフ変換により得られた書架画像中の書籍領域と照らし合わせることで特定したい書籍を得ることができる．

今後は，この提案手法の更なる詳細な検討を行うため，X 線画像や CT 画像などの医療画像のようなさまざまな画像に対しての適用実験を行っていくとともに，画像探索以外の問題への適用の可能性を探る必要があると考えられる．

第 4 章

免疫アルゴリズムを用いた複数画像探索と書籍特定への応用

4.1 はじめに

遺伝的アルゴリズム（Genetic Algorithm: GA）が唯一の大局的最適解を探索することを目的としているのに対して，生体の獲得免疫を模倣した免疫アルゴリズム（Immune Algorithm: IA） は，抗体の多様性を維持しながら解探索を行うことにより初期収束を回避し，大局的最適解を含む複数の局所的最適解を探索することができる [森 97, 三井 04, 飯村 05, 廣谷 06, 當間 00, 本間 05]．実世界の最適化問題では大局的最適解だけが要求されるわけではなく，設計者が複数の代替案の中から解を選択できるように，実現可能な優れた複数の局所的最適解を探索できることが望まれる [森 97, 廣谷 06, 本間 05]．例えば，建築構造の形態を求める問題では，大局的最適解だけではなく，デザインを考慮した実現可能な評価の優れた解も重要になる場合がある [本間 05]．

IA を複数画像領域探索問題に適用し，対象画像内からテンプレート画像と一致性の高い複数の部分画像領域を効率的に解探索できることが報告されている [飯村 05]．抗体の遺伝子として，対象画像にマッピングすべき姿勢パラメータをコーディングし，抗体群を獲得免疫に倣い進化させることで，対象とする画像の中から，縮尺と傾きの異なる複数の解を同時に探索することができる．

しかし，飯村らが提案する手法（以下，従来手法） [飯村 05] では，抗体の遺伝子型の空間と実際に探索を行う空間との位相構造が大きく異なるため，量子化の精度によっては，十分な精度の解が得られない場合や探索に無駄が生じる場合がある．また，探索空間の連続性を無視した交叉により，抗体の収束した有望な領域を重点的に探索することができない．さらに，1 種類のテンプレート画像探索を目的としているため，対象画像内から複数種類のテンプレート画像を探索する場合，テンプレート画像毎に独立して探索を行う必要があり，探索効率が低下する．

本章では，テンプレート画像を識別するための ID を遺伝子として組込んだ整数型 IA を提案する．提案する方式では，抗体の遺伝子を探索空間と同じ整数でコーディングすること

で量子化の精度を考慮する必要がない．また，遺伝子型に即した交叉方法の適用と，テンプレート画像の複数同時探索により探索効率の向上を図る．

今回の実験では，提案する方式を書籍特定に応用しその有効性を検証する．図書館や書店などでは，利用者によって書籍の出し入れが頻繁に繰返されるため書籍管理に多大な労力が必要となる．書籍管理の労力軽減を目的として，書籍の背表紙に書かれた文字を抽出しタイトルや巻号などの書籍情報を自動認識する技術が提案されている [澤木 00]．

従来の書架画像中の文字認識法では，まず書架画像中の各書籍の境界を検出し書籍画像の領域を決定する．次に，書籍領域の傾き補正と文字サイズの正規化を行い，書籍画像を抽出する．その後，抽出した書籍画像と，予め用意した文字パターンとをずらしながらマッチングすることで文字を認識する．

しかしながら，実際の書架中には，整理された書籍だけでなく傾いた書籍や大きさの異なる書籍が存在するため，書架画像中の書籍領域や文字領域を決定することが困難である．そのため，書籍情報の自動認識を支援することを目的として，書籍境界や書籍領域を決定する手法が検討されている [長尾 91, 大竹 92, 石川 97, 平 04]．

提案する方式では，抗体の遺伝子にコーディングされた縮尺と傾きに従ってマッチングを行うため，大きさや傾きの異なる書籍を対象に，画像中から書籍情報を探索することができる．

以下，4.2 節では，従来の方式であるバイナリ型 IA ついて概説する．4.4 節では，提案する方式について説明する．4.5 節では，実験内容について述べ，実験結果をもとに提案する方式の有効性について考察する．

4.2　IA による複数画像探索

まず，遺伝子をランダムに決定した抗体を N_v 個生成し初期母集団を形成する．次に，各抗体の指標である親和度，類似度，濃度をそれぞれ計算する．濃度の高い抗体は記憶細胞とサプレッサー T 細胞へ分化し，抗原を排除する有効な抗体の記憶と過剰に収束した抗体の排除を行う．その後，期待値の計算と母集団内で親和度の低い抗体の淘汰を行い，交叉，突然変異により淘汰された抗体の補充を行う．これらの処理が終了条件を満すまで繰返さることにより，大局的最適解を含む複数の局所的最適解を探索することができる．

以下，図 2.8 に示された IA の処理手順に従って，IA を複数画像探索問題へ適用した場合の各処理の詳細について述べる．

Step 1〔初期化〕

まず，抗原の認識を行う．今回の実験では，対象画像 $\mathcal{O}$ 内にある巻号情報を包含する矩形領域が抗原とする．

次に，ランダムに抗体の遺伝子を決定し，初期世代（$t = 0$）の母集団 $P(0)$ を生成する．このとき，第 t 世代の抗体群が形成する母集団を $P(t)$ とする．抗体には，探索すべき部分画像の姿勢パラメータをバイナリ型でコーディングする．つまり，部分画像

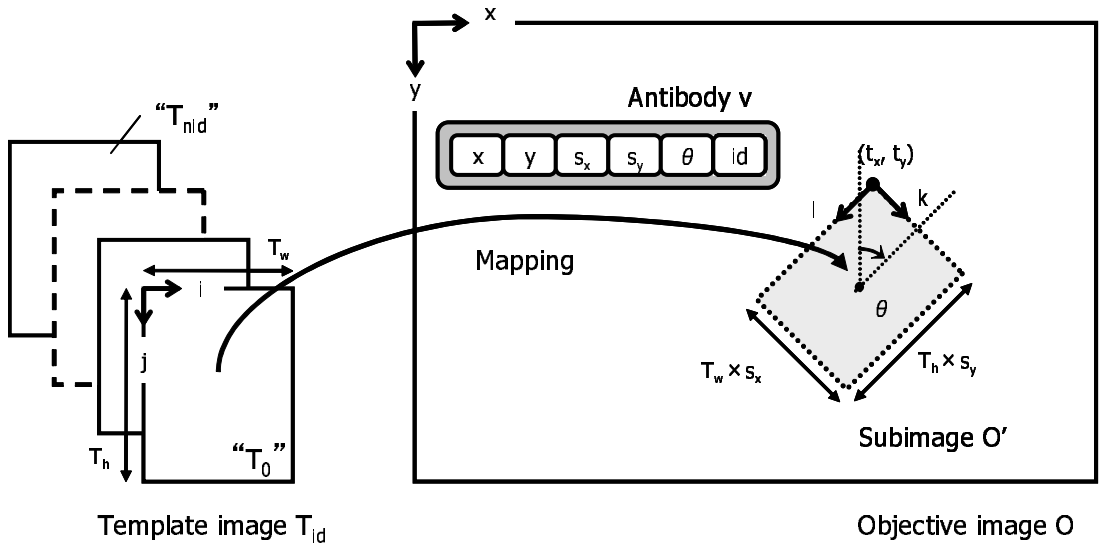


図 4.1 抗体の親和度評価

を囲む矩形の位置（t_x, t_y），縮尺（s_x, s_y），傾き θ をそれぞれ g ビットで表し，これら 5 種類のパラメータを 1 次元的に並べた配列を抗体とする．

Step 2〔評価〕

各抗体 v について抗原に対する有効性を示す親和度 Φ_v を計算する．親和度 Φ_v は，図 4.1 に示すように，各抗体 v の姿勢パラメータ $(t_x, t_y, s_x, s_y, \theta)$ によって決まる対象画像 $\mathcal{O}$ 内の部分画像 $\mathcal{O}'$ と，テンプレート画像とを対応する画素毎に比較することで求められる．このとき，各画素値の差 $\delta_C(i,j)$ の絶対値の合計が小さいほど解に近い抗体と考え，より高い親和度を与える．

$\mathcal{T}_w \times \mathcal{T}_h$ 画素からなるテンプレート画像 $\mathcal{T}$ と，抗体 v の姿勢パラメータ $(t_x, t_y, s_x, s_y, \theta)$ によってマッピングされた部分画像 $\mathcal{O}'$ との親和度 Φ_v は式（4.1）で定義される．

$$\Phi_v = \frac{\displaystyle\sum_{i=1}^{\mathcal{T}_w}\sum_{j=1}^{\mathcal{T}_h}\sum_{C=\{R,G,B\}} \mathcal{W}(i,j)\cdot(C_{\max} - |\delta_C(i,j)|)}{\displaystyle\sum_{i=1}^{\mathcal{T}_w}\sum_{j=1}^{\mathcal{T}_h} \mathcal{W}(i,j)\cdot 3\cdot C_{\max}}. \tag{4.1}$$

ただし，画素値の最大値を $C_{\max}$，重み画像 $\mathcal{W}$ 内の座標 (i,j) における重みを $\mathcal{W}(i,j)$ とする．重み画像 $\mathcal{W}$ 内の重み $\mathcal{W}(i,j)$ はテンプレート画像内の文字領域の画素数を n_{ch}，背景領域の画素数を n_{bg} とするとき，式（4.2）で表される．

$$\mathcal{W}(i,j) = \begin{cases} 1 & (\text{image area of character}) \\ \dfrac{n_{ch}}{n_{bg}} & (\text{otherwise}). \end{cases} \tag{4.2}$$

Step 3〔終了判定〕

予め指定された進化の終了条件を満たしていれば Step 8〔進化終了〕へ，そうでなければ Step 4〔濃度計算〕へ進む．

Step 4〔濃度計算〕

まず，各抗体 v について他の抗体 w との類似度 Ψ_{vw} を計算する．抗体 v，w 間の類似度は，式（4.3）で計算する．

$$\Psi_{vw} = 1 - D_{vw}. \tag{4.3}$$

D_{vw} は抗体 v，w 間の距離を表す．抗体 v，w の姿勢パラメータをそれぞれ $(t_x^v, t_y^v, s_x^v, s_y^v, \theta^v)$，$(t_x^w, t_y^w, s_x^w, s_y^w, \theta^w)$ とし，2 抗体間のユークリッド距離を δ_t^{vw}，縮尺の差を $\delta_{s_x}^{vw}$，$\delta_{s_y}^{vw}$，傾きの差を δ_θ^{vw}，また，δ_t^{vw}，δ_s^{vw}，δ_θ^{vw} が取りうる最大値をそれぞれ $\delta_t^{\max}$，$\delta_{s_x}^{\max}$，$\delta_{s_y}^{\max}$，$\delta_\theta^{\max}$ とすると，抗体 v，w 間の距離 D_{vw} は式（4.4）で定義される．

$$D_{vw} = \omega_t \frac{\delta_t^{vw}}{\delta_t^{\max}} + \omega_s \frac{|\delta_{s_x}^{vw}|}{\delta_{s_x}^{\max}} + \omega_s \frac{|\delta_{s_y}^{vw}|}{\delta_{s_y}^{\max}} + \omega_\theta \frac{|\delta_\theta^{vw}|}{\delta_\theta^{\max}}. \tag{4.4}$$

ただし，ω_t，ω_s，ω_θ は，各パラメータに対する重みであり，$\omega_t + 2\omega_s + \omega_\theta = 1$ とする．また，今回の実験では id 遺伝子の同異は濃度に反映させないものとした．

次に，類似度 Ψ_{vw} が閾値 $T_{\pi 1}$ 以上の抗体 v，w を類似抗体とみなし，式（2.2）で定義される各抗体 v の濃度 Θ_v を求める．

Step 5〔分化〕

濃度 Θ_v が閾値 T_{II} 以上の抗体 v_c を記憶細胞候補 v^* とする．まず，記憶細胞候補 v^* との類似度 Ψ_{v^*m} が閾値 T_m 以上の記憶細胞 m が存在する場合，記憶細胞の多様性向上を目的として，それぞれの親和度を比較し $\Phi_{v^*} > \Phi_m$ のとき記憶細胞 m を記憶細胞候補 v^* で更新する．次に，記憶細胞候補 v^* と類似した記憶細胞が存在しない場合，すでに分化した記憶細胞数が上限数 N_m に達していなければ v^* をそのまま記憶細胞に分化し，N_m に達していれば記憶細胞候補 v^* と最も親和度の低い記憶細胞 $m_{\min}$ との親和度を比較し，$\Phi_{v^*} > \Phi_{m_{\min}}$ のとき記憶細胞 $m_{\min}$ を記憶細胞候補 v^* で更新する．

記憶細胞候補 v^* をすべてサプレッサー T 細胞 s へ分化させ，s との類似度 Ψ_{vs} が閾値 T_s 以上の抗体を母集団 $P(t)$ から消滅させる．消滅させた抗体と同数の抗体をランダムに生成し，総数が N_v となるよう補充する．

Step 6〔選択〕

母集団 $P(t)$ 内で親和度 Φ_v が低い $N_v/2$ 個の抗体を淘汰し，残った抗体について式（2.4）で定義される期待値 E_v を計算する．

Step 7〔生殖〕

各抗体の期待値 E_v に従ったルーレット選択により $N_v/4$ 組の両親となる 2 つの抗体を決定する．決定された両親から交叉により $N_v/2$ 個の抗体を生成し，突然変異を適用することで，次世代の母集団 $P(t+1)$ を生成する．

従来手法であるバイナリ型 IA では，交叉方法として二点交叉，突然変異操作としてビット反転を用いる．

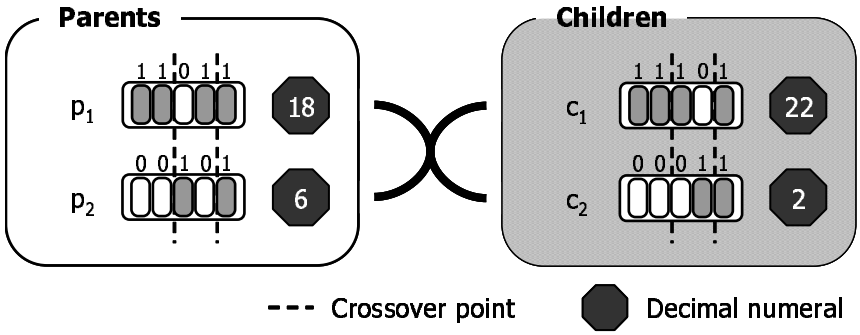
Step 8〔進化終了〕

予め指定された条件を満足したため，進化を終了する．

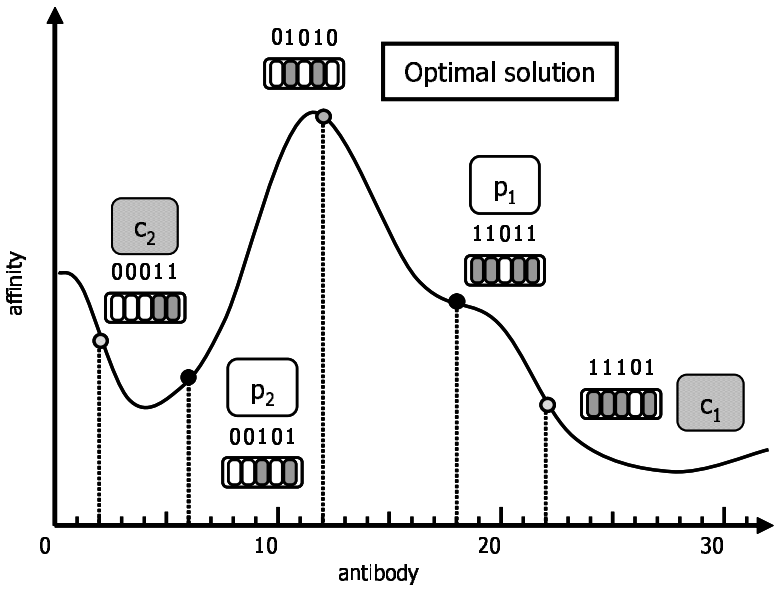
4.3　IA に基づく画像探索の利点と問題点

IA を画像探索に適用することで，対象画像内からテンプレート画像と一致性の高い複数の部分画像領域を探索することができる．また，抗体の縮尺 (s_x, s_y) と傾き θ に従ってテンプレートマッチングを行うため，対象画像内から縮尺と傾きの異なる目的の画像を探索することができる．

しかし，従来手法では，実際に探索を行う空間が整数型であるにも関わらず，抗体の遺伝子がバイナリ型でコーディングされているため，量子化の精度によっては探索空間が制限さ



(a) Two-point crossover



(b) Distribution of offspring by two-point crossover

図 4.2 グレイコーディングによる二点交叉と子の分布例

れ十分な精度の解が得られない場合や，探索空間を余計に広げてしまい探索に無駄が生じる場合がある．

また，探索空間内で近い抗体同士の遺伝子構造が必ずしも類似しているとは限らず，図 4.2 に示すように，探索空間内において近い抗体同士を両親とした交叉を行っても両親の近傍に子が生成されない場合がある．そのため，探索の中盤から終盤において，抗体が収束した有望な領域を重点的に探索することができない．

さらに，実際の書架中には多数の書籍が並べられており複数種類の書籍情報が存在するが，従来手法では 1 種類のテンプレート画像探索を目的としているため，対象画像内から複数種類のテンプレート画像を探索する場合は，テンプレート画像毎に独立して探索を行う必要がある．

そのため，問題に適した遺伝子のコーディング方法と交叉方法，また複数種類のテンプレート画像を同時に探索するための工夫が必要であると考えられる．

4.4　id 遺伝子を有する整数型 IA

4.4.1　方針と特徴

以下に提案手法の方針と特徴を示す．

遺伝子の整数型表現

本章では，抗体の各遺伝子を整数型でコーディングする．実際の探索空間と同じ型で遺伝子をコーディングすることで，バイナリコーディングの量子化精度による探索制限を回避できる．また，バイナリコーディング上で定義された探索空間の連続性を無視した交叉方法ではなく，変数間の依存関係を考慮した単峰性正規分布交叉（Unimodal Normal Distribution Crossover: UNDX） [小野 99] を用いることで，効率よく探索を進めることができる．

id 遺伝子の組込み

提案手法では，抗体の遺伝子として，姿勢パラメータ $(t_x, t_y, s_x, s_y, \theta)$ に加えて，新たにテンプレート画像を識別するための id 遺伝子を組込む．多様な id を持つ抗体を生成し探索を行うことで，一度の試行で複数種類のテンプレート画像を同時に探索することができる．また，Step 7〔生殖〕での交叉，突然変異を適用する際，id 遺伝子を積極的に変化させることで，進化が進むにつれて各抗体の近傍に存在する解に適した id が選択されると考えられるため探索効率の向上が期待できる．

4.4.2　処理手順

以下，提案手法における，4.2 節で述べた従来手法とは異なる処理について述べる．

Step 1〔初期化〕

抗原を認識し，解候補である抗体をランダムに生成する．抗体の遺伝子 $(t_x,\, t_y,\, s_x,\, s_y,\, \theta,\, id)$ を整数型でコーディングをする．

Step 2〔評価〕

各抗体 v について親和度 Φ_v を計算する．親和度 Φ_v は，各抗体 v の姿勢パラメータ $(t_x,\, t_y,\, s_x,\, s_y,\, \theta)$ によって決まる対象画像 $\mathcal{O}$ 内の部分画像 $\mathcal{O}'$ と，id 遺伝子によって決まるテンプレート画像 $\mathcal{T}_{id}$ とを対応する画素毎に比較することで求められる．

Step 5〔分化〕

濃度 Θ_v が閾値 T_H 以上の抗体 v_c に対してに最急勾配山登り法（Steepest-ascent hill climbing: HC）を適用し，記憶細胞候補 v^* とする．

従来手法では，局所的な解精度の向上を目的として，Step 7〔生殖〕の交叉，突然変異後に局所探索が適用されているが，本手法では記憶細胞候補へ分化する抗体に対して HC を適用する．濃度の高まった抗体に対して HC を適用することで，より有効な領域を探索することができる．

本実験の HC では，v_c を親として近傍抗体を n_h 個生成し，v_c を含めた $(n_h + 1)$ 個の抗体集合から最も親和度の高い抗体 v_c^{best} を選び出す．同様の処理を v_c^{best} に対して適用し，これらの処理を n_t 回繰返す．最終的に得られた v_c^{best} を v^* とする．

記憶細胞候補 v^* との類似度 Ψ_{v^*m} が閾値 T_m 以上の記憶細胞 m が存在する場合，記憶細胞の多様性向上を目的として，それぞれの親和度を比較し $\Phi_{v^*} > \Phi_m$ のとき記憶細胞 m を記憶細胞候補 v^* で更新する．記憶細胞候補 v^* と類似した記憶細胞が存在しない場合，すでに分化した記憶細胞数が上限数 N_m に達していなければ v^* をそのまま記憶細胞に分化し，N_m に達していれば記憶細胞候補 v^* と最も親和度の低い記憶細胞 $m_{\min}$ との親和度を比較し，$\Phi_{v^*} > \Phi_{m_{\min}}$ のとき記憶細胞 $m_{\min}$ を記憶細胞候補 v^* で更新する．

記憶細胞候補 v^* をすべてサプレッサー T 細胞 s へ分化させ，s との類似度 Ψ_{vs} が閾値 T_s 以上の抗体を母集団 $P(t)$ から消滅させる．消滅させた抗体と同数の抗体をランダムに生成し，総数が N_v となるよう補充する．

Step 7〔生殖〕

期待値 E_v に応じて親抗体を選択し，交叉，突然変異により次世代 $P(t)$ の母集団を生成する．本手法では交叉方法として UNDX，突然変異操作として境界突然変異を用いる．

4.4.3　抗体のコーディング

図 4.3(a) にバイナリ型の遺伝子表現，図 4.3(b) に整数型の遺伝子表現の方法を示す．バイナリ型で遺伝子を表現する場合，姿勢パラメータである位置 $(t_x,\, t_y)$，縮尺 $(s_x,\, s_y)$，傾き θ を各 g ビットで表現し，これら 5 種類の姿勢パラメータを 1 次元的に並べた配列を抗体とす

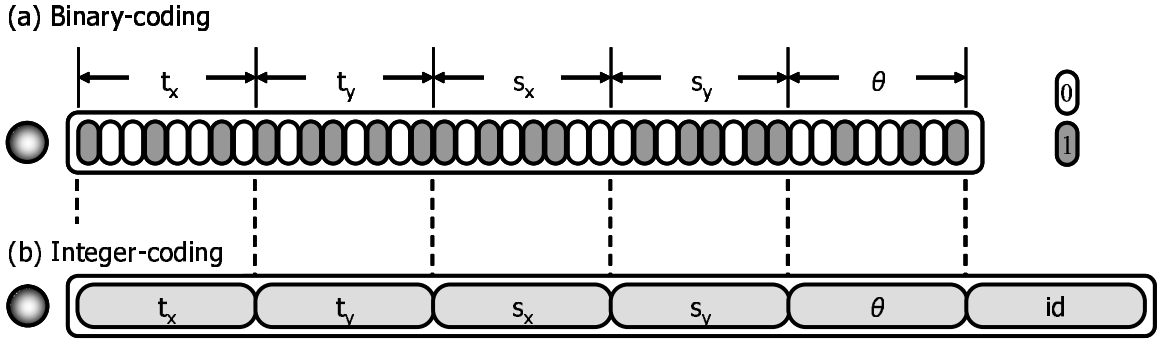


図 4.3 抗体の遺伝子表現

る．提案する手法では，姿勢パラメータ $(t_x, t_y, s_x, s_y, \theta)$ に加え，テンプレート画像番号 id を整数表現し，これら 6 種類の姿勢パラメータを 1 次元的に並べた配列を抗体とする．

4.4.4　UNDX

図 4.4 に UNDX で生成される子の正規分布領域を示す．基本的に子 c_1，c_2 は親 p_1 と p_2 とを結ぶ線分の周辺に正規分布に従って生成され，親 p_3 は正規分布の標準偏差を決定するために補助的に用いられる．p_1 と p_2 との距離を d_1，p_1 と p_2 とを結ぶ主軸と p_3 との距離を d_2 とすると，主軸方向の標準偏差 σ_1 と，それ以外の軸方向の標準偏差 σ_2 は以下の式で表すことができる．

$$\sigma_1 = \alpha d_1, \tag{4.5}$$

$$\sigma_2 = \beta d_2 \sqrt{n_D}. \tag{4.6}$$

α，β は任意の定数であり，n_D は次元数である．c_1，c_2 は σ_1，σ_2 によって決まる正規乱数を用いて生成される．これらの処理を両親 p_1，p_2 に対し n_c 回適用し，子を $2n_c$ 個生成する．次世代に残す子抗体は，すべての子の中から親和度によるエリート選択とルーレット選択を用いて選ばれる．ただし，各テンプレート画像に付加された固有の ID の順序に意味はないため，id 遺伝子を除いたパラメータ $(t_x, t_y, s_x, s_y, \theta)$ に対してのみ UNDX を適用し，c_1，c_2 の id 遺伝子はランダムに決定するものとした．

4.5　実験

4.5.1　実験内容

従来手法であるバイナリ型 IA（IA_{bin}）と，id 遺伝子を組込まずテンプレート画像毎に独立して探索を行う整数型 IA（IA_{int}），id 遺伝子を組込み複数のテンプレート画像を同時に探索する整数型 IA（IA_{int}^{mt}）を用いて，書架画像中に存在する複数の巻号画像領域を探索し，探索結果より本手法の有効性を検証する．

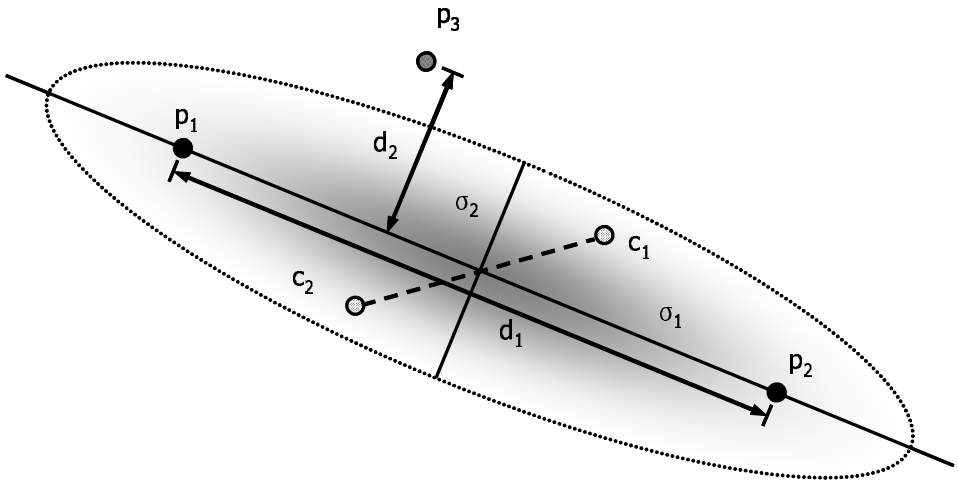


図 4.4 UNDX による子の生成

まず，3 枚のトレーニング用画像 $\mathcal{I}_1$，$\mathcal{I}_2$，$\mathcal{I}_3$ を用いてパラメータの調整を行い，次に，そのパラメータを用いて 3 枚のテスト用画像 $\mathcal{I}_4$，$\mathcal{I}_5$，$\mathcal{I}_6$ を対象とした評価実験を行う．それぞれの実験について 10 回の試行を行う．

4.5.2　実験条件

対象画像（929 × 697 画素）には 30 冊の書籍が配架された書架画像を用い，各書籍に存在する巻と号をそれぞれ探索する．すなわち，書架画像中に存在する合計 60 個の解探索に成功すれば解探索率が 100 % となる．このとき，対象画像内の巻号画像領域を示す，正解のパラメータ $(t_x, t_y, s_x, s_y, \theta)$ は局所的な全解探索により決定し，正解との類似度が閾値 T_a 以上の記憶細胞を解とする．

なお本実験では，$(t_x, t_y, s_x, s_y, \theta)$ が正解と類似し，id 遺伝子が正解と一致した解に加え，id 遺伝子が正解と異なるものの $(t_x, t_y, s_x, s_y, \theta)$ が正解と類似している解（ID 違いを含めた解と呼ぶ）についても記録を行った．これは，本章で使用した対象画像の解像度不足により，正解とは id 遺伝子が異なるが，より評価の高い解がその正解の近傍に存在したためである．

テンプレート画像（13 × 12 画素）には書架画像に存在する 15 種類の巻号画像を用いる．抗体数 N_v を 3,000 個，終了条件である親和度の評価回数を 3.0×10^7 回，記憶細胞数 N_m を 120 個とする．ただし，IA_{bin} と IA_{int} は 15 種類のテンプレート画像を独立して探索するため，抗体数 N_v を 600 個，終了条件である親和度の評価回数を 2.0×10^6 回とする．また，記憶細胞数 N_m は探索を行うテンプレート画像毎に決定し，各正解数の割合に応じて合計 120 個となるよう設定する．実験で用いたパラメータを表 4.1 に示し，対象画像 $\mathcal{I}_1$，$\mathcal{I}_5$ の正解をそれぞれ図 4.5(a)，(b) に示す．

表 4.1 実験で使用したパラメータ値

パラメータ	使用した値
総抗体数 N_v	3,000
総記憶細胞数 N_m	120
交叉率	100 %
突然変異率	0.1 %
閾値 T_{Π}	0.005
閾値 $T_{\pi 1}$，$T_{\pi 2}$	0.990
閾値 T_s，T_m	0.992
終了条件（親和度の総評価回数）	3.0×10^7
t_x の範囲	$0 \leq t_x < 929$
t_y の範囲	$0 \leq t_y < 697$
s_x の範囲	$80 \leq s_x \leq 120\%$
s_y の範囲	$80 \leq s_y \leq 120\%$
θ の範囲	$0 \leq \theta < 360$
$\omega_t : \omega_s : \omega_\theta$	45 : 1 : 2
閾値 T_a	0.994
UNDX の範囲 (α, β)	$(0.5, 0.35)$
UNDX の交叉回数 n_c	200
HC で生成する近傍抗体数 n_h	20
HC の繰返し回数 n_t	5
遺伝子数 g	10 bit

(a) Objective image $\mathcal{I}_1$　　(b) Objective image $\mathcal{I}_5$

図 4.5 対象画像と探索結果の例

4.5.3　実験結果と考察

$\mathcal{I}_1$ における抗体の進化と ID 違いを含めた解探索の推移を図 4.6 に示し，ID が一致した解探索の推移を図 4.7 に示す．また，他の対象画像の進化終了時の解探索率を図 4.8 に示す．

実験の結果，図 4.6 に示すように，探索開始直後は IA_{bin} の方が IA_{int} より早く解に収束しているが，母集団が進化するにつれて IA_{int} の方が効率的に複数の解を探索していることが

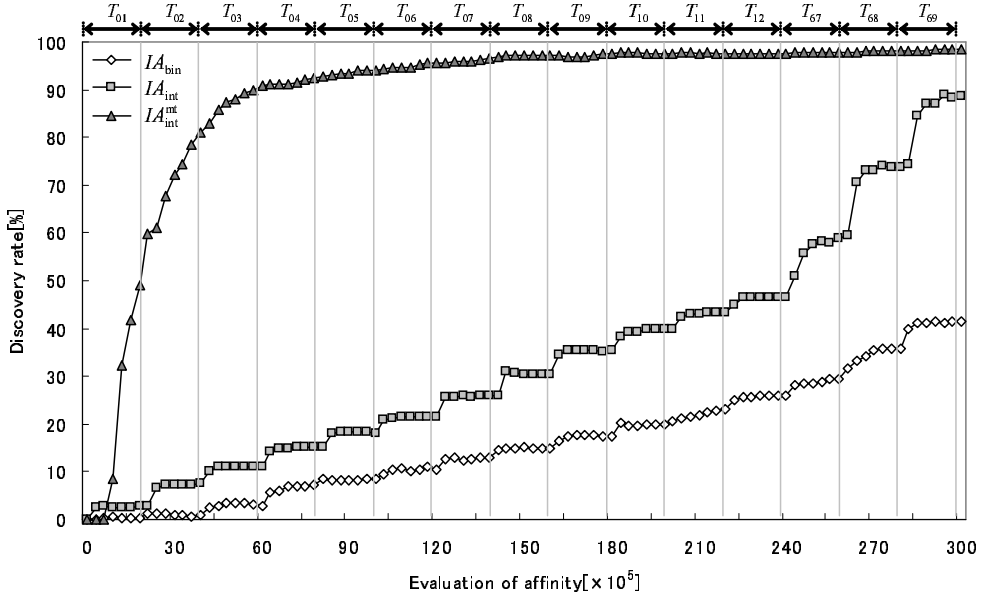


図 4.6 ID 違いを含めた解探索率の推移（$\mathcal{I}_1$）

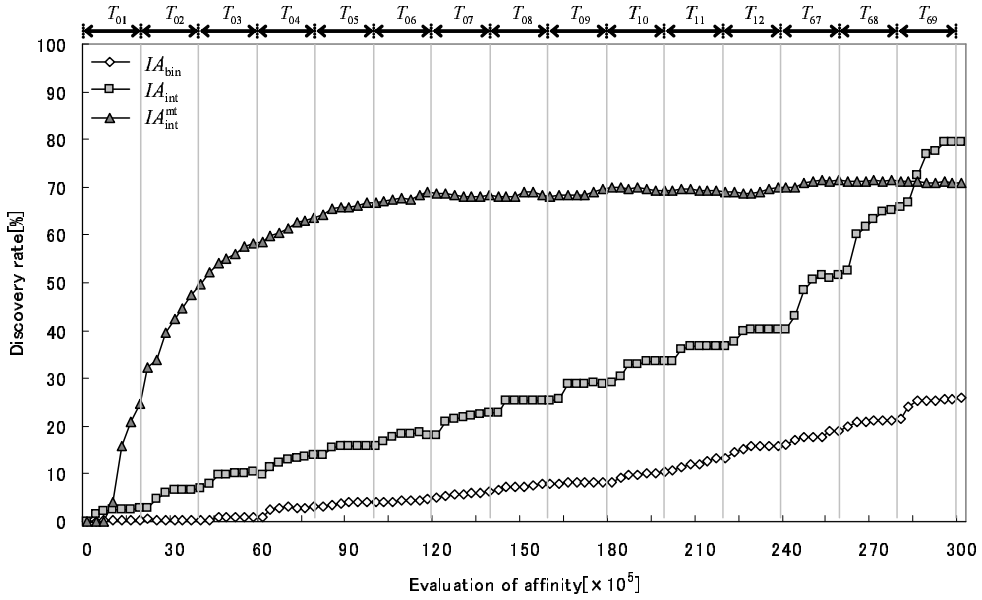


図 4.7 ID が一致した解探索率の推移（$\mathcal{I}_1$）

分かる．また，図 4.8 より，対象画像 $\mathcal{I}_1$ において最終的に得られた ID 違いを含めた解探索率は，IA_{bin} では 44.0% であるのに対して，IA_{int} では 89.0% であることが分かる．同様に，他の対象画像（$\mathcal{I}_2 \sim \mathcal{I}_6$）を用いた実験でも，$IA_{bin}$ より IA_{int} の方が 2 倍以上の高い探索率を得ることができた．これは，抗体の遺伝子のコーディング法と交叉方法の違いによるものだと考えられる．

IA_{bin} では，抗体の遺伝子がバイナリ型でコーディングされているため，得られる解の精度や探索空間の広がりを考慮して量子化する必要がある．今回の実験では，十分な精度の解

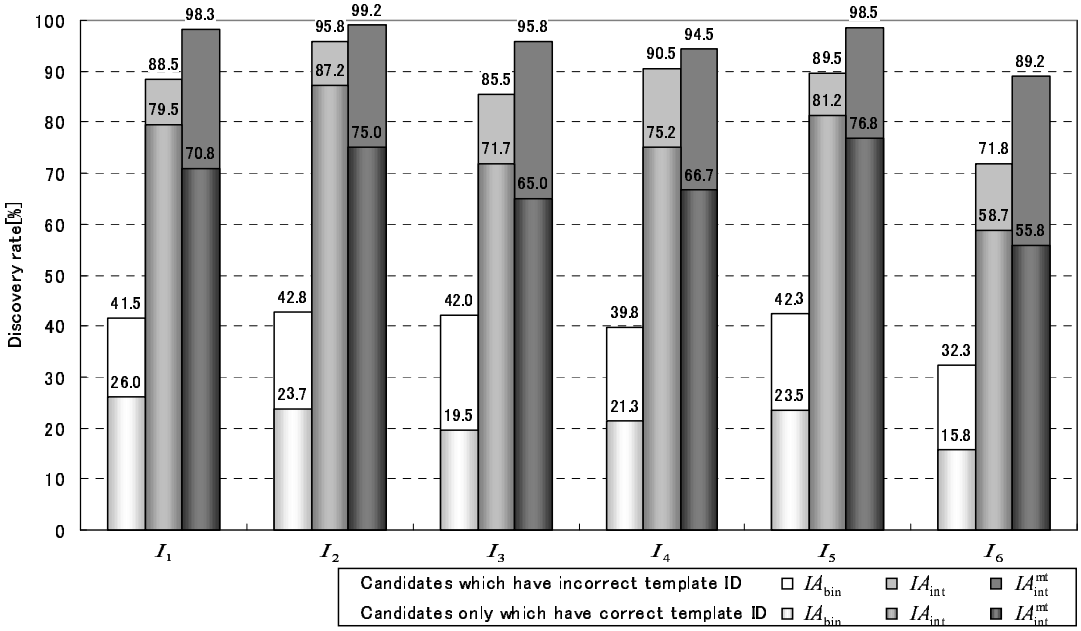


図 4.8 進化終了時の解探索率（$\mathcal{I}_1$）

を得るために抗体の各パラメータ $(t_x, t_y, s_x, s_y, \theta)$ を 10 ビットで表現したため，探索空間が余計に拡張され，探索に無駄が生じたことに起因するものと考えられる．

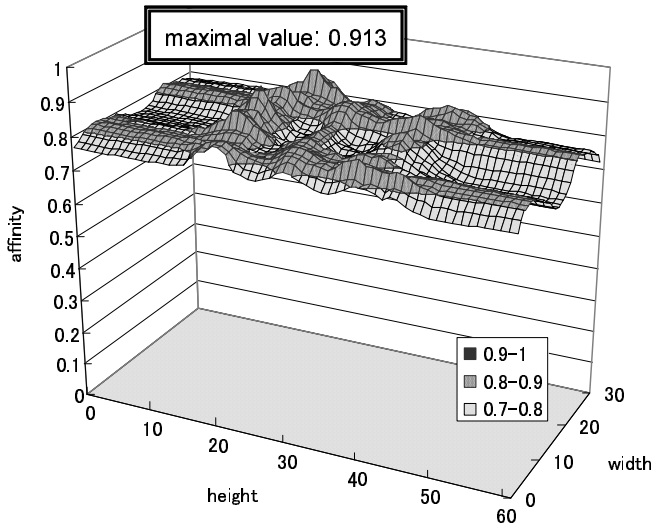
また，抗体の遺伝子型の空間と探索空間との位相構造が異なる場合，探索空間内において近い 2 つの抗体の遺伝子型が必ずしも類似しているとは限らず，IA_{bin} では交叉により有望な領域を重点的に探索できないことも要因であると考えられる．

よって，抗体の遺伝子を整数でコーディングし交叉方法に UNDX を用いることで，精度の高い解を効率的に探索できることを示した．

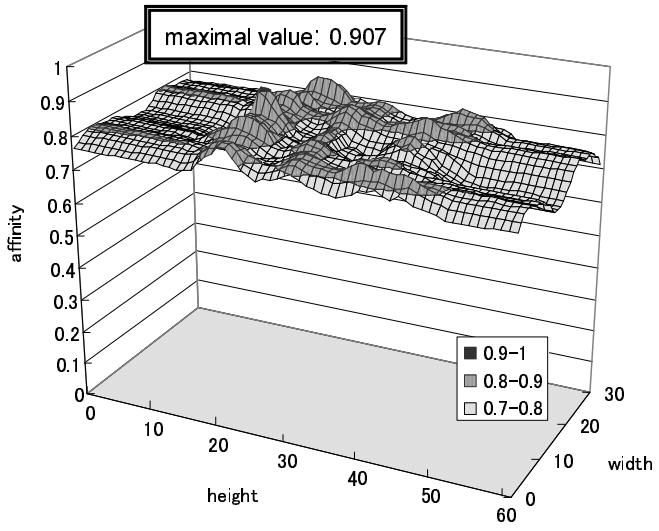
IA_{int} と IA_{int}^{mt} とを比較する．図 4.6 に示すように，ID 違いを含めた解を探索する場合，IA_{int} より IA_{int}^{mt} の方がより早い世代数で複数の解を発見できることが分かる．しかし，図 4.7 に示すように，ID が一致した解を探索する場合は，親和度の評価回数が約 1.2×10^7 回までは IA_{int}^{mt} の方がより多くの解を発見しているが，その後は逆転し，最終的に IA_{int} の方がより多くの解を得られた．また，全対象画像においても，図 4.8 より IA_{int} の方が高いことが分かる．これは，本実験で使用した画像の解像度不足と，記憶細胞への分化手法の違いによるものだと考えられる．

本章の実験では，実際に撮影した書籍の巻号をもとに正解の id 遺伝子を決定した．しかし，正解近傍の局所的な全解探索を行った結果，その正解とは異なる id 遺伝子を持ち，さらに評価の高い解が存在することを確認した．例えば，図 4.9 に示すように，正解の ID は「69 巻」であるにも関わらず,「69 巻」のテンプレート画像と類似した,「68 巻」のテンプレート画像で正解の近傍を探索した結果，正解より評価が高い解が存在した．

また，IA_{int}^{mt} では，記憶細胞への分化を行う際，記憶細胞の多様性を維持するため，すでに記憶細胞候補 v^* と類似した記憶細胞 m が存在する場合，その m が v^* で更新されるため，基本的に一つの解付近には一つの記憶細胞のみ分化される．しかし，IA_{int} では，テンプレー



(a) Fitness landscape of the template image “vol.68”



(b) Fitness landscape of the template image “vol.69”

図 4.9 正解「69 巻」付近の fitness landscape

ト画像毎に独立して探索を行うため，最終的に得られるテンプレート画像毎の記憶細胞を統合すると，同一の解付近に複数の記憶細胞が存在する場合がある．

これら 2 つの要因により，IA_{int} では ID が一致した解よりも評価の高い ID 違いの解の存在が許され，またそのような解の組が存在することを実験により確認した．一方，IA_{int}^{mt} では，正しい ID を持つが親和度の低い記憶細胞は進化の過程で，誤った ID を持つが親和度の高い記憶細胞によって更新されるため，ID が一致した解の探索率は IA_{int}^{mt} よりも IA_{int} の方が高くなったと考えられる．

図 4.8 より，全対象画像において，ID 違いを含めた解の探索率は IA_{int} より IA_{int}^{mt} の方が高

いことが分かる．また，図 4.6 より，IA_{int} より IA_{int}^{mt} の方が少ない世代数で複数の ID 違いを含む解を発見していることが分かる．よって，複数種類のテンプレート画像を同時に探索することで，いずれかのテンプレート画像と類似する画像領域を，より正確にかつ効率よく発見することができた．提案する方式単体では，テンプレートの特定は 70 % 程度の正解率となってしまうものの，文字の劣化を考慮した文字認識 [澤木 00] を後処理として併用することで，煩雑な前処理を必要としない書籍特定の実現が期待できる．

4.6　まとめ

本章では，抗体の遺伝子として，新たに id 遺伝子を組込んだ整数型 IA を提案し，書架画像中の書籍特定に応用することで，その有効性を検証した．

実験の結果，実際に探索を行う空間と遺伝子型の空間を一致させ，遺伝子のコーディング方法に則した交叉を適用することで，従来手法と比べて 2 倍以上の ID 違いを含む解を探索することができた．また，抗体に id 遺伝子を組込み複数種類のテンプレート画像を同時に探索することで，探索効率を向上することができた．

表 4.1 のように，IA が複数の局所的最適解を獲得するには適切なパラメータ設定が必要不可欠となる．そこで，パラメータ調整を容易にすることを目的とし TSP においてその有効性が証明されている，記憶獲得と際探索抑制を二種類の記憶細胞として独立させた適応的 IA が考案されている [當間 00]．今後は，適応的 IA への展開も必要であると考えられる．

第 5 章

ジョブショップスケジューリング問題での免疫アルゴリズムにおける螺旋交叉の検討

5.1 はじめに

近年，進化的アルゴリズムと量子系とが融合した確率的な組合せ探索アルゴリズムに関する研究が盛んに行われている．量子風進化的アルゴリズム（Quantum-Inspired Evolutionary Algorithm: QEA）[Han 02, Han 03, 中山 06c] は，$|0\rangle$ と $|1\rangle$ の量子重ね合わせ状態である量子ビット（Qubit）を用いて個体の遺伝子を表現した手法であり[*1]，大局的探索から局所的探索へと自動的に移行する特徴がある．また，1 個体のみによる探索が可能である．最近では，量子系の干渉効果を模擬してニューラルネットワークの重みに利用した量子風ニューラルネットワーク [Menneer 95, Kak 95, Ricks 03] や，高速にデータベースを検索するための Grover の量子アルゴリズムを利用し多くの想起パターンが設定できる量子風連想記憶 [Ventura 98]，画像探索で利用されるテンプレートマッチングに使われる相関関数に量子フーリエ変換を適用した量子風テンプレートマッチング [Curtis 03] などが提案されている．

その一つとして，量子系の干渉効果を模擬し，GA における交叉オペレータに利用した干渉交叉（Interference Crossover: IX）が，1996 年に Narayanan らによって提案されている [Narayanan 96]．彼らの研究では，巡回セールスマン問題（Traveling Salesman Problem: TSP）に干渉交叉を適用し，従来の GA と比較して，その有効性が示されている．一方，脊椎動物が持つ獲得免疫の仕組みを参考にした IA における抗体産生のオペレータにその干渉交叉を組込み，TSPLIB に含まれる eil01（101 都市配置 TSP）に対してその効果を調べたものであるが，従来の IA(Classical IA: 古典的 IA) と比較して約 3.2 倍の最適解発見率が得られ，IA における干渉交叉が最適解発見率向上の観点で有効であることが確認されている [中山 05]．さらに，この干渉交叉を量子系の干渉効果と見るのではなく，DNA の螺旋構造に由来する螺旋交叉（Helical Crossover: HX）として解釈する見方が提案されている [中山 06b] ．

本章では，文献 [中山 06b] の解釈のもと，干渉交叉を螺旋交叉と呼ぶことにし，TSP で

[*1] 個体の各遺伝子は，$|0\rangle$ と $|1\rangle$ の 2 つの状態を同時に重ね合わせた状態でもつ．$|0\rangle$ と $|1\rangle$ が観測される確率は各ベクトルの複素確率振幅によって決定される．

その有効性が確認されている螺旋交叉をジョブショップスケジューリング問題（Job-shop Scheduling Problem: JSP）に適用して，螺旋交叉が TSP に特化したものになっていないことを IA を対象とした計算機実験を通して明らかにする．

以下では，IA における干渉交叉法とその螺旋構造的解釈について述べ，JSP への適用実験を通して IA における螺旋交叉法の探索性能を評価し，考察を加える．

5.2　ジョブショップスケジューリング問題

ジョブショップスケジューリング問題の概要

ジョブショップスケジューリング問題（JSP）とは，複数の仕事を複数の機械に割り当てる際，すべての仕事を処理し終えるまでの総所要時間（makespan）を最小にする各仕事の処理順序を決定する問題である．前提条件として，技術的順序（technological ordering）すなわち各仕事を処理する機械の順序と，各仕事の各機械上での加工時間（processing time）は与えられているものとする．ただし,「各機械の種類はすべて異なり，複数の仕事を処理することはできない」また,「各仕事は与えられた処理時間をかけて，各機械上で中断されることなく処理される」という制約がある．

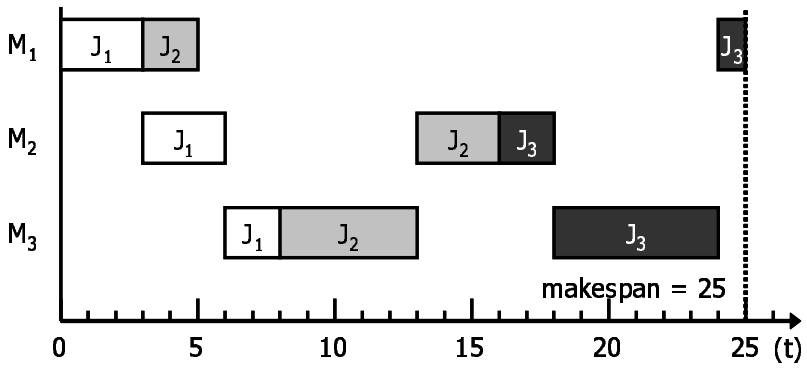
探索対象とするケジュール

JSP の制約条件を満たす実行可能なスケジュールは無数に存在する．例えば，図 5.1(a) に示すセミ・アクティブスケジュール（Semi-Active Schedule）は，各機械について仕事の処理順序（$J_1 \rightarrow J_2 \rightarrow J_3$）を変えず，できる限り無駄な時間を省いたスケジュールである．このセミ・アクティブスケジュールは，図 5.1(b) に示す仕事を繰り上げる操作（Left shift）等によるスケジュールのアクティブ化が解の改善に結びつく場合が多い．一方で，アクティブスケジュール（Active Schedule）は，仕事の処理順序を変えること（アクティブ化）を許して，できる限り無駄な時間を省いたスケジュールである．最適解は必ずアクティブスケジュールに含まれるため，本研究ではアクティブスケジュールを対象として探索を行う．アクティブスケジュールを対象とすることで解探索空間を狭めることができる．図 5.1 は，それぞれ，各仕事における全作業の処理開始から終了までの時間を帯状に示したガントチャートと呼ばれるグラフである．以下，アクティブスケジュールを生成する手法について述べる．

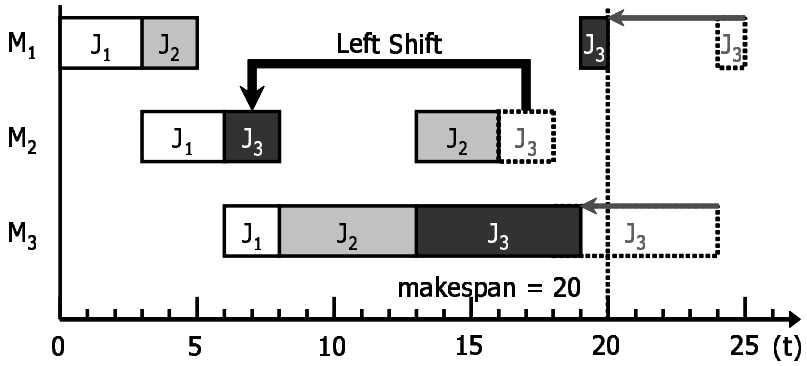
GT 法によるアクティブスケジュールの生成

すべてのアクティブスケジュールを作成する方法として GT（Giffer & Thompson）法 [Giffer 60] がある．以下に GT 法のアルゴリズムを示す．

Step 1 すべての仕事がスケジュールされていないものとする．ここで，スケジュールするとは，各仕事の処理開始時刻および終了時刻を決定することをいう．



(a) Semi-Active Schedule.



(b) Left shift.

図 5.1 セミアクティブスケジュールのアクティブ化.

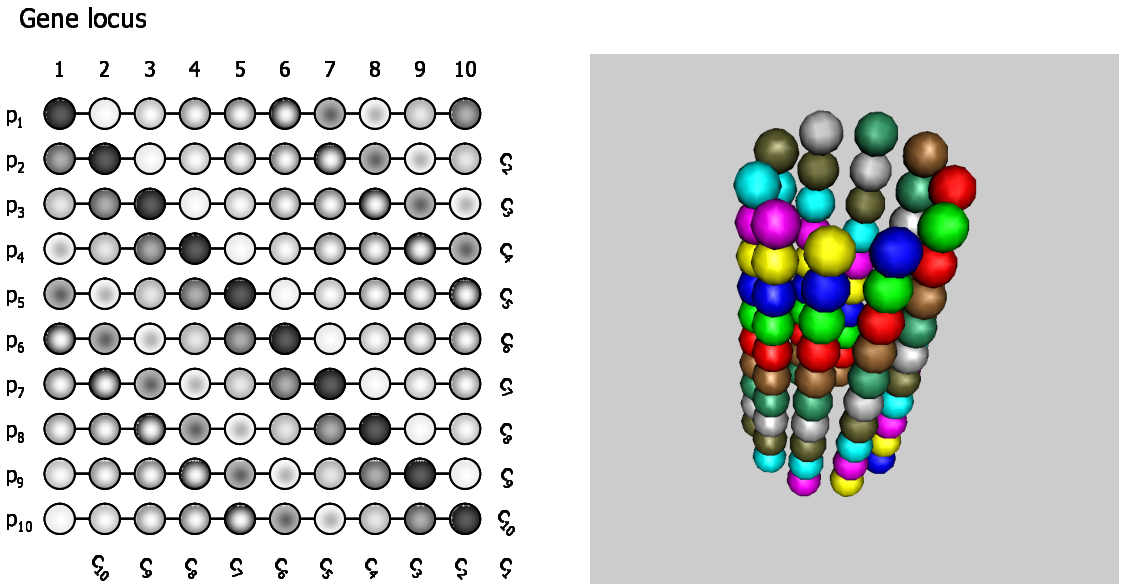
Step 2 まだスケジュールされていない仕事の中で，技術的順序に従い，次に処理開始可能な仕事の集合 C を作成する．集合 C の中で，できる限り早く処理を開始したとき，最も早く処理が終了する仕事 J^* を選ぶ．また，J^* を処理する機械を M^* とする．

Step 3 集合 C の中で，M^* 上で処理され，かつ J^* と処理が重なる仕事の集合 G（コンフリクト集合）を作る．コンフリクト集合の要素が複数個ある場合，コンフリクトが起こるという．

Step 4 コンフリクト集合 G の中から任意に一つの仕事を選び，次に処理する仕事としてスケジュールする[*2]．

Step 5 すべての仕事がスケジュールされるまで Step 2 から Step 4 を繰り返す．

[*2] このとき，すべての仕事について考えると，全アクティブスケジュールが生成される．また，ランダムに仕事を選択することで任意のアクティブスケジュールが得られる．



(a) IX.　　(b) HX.

図 5.2 IX および HX.

5.3　ジョブショップスケジューリング問題への適用

5.3.1　干渉交叉法とその螺旋構造的解釈

Narayanan らが提案した干渉交叉は，集団内のすべての抗体の中からいくつかの親を選び，それらの遺伝子情報を並べて斜め右下方向に 1 遺伝子ずつ抽出することで，子を産生するものである [Narayanan 96]．その様子を図 5.2(a) に示す．この図は遺伝子数が 10 で，干渉交叉の対象となる親数が 10 個の場合の例である．各行は干渉交叉前の親の遺伝子情報であり，1 遺伝子を表すセルの濃淡のそれぞれの集合は干渉交叉後に新たに産生される子の遺伝子情報を表している．すなわち，干渉交叉によって新しく産生される 1 番目の子 c_1 は親集団内の親 p_1 の遺伝子座 1 の遺伝子，親 p_2 の遺伝子座 2 の遺伝子，...，と抽出していき，最後に親 p_{10} の遺伝子座 10 の遺伝子を繋ぎ合わせたものとなる．2 番目の子 c_2 以降も，干渉交叉の開始位置を親 p_2 以降に替えて，同様の方法で産生する．つまり，2 番目の子は，親 p_2 の遺伝子座 1 の遺伝子から親 p_{10} の遺伝子座 9 の遺伝子までと，最初に戻って親 p_1 の遺伝子座 10 の遺伝子によって構成される．

ここで，干渉交叉の対象となる抗体数は Narayanan らによると抗体の遺伝子数と等しい場合にのみ述べられているが [Narayanan 96]，抗体数が遺伝子数より大きくても小さくても実行可能である．また，このような干渉交叉は抗体がコヒーレントな（可干渉性の）状態となり，多くの抗体間で干渉していることになる．

このように干渉交叉では，干渉対象の遺伝子の抽出は，最下段の親まで来れば，最上段の親にしている．これを連続的であると考えると，図 5.2(b) のように各親を管状の側面に中心軸に平行して順に配置すれば連続的になる．そして，干渉交叉と同様に遺伝子を抽出すると，

子が螺旋状に配置することになり，これがあたかも DNA の螺旋構造のように見えることから，干渉交叉の螺旋構造的解釈が可能となる．

5.3.2　処理の流れ

Step 1〔初期化〕

JSP では，各機械上で処理すべき作業の順序とその処理時間が抗原となる．ランダムに初期世代（$t = 0$）の母集団 $P(0)$ を生成する．抗体には，各機械について作業の順序と開始時刻を定めたガントチャートをコーディングする．

Step 2〔評価〕

抗原と抗体 v の親和度 Φ_v は式 (5.1) で定義する．ただし，M_v は抗体 v の総処理時間（makespan）である．

$$\Phi_v \quad = \quad 1/M_v. \tag{5.1}$$

Step 3〔終了判定〕

予め指定された進化の終了条件を満たしていれば Step 8〔進化終了〕へ，そうでなければ Step 4〔濃度計算〕へ進む．

Step 4〔濃度計算〕

まず，抗体 v と抗体 w の類似度 Ψ_{vw} を式 (5.2) で定義する．ただし，H_{vw} は抗体 v，w の各遺伝子座に共通した遺伝子の数，N_g は個体の遺伝子数である．

$$\Psi_{vw} \quad = \quad H_{vw}/N_g. \tag{5.2}$$

次に，類似度 Ψ_{vw} が閾値 $T_{\pi 1}$ 以上の抗体 v，w を類似抗体とみなし，各抗体 v の濃度 Θ_v を求める．

Step 5〔分化〕

Step 4 で求めた濃度 Θ_v の高い（閾値 T_Θ を超えた）抗体 v を記憶細胞候補 v^* とする．記憶細胞が上限数 N_m に達するまでは，候補 v^* と現時点の記憶細胞 m との最も高い類似度 Ψ_{v^*m} が閾値 T_Ψ^m 未満であるときのみ候補 v^* を記憶細胞 m に分化させる．記憶細胞が上限数 N_m に達した後は，候補 v^* との類似度 Ψ_{v^*m} が最も高い記憶細胞 m の親和度 Φ_m と候補 v^* の親和度 Φ_{v^*} を比較し，$\Phi_m < \Phi_{v^*}$ のときのみ当該記憶細胞 m を候補 v^* で更新する．

次に，記憶細胞候補 v^* と同じ遺伝子を持つサプレッサー T 細胞 s を分化させ，サプレッサー T 細胞 s との類似度 Ψ_{vs} の高い（閾値 $T_{\Psi 1}^s$ 以上の）抗体 v を消滅させる．その後，消滅した抗体に代わる新しい抗体を Step 2 と同じ方法で生成し，親和度と類似度を評価する．

Step 6〔選択〕

母集団 $P(t)$ 内で親和度 Φ_v が低い $N_v/2$ 個の抗体を淘汰し，残った抗体について式（2.4）で定義される期待値 E_v を計算する．

Step 7〔生殖〕

本実験では，交叉法に量子系の干渉効果を模擬した螺旋交叉と，JSP に対して従来から用いられている Inter-Machine JOX（以下，古典的交叉（Classical Crossover: CX）と呼ぶ）を，それぞれ r_{HX}，r_{CX}（$r_{HX} + r_{CX} = 1.0$）の割合で適用させる．つまり，期待値 E_v に比例する確率分布を用いて重複を許して $(N_a/2 \times r_{CX})$ 組の親抗体のペアを選び，古典的交叉により $(N_a/2 \times r_{CX})$ 個の子抗体を産生し，さらに，$(N_a/2 \times r_{HX})$ 個の親抗体をランダムに選び，螺旋交叉により $(N_a/2 \times r_{HX})$ 個の子抗体を産生する．次に，両交叉で産生された合計 $N_a/2$ 個の子抗体に突然変異のオペレータを突然変異率 r_M により決まる確率で作用させ，Step 6 で淘汰された分の抗体を補充する．その後，Step 2 へ戻る．

Step 8〔進化終了〕

予め指定された条件を満足したため，進化を終了する．

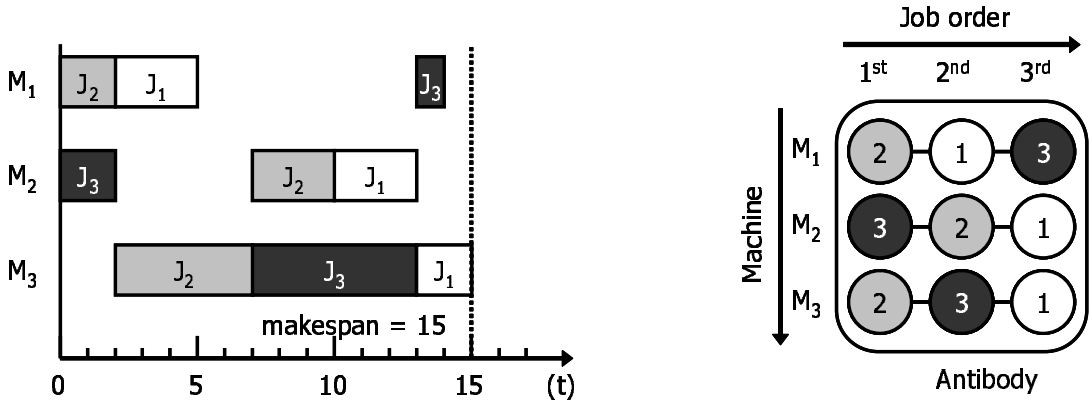
5.3.3　JSP における遺伝子コーディング

以降に述べる実験で用いた遺伝子のコーディング法はガントチャートをもとにした各機械における遺伝子の投入順序を順列で表し，2 次元配列として表現したものを遺伝子とする方法である．例えば，仕事数が 3 個，機械数が 3 台である 3 仕事 3 機械 JSP の各仕事について作業順序と作業時間が，それぞれ次式で与えられているとする．

$$O^M = \begin{array}{c} \\ J_1 \\ J_2 \\ J_3 \end{array} \begin{array}{c} \begin{array}{ccc} \text{1st} & \text{2nd} & \text{3rd} \end{array} \\ \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \\ 2 & 3 & 1 \end{pmatrix} \end{array}, \tag{5.3}$$

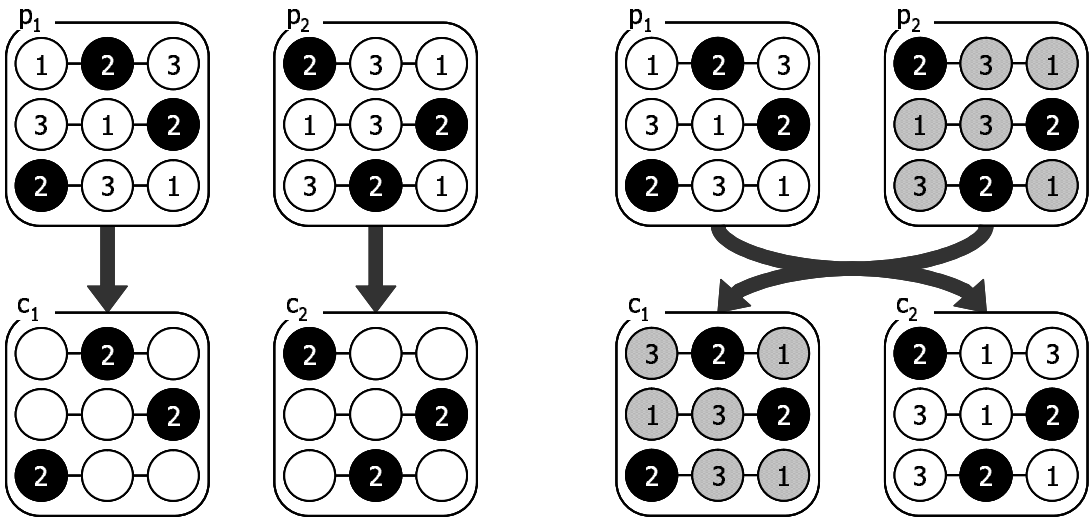
$$O^T = \begin{array}{c} \\ M_1 \\ M_2 \\ M_3 \end{array} \begin{array}{c} \begin{array}{ccc} J_1 & J_2 & J_3 \end{array} \\ \begin{pmatrix} 3 & 2 & 1 \\ 3 & 3 & 2 \\ 2 & 5 & 6 \end{pmatrix} \end{array}. \tag{5.4}$$

ここで，行列 O^M において j 行 k 列の要素は，仕事 J_j の k 番目の作業を処理する機械番号を表す．また，行列 O^T において i 行 j 列の要素は，機械 M_i での仕事 J_j の作業時間を表す．上述の制約条件を満たす JSP の解は，各機械について作業の順序と開始時刻を定めて，例えば図 5.3(a) のようなガントチャートとして図示できる．そして，図 5.3(b) がこのガントチャートに対する遺伝子型となる．



(a) Gantt chart.　(b) Gene coding.

図 5.3 ガントチャートおよび JSP における抗体遺伝子のコーディング



(a) Select jobs to keep.　(b) Swap no-selected jobs.

図 5.4 従来の交叉法（Inter-Machine JOX）

この遺伝子型に対する交叉法として，JSP に対して従来から用いられている Inter-Machine JOX [小野 98] の操作を図5.4に示す．なお，本研究で対象としている螺旋交叉と区別するため，このような従来から用いられている交叉をここでは総じて古典的交叉（Classical Crossover: CX）と呼ぶ．Inter-Machine JOX は，各機械における仕事の投入順序をなるべく保存しながら子を生成する交叉法である．まず，仕事をランダムに複数個選択する．このとき，それぞれの仕事が選択される確率は 1/2 であり，選択された仕事は全機械で共通とする．図 5.4(a) は J_2 のみが選択された例である．この選択された仕事については，親 p_1 から子 c_1 へ，親 p_2 から子 c_2 へ投入位置が引き継がれる．一方，選択されなかった仕事については，図 5.4(b) に示すように，各機械ごとに親 p_1 から子 c_2 へ，親 p_2 から子 c_1 へ作業順序を保持するように左

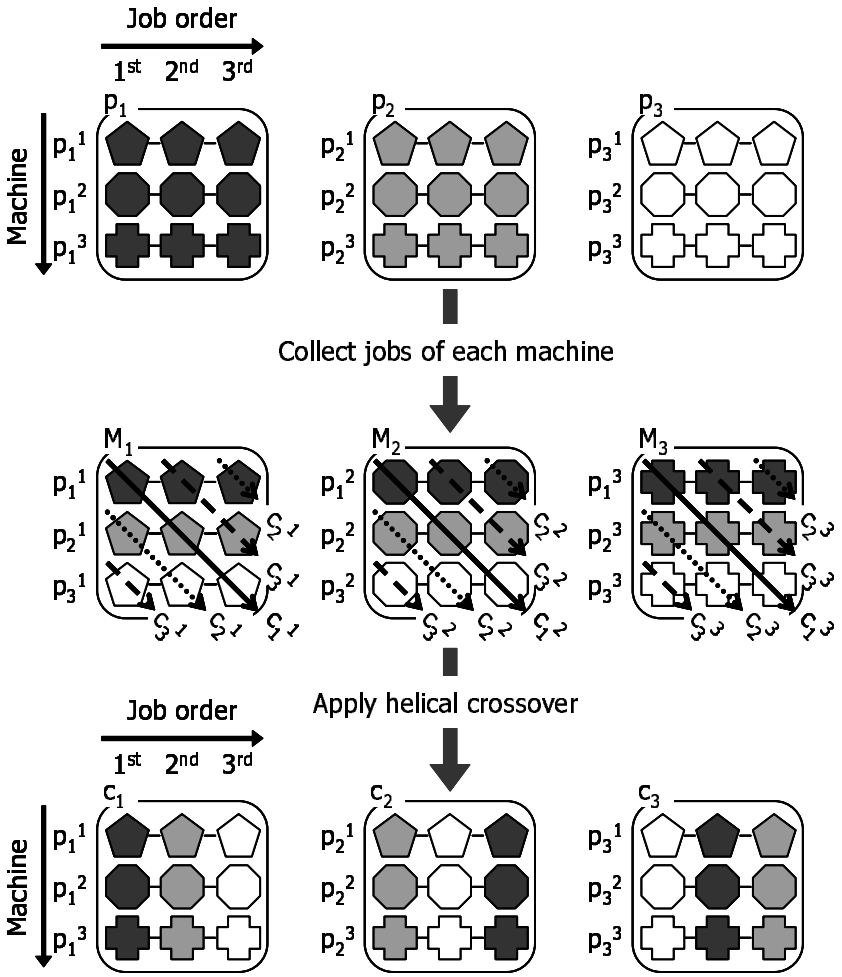


図 5.5 JSP における HX の適用

から右へ空いている投入位置（遺伝子座）へ挿入することとなる．

5.3.4　JSP への螺旋交叉法の適用方法

ここでは，このような 2 次元配列表現を抗体の遺伝子型としているため，抗体間での螺旋交叉は機械ごと，つまり遺伝子型として 2 次元配列の各行ごとに遺伝子情報を集め螺旋交叉を適用する．親 p_k の i 行目つまり機械 M_i における仕事の投入順序に関する遺伝子情報を p_k^i，同様に子 c_k の i 行目を c_k^i とすると，前述の 3 仕事 3 機械 JSP における螺旋交叉は図 5.5 のようになる．まず，親 $p_k(k = 1, 2, 3)$ の遺伝子情報を機械ごとにまとめる．次に機械 M_1 に関する遺伝子情報のうち，p_1^1 の遺伝子座 1 から斜め右下方向に遺伝子を順に抽出していき，子 c_1 の 1 行目の遺伝子情報 c_1^1 を生成する．機械 M_2，M_3 についても同様にして，p_1^2，p_1^3 の遺伝子座 1 からそれぞれ斜め右下方向に遺伝子を順に抽出していき，c_1^2，c_1^3 を生成し，これらの集合が新たな子 c_1 となる．子 c_2，c_3 についても，機械 M_i に関する螺旋交叉の開始位置をそれぞれ p_2^i，p_3^i の遺伝子座 1 に替えて同様に処理する．

なお，JSP では次に来るべき遺伝子情報がすでに産生中の新たな子に含まれていた場合，

表 5.1 実験で使用したパラメータ値

パラメータ	使用した値
総抗体数 N_v	200
総記憶細胞数 N_m	5
終了条件（進化させる世代数）$t_{\max}$	10,000
CX 率 r_{CX}	1.0 to 0.0
HX 率 r_{HX}	0.0 to 1.0
突然変異率 r_M	1 %
閾値 T_{Π}	0.7
閾値 $T_{\pi 1}$，$T_{\pi 2}$，T_s，T_m	0.8

2 度目に現れた重複仕事名の昇順で次の仕事名が代わりとなり，それを遺伝子情報として採用することとする．つまり，2 度目に現れた重複仕事名が例えば “J_2” であったならば “J_3” を代わりとし，“J_3” も既に使われていたならば，“J_4” とし，最後の仕事名までいってもなければ最初の仕事名 “J_1” に戻ることになる．

5.4　実験

5.4.1　実験内容

中山らは，先行研究として，螺旋交叉を IA における抗体産生のオペレータに組込み，TSP を対象とした数値実験によりその有効性を確認した [中山 05] ．ここでは，螺旋交叉法が TSP に特化したものになっていないことを確認するために，対象問題を JSP に置き換えて実験を行うものとする．実験方法は，図 2.8 における Step6〔選択〕で淘汰された $N_v/2$ 個（N_v は全抗体数）の抗体を補充する Step7 〔生殖〕において，古典的交叉で新たな交叉を産生する割合 r_{CX} と，螺旋交叉で産生する割合 r_{HX} を，それぞれ 1.0 から 0.0 まで-0.1 の間隔，0.0 から 1.0 まで+0.1 の間隔で変化させるものとして，各条件ごとに 50 回試行したときの最適解発見率と平均総所要時間（平均メイクスパン）を評価した．なお，古典的交叉は前述の Inter-Machine JOX，突然変異は Job-Based Shift Change [小野 98] を採用し，各抗体の親和度を算出する際に必要となるデコーディング法には GT 法 [Giffer 60] を用いた．対象はベンチマーク問題の Fisher & Thompson の 10 仕事 10 機械 JSP である ft10（最適総所要時間は 930）とし，最大世代数に達するか最適解を発見できた時点で探索を終了するものとした．実験で用いたパラメータ値を表 5.1 に示す．

5.4.2　実験結果と考察

実験結果を図 5.6 に示す．最適解発見率（棒グラフ）は試行回数あたりの最適解発見率を，平均総所要時間（折れ線グラフ）は各試行における最短所要時間の平均である．なお，IA は

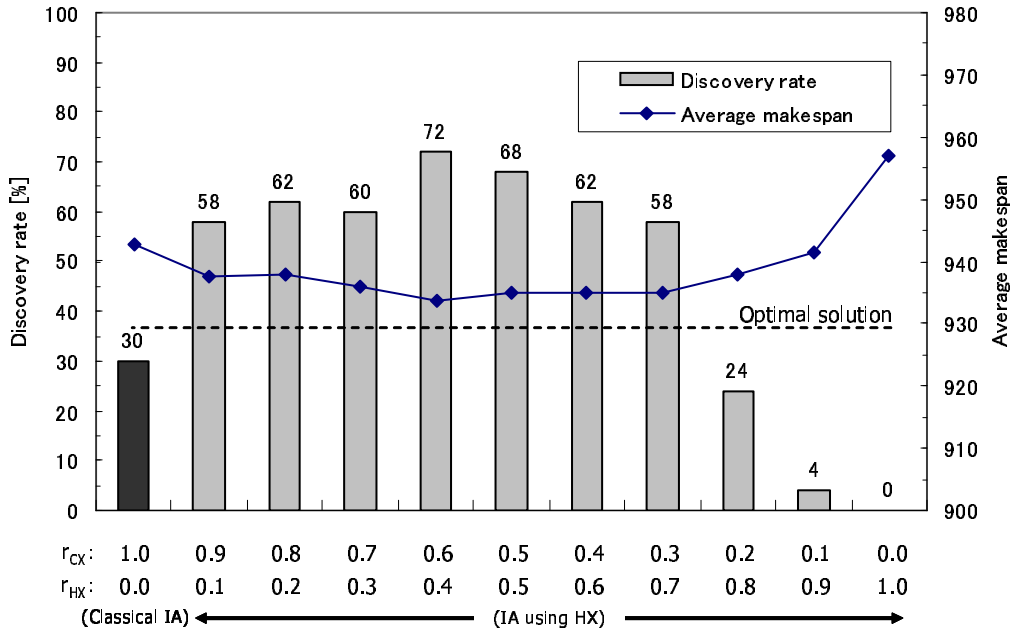


図 5.6 最適解発見率と平均総所要時間による比較

記憶細胞が発見した解の集合であるため，平均総所要時間には記憶細胞に対する平均総所要時間を示すものとする．

この結果から，螺旋交叉を組込んだ IA は，最適解発見率および平均総所要時間の観点で，古典的交叉のみの場合に比べて優れていることが分かる．特に，交叉率が $(r_{CX}, r_{HX}) = (0.6, 0.4)$ のときの最適解発見率は 72% で，古典的交叉のみの 30% に比べ 2.4 倍の最適解発見率の向上が確認できる．さらに，平均総所要時間でも $(r_{CX}, r_{HX}) = (0.6, 0.4)$ のとき約 933.54 で，古典的交叉のみの約 942.91 の場合に比べて短縮できていることが確認できる．これらは，螺旋交叉が抗体の多様性を維持し，局所的最適解からの脱出に貢献できた結果であると考えられる．

このように，螺旋交叉を組込むと，古典的交叉のみの場合に比べ探索能力の向上が認められるが，螺旋交叉の割合 r_{HX} が高過ぎるとその効果が劣ることも分かる．これは，螺旋交叉による過剰な多様化によるものと思われ，螺旋交叉を有効に組込むためには，適切な r_{HX} の設定が必要である．

5.5　まとめ

本章では，TSP でその有効性が確認されている螺旋交叉法を JSP に適用し，IA における螺旋交叉法が TSP に特化したものになっていないという実証実験の結果について述べた．10 仕事 10 機械 JSP の ft10 を用いた実験に対して調べたものであるが，IA における螺旋交叉法は JSP に対しても有効に作用し，古典的 IA と比較して 2.4 倍の最適解発見率が得られた．今後は，JSP の他のベンチマーク問題に適用し，螺旋交叉法の効果をより詳細に分析していく必要がある．

第 6 章

免疫アルゴリズムのためのImmune言語の開発

6.1 はじめに

組合せ最適化問題とは，数多くの組合せの中から最適な解を求める問題であり，遺伝的アルゴリズム（Genetic Algorithm: GA）などの近似解法を用いることによって，近似的な最適解（準最適解）を可能な限り高速に求める研究がなされている．この GA は，進化の過程において集団内で同じ個体が急増するなどして，集団の多様性が失われる過剰な収束 [飯村 03, 飯村 05] という探索を停滞させる現象が生じる場合がある．それに対して，大局的最適解を含む複数の局所的最適解を得ることを目的とした免疫アルゴリズム（Immune Algorithm: IA） [森 97, 飯村 05, 三井 04] がある．この IA は，生体の免疫システムを模倣したアルゴリズムであり，多様性のある抗体産生機構と，必要以上に増え過ぎた類似抗体の産生を抑制する自己調節機構により多様性のある抗体を産生し，過剰収束の回避を期待できる．しかし，IA は GA 同様，選択や交叉，突然変異などの遺伝的操作に関する知識が必要であり，それらをシミュレートするための煩雑なコーディング作業をプログラム開発者が行う必要がある．コーディングの煩雑さを軽減する策としては，IA 用のクラスを開発しそれらをプログラム開発者に提供する，または IA 専用の DDL (Dynamic Link Library) を開発し提供するなどが容易に考えられる．しかしながら，Java 言語が詳しくないと，IA 用のクラスタの使用や専用の DDL さえも IA が複雑なために難しいと思われる．

一方，本研修室では，これまでに，コーディングの煩雑さを軽減し，一般ユーザでも容易に利用できる環境構築を目的として，GA に特化した Gene 言語 [吉永 05] と，分散並列化のための Espace 言語 [原 03, 岩川 06]，アントコロニー最適化法に特化した Ant 言語 [森山 07] の開発を行ってきた．そこで本章では，IA のコーディングの煩雑さを軽減し，一般ユーザでも IA を容易に適用できる環境構築を目的とした IA のための Immune 言語 [森山 06] を提案する．そして，組合せ最適化問題の代表例であるナップザック問題（Knapsack Problem: KP）と巡回セールスマン問題（Traveling Salesman Problem: TSP）に対して実際に適用することにより，Immune 言語の効果を検証する．提案する Immune 言語は，Java 言語に IA

処理に関するキーワードを追加したもので，Java 言語の基本的な知識さえあれば容易に IA を取り扱うことができる．

GA と IA とを比較すると，その起源が異なるが，計算機に実装した場合，基本的な処理過程は類似した部分が多いという特徴があるため，Gene 言語同様，Espace 言語 [原 03, 岩川 06] との融合により，グリッドコンピューティングのような分散並列処理を加えて分散並列 IA に拡張可能な Immune 言語への展開も期待できる．

以下，IA を KP と TSP へ適用した場合の処理手順と，提案する Immune 言語の詳細およびコーディング事例について述べる．その後，Immune 言語を KP と TSP へ適用した場合の実験結果を示し，その結果をもとに考察を加える．

6.2　ナップザック問題と巡回セールスマン問題への応用

6.2.1　ナップザック問題とその定式化

ナップザック問題

まず，実験対象とした組合わせ最適化問題の 1 つであるナップザック問題（Knapsack Problem: KP）について述べる．いくつかの荷物があり，それぞれの重量および価値が決めれられているものとする．その際，許容重量が決められたナップザックに荷物を入れるとき，価値の総和が最大となる荷物の組合せを選ぶ問題を KP という．

ナップザック問題の定式化

荷物の個数を n，各荷物 i（$i = 1, 2, \cdots, n$）の重量を $wt_i (wt_i > 0)$，価値を $vl_i (vl_i > 0)$，またナップザックの許容重量を $cw (cw > 0)$ とすると，KP は次のように定式化される．

$$\max_{x_i} \quad \sum_{i=1}^{n} vl_i x_i, \tag{6.1}$$

$$\text{subject to} \quad \sum_{i=1}^{n} wt_i x_i \leq cw, \tag{6.2}$$

$$x_i \in \{0, 1\} \ (i = 1, 2, \cdots, n). \tag{6.3}$$

ただし，決定変数 x_i については，荷物 i をナップザックに入れることを $x_i = 1$，入れないことを $x_i = 0$ で表すものとする．

IA を KP に適用する場合，単に n 個の荷物の有無を並べた 0，1 の文字を対立遺伝子として実現できる．つまり抗体の染色体 C は，

$$C = (x_1 \ x_2 \ \cdots \ x_n), \tag{6.4}$$

となる．

6.2.2　巡回セールスマン問題とその定式化

巡回セールスマン問題

次に，巡回セールスマン問題（Traveling Salesman Problem: TSP）について述べる．都市間の距離[*1]が決められたいくつかの都市が存在するものとする．あるセールスマンが各都市を一度ずつ巡回し，巡回路長（総距離）が最短となるような巡回路を決定する問題が TSP である．ただし，セールスマンは必ず出発した都市に戻るものとする．

巡回セールスマン問題の定式化

都市数を n，都市には 1 から n までの番号がつけられているものとし，また都市 i，j 間の距離を $d_{i,j}$ とすると，TSP は次のように定式化される[*2]．ただし，TSP を定式化する方法としては，巡回路を構成する辺を変数にする方法と，巡回路上の都市の訪問順序を変数とする方法があるが，ここでは，後者の方法で定式化する．決定変数として都市の訪問順序を $C = (c_1\ c_2\ \cdots\ c_n)$（$c_i$ は i 番目に訪問する都市番号）で表すと，問題は，

$$\min_{C} \quad \sum_{i=1}^{n} d_{c_i,c_{i+1}}, \tag{6.5}$$

と表される．ただし，訪問順番 i についての増加操作は，巡回路であるため n の次は 1 であるものとする．すなわち，c_n の次の都市は c_1（$= c_{n+1}$）となる．

GA における TSP では，この定式化に基づき，n 個の都市を訪問順に並べた相異なる n 個の整数を対立遺伝子として実現できる．つまり染色体 C は，

$$C = (c_1\ c_2\ \cdots\ c_n), \tag{6.6}$$

となる．しかしながら，単純に対立遺伝子として n 個の整数を用いたパス表現（path representation）で染色体を構成すると，一点交叉等の遺伝的オペレータを適用することによって生じる子について，その染色体が相異なる n 個の整数からなる順列であるという条件を満たさないといった不具合が生じてしまう．そこで，任意の抗体が正当な巡回路に対応するように，順序表現（ordinal representation）などの遺伝子表現（コーディング）法や順序交叉（order crossover）などの遺伝的オペレータについて種々の工夫が報告されている．

6.2.3　KP と TSP における免疫アルゴリズムの処理手順

図 2.8 に示された IA の処理手順に従って，本実験で対象とした組合せ最適化問題の代表例である KP と TSP を用いて各処理の詳細を以下に説明する．実際の免疫システムでは，抗体産生細胞が遺伝子の再構成を行い多様性のある抗体を産生するが，本研究では抗体産生細

[*1] 都市間の重み（距離）は，セールスマンが移動に要する時間または費用と考えてもよい．

[*2] 都市間距離が $d_{i,j} = d_{j,i}$ を満たすときは対称 TSP（symmetric TSP），そうでないときは非対称 TSP（asymmetric TSP）と呼ばれる．

胞と抗体を同じ細胞とみなし，抗体自身が交叉や突然変異を繰返すことで，多様な抗体を産生するものとする．また，6.3.1 項で述べるように，親和度計算メソッドと類似度計算メソッドは，引数に細胞名と細胞番号を設定し汎用性を持たせる必要があるため，記憶細胞とサプレッサー T 細胞を，抗体と同様の遺伝子配列で表現する．

Step 1〔初期化〕

第 t 世代の抗体群が形成する母集団を $P(t)$ とする．まず，ランダムに初期世代（$t = 0$）の母集団 $P(0)$ を生成する．

KP の場合，抗体には全荷物についてナップザックに入れる（1）か入れない（0）かをバイナリ型でコーディングする．つまり，1 または 0 を一次元的に並べた配列を抗体とする．

また，TSP の場合，抗体には整数で表現された都市名を巡回する順番にコーディングする．つまり，都市名を一次元的に並べた配列を抗体とする．

Step 2〔評価〕

現在の母集団 $P(t)$ 内の各抗体 v に対して親和度 Φ_v を計算する．

KP の場合，親和度は抗体から荷物の組合せを読み出し，荷物の価値の合計を計算することで得られ，価値の合計が高い抗体ほど高くなる．

TSP の場合，親和度は抗体から都市の巡回路を読み出し，巡回路長を計算することで得られ，巡回路長が短い抗体ほど高くなる．

提案する Immune 言語では，親和度計算メソッドは問題によって異なるためにユーザ自身が定義するものとする．

Step 3〔終了判定〕

予め指定された進化の終了条件を満たしていれば Step 8〔進化終了〕へ，そうでなければ Step 5〔分化〕へ進む．

Step 4〔濃度計算〕

各抗体 v について $P(t)$ 内の他の抗体 w との類似度 Ψ_{vw} を計算する．

KP の場合，類似度は 2 つの抗体間に共通する荷物の個数をもとに計算され，共通する荷物が多いほど高くなる．

TSP の場合，類似度は 2 つの抗体間に共通する経路（枝）数をもとに計算され，共通した経路（枝）が多いほど高くなる．

提案する Immune 言語では，類似度計算メソッドは問題によって異なるためにユーザ自身が定義するものとする．

また，$\Psi_{vw} \geq T_{\pi 1}$ のとき v，w を類似抗体とみなし，式（2.2）で定義される濃度 Θ_v を求める．

Step 5〔分化〕

$\Theta_v \geq T_{II}$ のとき v は記憶細胞候補 v^* へ分化する．記憶細胞が上限数 N_m に達するまでは，v^* をそのまま記憶細胞 m に分化する．上限数 N_m に達した場合は，v^* と m との類似度 Ψ_{v^*m} が最も高い記憶細胞 m を選び出し，$\Phi_{v^*} > \Phi_m$ のときのみ v^* で m を更新する．v^* が m に分化した場合，v^* は類似抗体を抑制するサプレッサー T 細胞 s に分化する．$P(t)$ 内の各抗体 v とサプレッサー T 細胞 s との類似度 Ψ_{vs} を計算し，$\Psi_{vs} \geq T_s$ のとき v を $P(t)$ から消滅させる．その後，消滅した抗体に代わる新しい抗体を Step 1 と同じ方法で産生し，Step 2，Step 4 と同じ計算処理を行う．

Step 6〔選択〕

$P(t)$ 内で Φ_v の低い抗体を淘汰する．残った抗体について式（2.4）で定義される期待値 E_v を計算し，E_v に応じたルーレット選択により親抗体 p_1，p_2 を選び出す．

Step 7〔生殖〕

親抗体として選ばれた p_1，p_2 に交叉，突然変異を作用させ 2 つの子抗体を産生する．親抗体の選出と子抗体の産生を繰返し，淘汰された抗体と同じ数だけ補充し次世代の母集団 $P(t+1)$ を生成する．その後，Step 2〔評価〕へ戻る．

Step 8〔進化終了〕

予め指定された条件を満足したため，進化を終了する．

6.3　提案する Immune 言語の仕様およびそのコーディング事例

Immune 言語とは，Java 言語に `immune` キーワードによる `immune` 構文が追加定義されたものである．この `immune` 構文では，IA 特有の処理を自動補完するために必要な情報を指定することになる．Immune 言語で記述されたプログラムは，開発した Immune 言語コンパイラにより Java 言語のプログラムに変換され，その後 Java 言語コンパイラにより Java バイトコードとなる．なお，Immune 言語のような新たなプログラミング言語の提案は，分散並列処理の分野においては OpenMP や HPF (High Performance Fortran) など種々のものが提案されているが，IA に関しては筆者らの知る限りにおいて従来手法は存在しない．

以下，提案する Immune 言語の仕様について述べ，その後 Immune 言語による開発の流れについて述べる．

6.3.1　文法

Immune 言語では，IA の処理を開始するための情報と，対象とする組合せ最適化問題に依存する情報を記述し，それらの情報を `immune` 構文で指定する．Immune 言語コンパイラは，

この immune 構文が指定する情報をもとに IA の処理に必要なメソッド等を補完して Java 言語プログラムを自動生成する．

以下，Immune 言語で記述すべき「IA 処理開始メソッドの呼び出し」,「前提条件変数の定義」,「親和度メソッドの定義」,「類似度メソッドの定義」,「immune 構文による必要情報の指定」について述べ，Immune 言語の各文法を，今回評価の対象とした整数型遺伝子が使われる TSP とバイナリ型遺伝子が使われる KP を例に説明する．

IA 処理開始メソッドの呼び出し

IA の処理を開始するためのメソッドを呼び出す．ここではメソッドの呼び出しのみを行い，メソッド自体の定義は Immune 言語コンパイラが行う．

開始メソッドの引数には抗体の遺伝子数，抗体数，1 回の実行で更新される世代数，探索結果を保存するためのファイル名がある．世代数とファイル名は省略することができ，世代数が省略された場合は 1 回の実行で 1 世代ごとの更新となり，ファイル名が省略された場合は探索結果は保存されない．

抗体の遺伝子数を nGenes，抗体数を nAntibodies，更新される世代数を nSteps，探索結果を保存するためのファイル名を fName とし，開始メソッド名を start としたオーバーロード記述例を以下に示す．

```
//記述例 1
start(nGenes, nAntibodies, nSteps, fName);
//記述例 2
start(nGenes, nAntibodies, nSteps);
//記述例 3
start(nGenes, nAntibodies, fName);
//記述例 4
start(nGenes, nAntibodies);
```

前提条件変数の定義

組合せ最適化問題の前提条件は対象とする問題によって異なるため，前提条件を格納するための変数をユーザが定義する必要がある．例えば，KP の場合は各荷物の質量および価値が前提条件となり，TSP の場合は各都市の位置を表す x 座標，y 座標が前提条件となる．

前提条件は，親和度の計算処理と視覚的な探索結果の描画処理のために用いられる．

KP を対象とした int 型変数名を luggage，TSP を対象とした double 型変数名を city とした記述例を以下に示す．

```
//ナップザック問題の記述例
int luggage[][] = {{50, 10}, ... , {100, 70}};
//巡回セールスマン問題の記述例
```

```
double city[][] = {{37, 52}, ... , {30, 40}};
```

親和度メソッドの定義

親和度を計算する処理は，対象とする組合せ最適化問題の目的関数や前提条件に依存し，問題によって処理が異なるため，ユーザが問題に応じて定義する必要がある．例えば，KP の場合は重量と価値をもとに，TSP の場合は各都市間の距離をもとに親和度を算出することになる．

親和度メソッドは，特定の細胞に対して 1 世代当たり数回呼び出されるため，計算対象となる細胞情報が必要となる．そのため，親和度メソッドの引数に細胞名と細胞番号を設定する必要があり，Gene 言語とは異なり,「抗体を格納する変数」が省略できる．細胞名には抗体，記憶細胞またはサプレッサー T 細胞のうちいずれかの細胞を指定し，細胞番号には各細胞に割り振られている細胞の番号を指定する．ここで，細胞を格納する変数は，何番目の細胞の遺伝子座がいくつの遺伝子なのかを指定できるようにするために，配列の第 1 添え字を細胞番号，第 2 添え字を遺伝子座とした二次元配列で定義する．バイナリ型 IA の場合は `boolean` 型の二次元配列，整数型 IA の場合は `int` 型の二次元配列となる．

細胞名を `cName`，細胞番号を `nCell`，親和度の定義メソッド名を `setAffinity` とし，`double` 型の親和度を戻り値として返す記述例を以下に示す．

```
//ナップザック問題の記述例
public double setAffinity(boolean[][] cName, int nCell) {
  double affinity = 0.0d;
  int weight = 0;
  int value = 0;
  int limit = 550;
  for(int g = 0; g < nGenes; g++) {
    if(cName[nCell][g]) {
      weight += luggage[g][0];
      value += luggage[g][1];
    }
  }
  affinity = value;
  if(weight > limit) affinity = affinity / weight;
  return affinity;
}
//巡回セールスマン問題の記述例
public double setAffinity(int[][] cName, int nCell) {
  double affinity = 0.0d;
  double len = 0;
```

```
    for(int g = 0; g < nGenes; g++) {
      double dx;
      double dy;
      int gNext = g + 1;
      if(gNext == nGenes) gNext = 0;
      dx = city[cName[nCell][g]][0]
                                - city[cName[nCell][gNext]][0];
      dy = city[cName[nCell][g]][1]
                                - city[cName[nCell][gNext]][1];
      len += Math.sqrt(Math.pow(dx, 2) + Math.pow(dy, 2));
    }
    affinity = 1.0d / len;
    return affinity;
  }
```

類似度メソッドの定義

類似度を計算する処理は遺伝子をコーディングする手法に依存し，対象とする組合せ最適化問題によって処理が異なるため，ユーザが問題に応じて定義する必要がある．例えば，KP の場合は同じ荷物の有無をもとに，TSP の場合は都市間の繋がりをもとに類似度を算出することになる．

類似度メソッドは，特定の 2 つの細胞に対して 1 世代当たり数回呼び出されるため，計算対象となる 2 つの細胞情報が必要となる．そのため，類似度メソッドの引数に 2 組の細胞名と細胞番号を設定する必要があり，Gene 言語とは異なり,「抗体を格納する変数」が省略できる．細胞名には抗体，記憶細胞またはサプレッサー T 細胞のうちいずれかの細胞を指定し，細胞番号には各細胞に割り振られている細胞の番号を指定する．ここで，細胞を格納する変数は，何番目の細胞の遺伝子座がいくつの遺伝子なのかを指定できるようにするために，配列の第 1 添え字を細胞番号，第 2 添え字を遺伝子座とした二次元配列で定義する．バイナリ型 IA の場合は boolean 型の二次元配列，整数型 IA の場合は int 型の二次元配列となる．

1 つの細胞の名前を cName1，番号を nCell1，もう 1 つの細胞の名前を cName2，番号を nCell2，類似度の定義メソッド名を setSimilarity とし，double 型の類似度を戻り値として返す記述例を以下に示す．

```
//ナップザック問題の記述例
public double setSimilarity(boolean[][] cName1, int nCell1,
                            boolean[][] cName2, int nCell2) {
  double similarity = 0.0d;
  int commonLug = 0;
  for(int g = 0; g < nGenes; g++) {
```

```
      if(cName1[nCell1][g] == cName2[nCell2][g]) commonLug++;
    }
    similarity = (double) commonLug / nGenes;
    return similarity;
  }
  //巡回セールスマン問題の記述例
  public double setSimilarity(int[][] cName1, int nCell1,
                              int[][] cName2, int nCell2) {
    double similarity = 0.0d;
    int commonPath = 0;
    int commonPath_Rev = 0;
    for(int g = 0; g < nGenes; g++) {
      int gNext1 = g + 1;
      if((g + 1) == nGenes)gNext1 = 0;
      int gSame = 0;
      while(cName1[nCell1][g] != cName2[nCell2][gSame])gSame++;
      int gNext2 = gSame + 1;
      if(gNext2 == nGenes) gNext2 = 0;
      int gNext2_Rev = gSame - 1;
      if(gNext2_Rev < 0) gNext2_Rev = nGenes - 1;
      if(cName1[nCell1][gNext1] == cName2[nCell2][gNext2])
                                                commonPath++;
      if(cName1[nCell1][gNext1] == cName2[nCell2][gNext2_Rev])
                                              commonPath_Rev++;
    }
    if(commonPath < commonPath_Rev)commonPath = commonPath_Rev;
    similarity = (double) commonPath / nGenes;
    return similarity;
  }
```

immune 構文による必要情報の指定

immune 構文では，IA の型名，つまりバイナリ型 IA の場合は boolean 型，整数型 IA の場合は int 型と，これまで定義してきたメソッドや変数に関する情報を指定する．Immune 言語コンパイラは，この情報をもとに IA 処理に必要なメソッド等を補完して Java 言語プログラムを自動生成する．

「IA 処理開始メソッドの呼び出し」で述べた IA の処理を開始するために呼び出すメソッド名を start,「親和度メソッドの定義」で述べた IA の親和度の算出方法を定義したメソッド名を setAffinity,「類似度メソッドの定義」で述べた IA の類似度の算出方法を定義した

メソッド名を setSimilarity,「前提条件変数の定義」で述べた対象とする組合せ最適化問題に関する前提条件を格納する変数名を KP の場合 luggage，TSP の場合 city とした記述例を以下に示す.

```
//ナップザック問題の記述例
immune start(boolean, setAffinity, setSimilarity, luggage);

//巡回セールスマン問題問題の記述例
immune start(int, setAffinity, setSimilarity, city);
```

6.3.2　Immune 言語コンパイラと Immune 言語によるプログラム開発の流れ

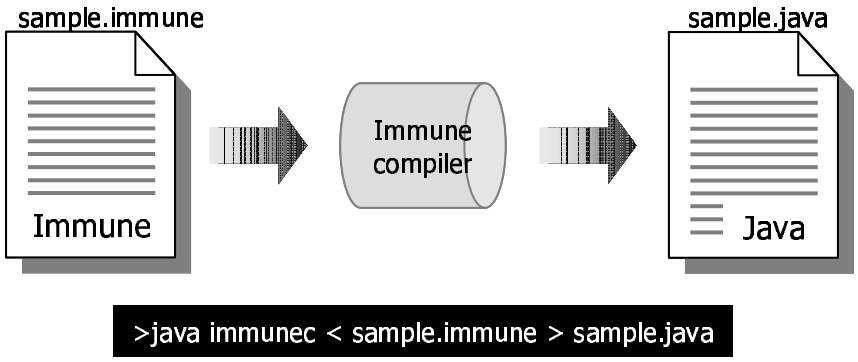
本研究では，UCLA の Computer Science Department による “The Cooperative Database Project Web Page” で公開されていた “java1_4c.jj” をベースとして，Immune 言語で記述されたプログラムを Java 言語のプログラムに変換するための Immune 言語コンパイラ “immunec” を開発した．この Immune 言語用コンパイラを用いることで，Immune 言語で記述されたプログラムを Java 言語のプログラムに変換することができる．また，Immune 言語用コンパイラは Immune 言語で必要な字句解析や構文解析だけでなく，Java 言語のすべての文法（JDK1.4 仕様）に則した構文解析も可能である.

図 6.1 に Immune 言語による開発の流れを示す．6.3.1 項で述べた Immune 言語の文法に基づいて記述されたプログラムは，Immune 言語コンパイラを通すことで構文解析が行われ，文法的にミスがなければ IA 処理に必要なメソッド等が補完された Java 言語プログラムに変換される．補完されるメソッドは，免疫システム特有の処理に関するメソッドと，GUI（Graphical User Interface）による視覚的な結果表示を行うためのメソッドである．その後，変換された Java 言語プログラムは Java 言語コンパイラを通すことで Java バイトコードに変換され，Java バイトコードが Java 仮想マシン上で実行されることになる.

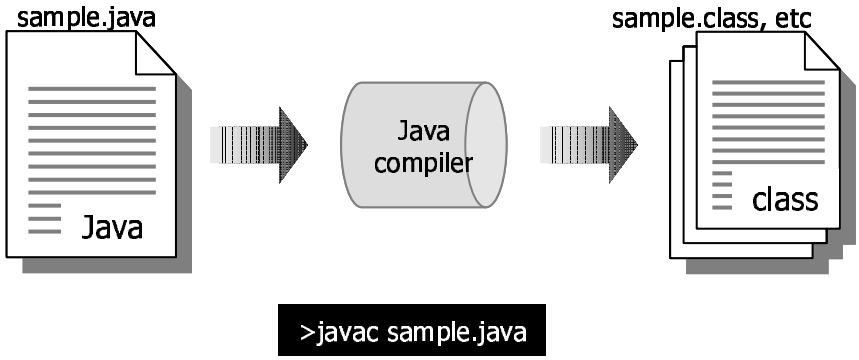
6.4　実験

本節では，Immune 言語で記述されたプログラムと，そのプログラムを Immune 言語コンパイラで変換した Java 言語で記述されたプログラムとを比較することにより，Immune 言語による開発効率を考察する．本実験では，組合せ最適化問題の代表例である KP と TSP を対象に，それぞれ各言語により記述された IA プログラムのステップ数を比較する．また，ステップ数ではコーディングする際の書式によって多少増減することが考えられるため，併せてプログラムのファイルサイズによる比較を行うものとする.

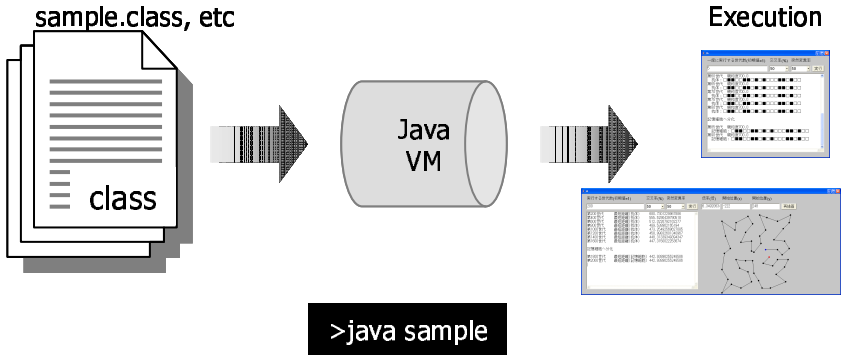
さらに，被験者 7 名を対象として，Immune 言語を用いて KP と TSP を解くプログラムを実際に作成してもらう実験を行い，その後，Java 言語の理解度や IA に関する知識，Immune 言語の利便性に関する 5 段階評価と自由記述によるアンケート調査を行った．このアンケー



(a) Immune file to Java file by Immune compiler “immunec”.



(b) Java file to class files by Java compiler “javac”.



(c) Executing class files on Java VM.

図 6.1 Immune 言語による開発の流れ

ト結果をもとに，免疫システムに関する知識の有無と Immune 言語を用いた IA の取扱いの関係について考察する．

参考までに，KP と TSP について，Immune 言語プログラムの実行結果の画面例を図 6.2，図 6.3 に，それらのもととなる Immune 言語で記述されたプログラムを付録の B.1 節および B.2 節にそれぞれ示す．

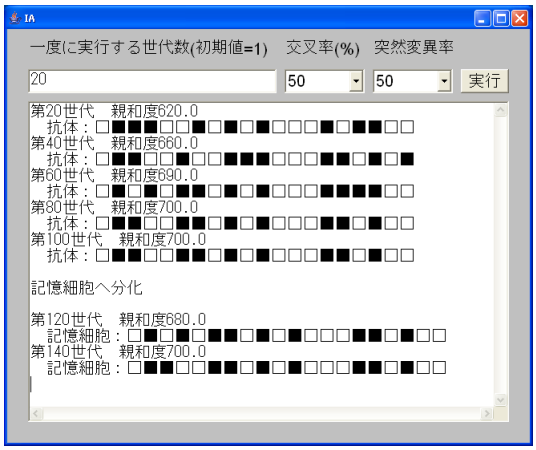


図 6.2 KP を対象とした Immune 言語プログラムの実行画面例

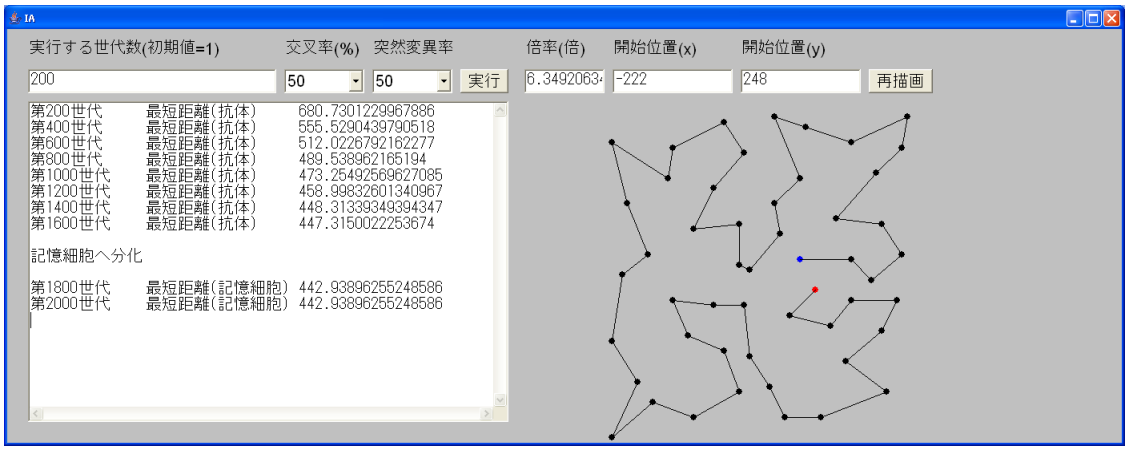


図 6.3 TSP を対象とした Immune 言語プログラムの実行画面例

6.4.1　Java 言語との比較実験と考察

Immune 言語で記述されたプログラムと Java 言語で記述されたプログラムとを，ステップ数とファイルサイズで比較した結果を図 6.4 に示す.

まず，KP をバイナリ型 IA で解くためのプログラムをステップ数で比較すると，Immune 言語での記述に要した 34 ステップは，Java 言語での記述に要した 403 ステップの約 1/12 倍であることが分かる．また，プログラムのファイルサイズについて比較すると，Immune 言語での記述に要した 1.23KB は，Java 言語での記述に要した 13.4KB の約 1/11 倍であることが分かる．なお，交叉は一点交叉とし，交叉率によって作用させる抗体数を決定する．突然変異は各遺伝子をランダムに対立遺伝子で置き換える操作を採用し，交叉により産生された抗体に対して突然変異を作用させる．突然変異を作用させる確率は突然変異率によって決定する．荷物の組合せがナップザックの許容重量を超えた抗体は，荷物の価値を総重量で除した値を親和度とした.

次に，TSP を整数型 IA で解くためのプログラムをステップ数で比較すると，Immune 言語での記述に要した 45 ステップは，Java 言語での記述に要した 623 ステップの約 1/14 倍であることが分かる．また，プログラムのファイルサイズについて比較すると，Immune 言語での記述に要した 2.09KB は，Java 言語での記述に要した 22.0KB の約 1/11 倍であることが

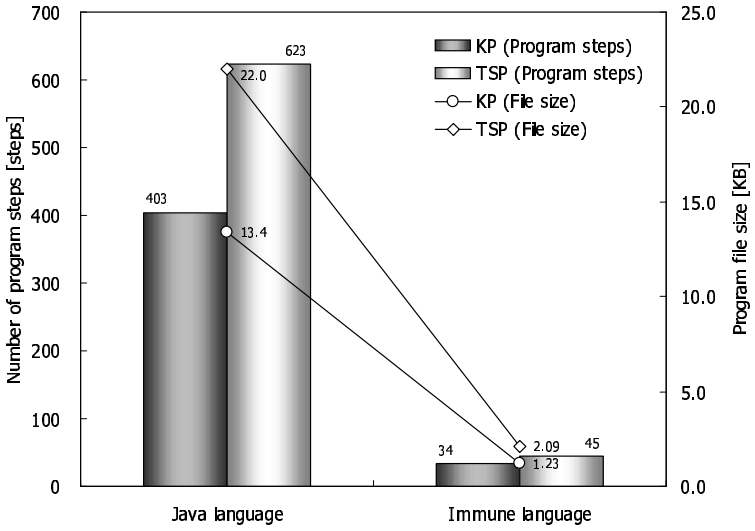


図 6.4 Java 言語と Immune 言語におけるプログラム・ステップ数とファイルサイズの比較

分かる．なお，交叉は順序交叉とし，交叉率によって作用させる抗体数を決定する．突然変異はランダムに選んだ 2 つの遺伝子を入替える操作を採用し，交叉により産生された抗体に対して突然変異を作用させる．突然変異を作用させる確率は突然変異率によって決定する．また，これら遺伝的オペレータとは別に，経験的な解の改良として，2-opt 法という局所的な改良法を Immune 言語コンパイラが自動的に組込んでいる．

以上の結果から，Immune 言語を用いることでプログラム開発者によるコーディングのステップ数，ファイルサイズ共に大幅に削減でき，IA の開発効率を向上できると考えられる．さらに，Immune 言語コンパイラにより変換されたプログラムは Java 言語で記述されているため，Java 言語の知識さえあればプログラムの変更も可能であり，容易に IA を取り扱うことができる．そのため，IA による組合せ最適化問題の解法に対するハードルを下げ，より多くの一般ユーザに対して IA 活用の機会を拡大させる効果も期待できる．

6.4.2　アンケートによる主観評価

Java 言語の基本文法の学習経験がある社会科学系の学部生 7 人を被験者として，アンケートによる主観評価を行った．本実験では，まず被験者に対して筆者らが作成したマニュアルを用いて Immune 言語および対象問題である KP と TSP について説明し，その後，Immune 言語を用いたコーディングからプログラム実行までの作業を被験者にしてもらうものとした．アンケート調査では，Java 言語の理解度や IA に関する知識，Immune 言語の利便性について，5 段階の評価を行った．そのアンケート結果を図 6.5 に示す．図中のエラーバーは 95% の信頼区間を表す．図 6.5 より，Java 言語をある程度理解していれば免疫システムに関する知識がなくとも，Immune 言語を用いることで IA を扱えることが分かる．また，自由記述を集計した結果,「思いの外，プログラムがしやすかった．詳しい内容までは理解できなくとも簡単．短いコード記述で実行ができた.」,「難しいプログラムが簡単に作れた．使いやすかった.」,「Immune 言語でソースを書くと，Java 言語と比べてコードがとても短くすんだ

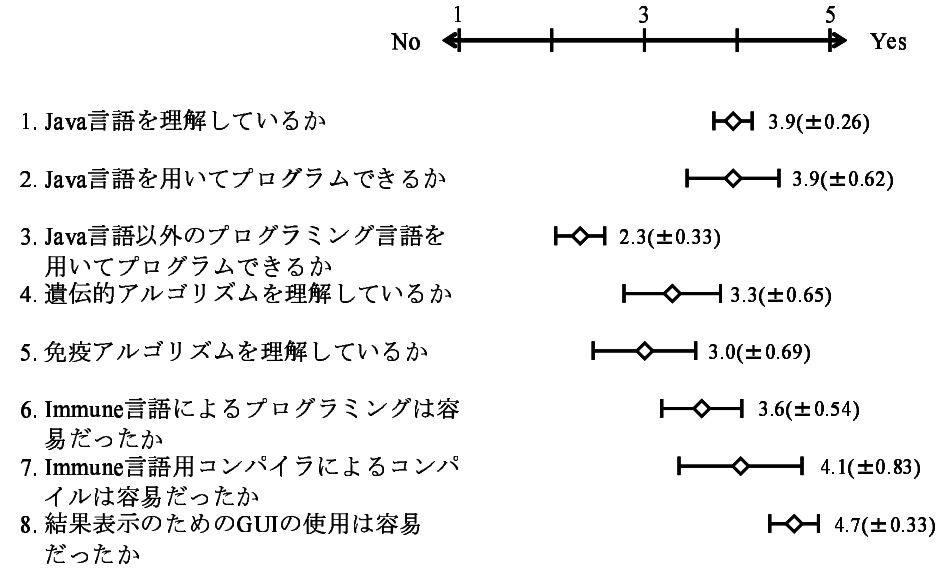


図 6.5 アンケート結果

ので驚いた.」,「慣れればすごく使いやすい言語だと思った.」などの回答が得られた．さらに,「エラーがある時，Immune コンパイラに表示されれば便利だと思った.」という回答も得られた.

以上の結果より，Immune 言語を用いることにより，免疫システムに関する知識がなくとも IA を KP や TSP に適用できると考えられる．しかしながら，現在の Immune 言語用コンパイラでは Immune 言語の文法的なミスを検出するものの，そのエラーメッセージは画面に表示されるのではなくファイルに出力されるため，エラーが発生した行番号やその内容などを，エラーメッセージとして画面に出力する機能を Immune 言語用コンパイラに追加し，ユーザの負担を軽減する必要があると考えられる．

6.5　まとめ

本章では，Java 言語に `immune` キーワードを追加定義した Immune 言語を提案した．Immune 言語を用いることでコーディングの煩雑さを軽減でき，また，Java 言語の知識さえあれば IA を容易に取り扱うことができる．IA の基本的な処理は GA と似ているため，Gene 言語同様 Espace 言語との融合により分散並列 IA を可能にする Immune 言語への展開も可能である．つまり，Immune 言語プログラムを Java 言語プログラムに変換する際，Immune 言語コンパイラが Espace 言語に関するキーワードを自動補完することで，容易に分散並列 IA を実装できると期待できる．

今回開発した Immune 言語では，Gene 言語同様，KP と TSP を解くことに焦点を絞り IA の実装を行ったため，交叉や突然変異の処理がそれぞれの問題に対して固定的に定められている．しかしながら，Immune 言語の汎用性を向上させるために，親和度計算メソッドや類似度計算メソッド同様，交叉や突然変異の方法についてもユーザ自身が記述できるよう拡張を行う必要があると考えている．また，これらの対象とする組合せ最適化問題に依存する処

理に対して，予め多種多様な問題を想定した処理を Immune 言語に組込み，ユーザが単純に選択することによる自動補完を可能とすることで，更なる開発効率の向上が見込まれる．さらに，本研究では，6.2.3 項で述べた IA 特有の多様性に関する閾値を固定値として扱ったが，その閾値を任意に決めるための機能を GUI として付加するなど，更なる機能の充実が必要であると思われる．

第 7 章

結論

本論文は，最適化問題のロバストな解を得るための一手法として複数の局所的最適解を探索し，免疫システムに着想を得たIA の研究とその応用に関する結果と，IA のためのImmune 言語に関する結果とをまとめたものである．

まず，第1章では，本研究の背景を明らかにし，その目的を明らかにした．

第2章では，まずIA の基本となる，生物の進化を模倣したGA と獲得免疫の仕組みについて述べた．次に，GA に免疫システムの一部を導入したGAIS について述べ，その後，獲得免疫の仕組みを模倣したIA について述べた．

第3章では，従来のGAIS に抑制機構と局所探索を導入した手法を提案し，対象画像内から2次元的な姿勢自由度を持つ複数の部分画像領域を探索する実験とその考察を行った．生体内に過剰に増殖した抗体を定常状態に戻すサプレッサーT 細胞の働きに着目した抑制機構により，解候補である個体の多様性を維持しながら探索を行うことで，長い世代進化を続けても大局的最適解や一つの局所的最適解に収束することなく，また適応度の優れた個体に対して局所探索を行うことで，より早い世代で複数の局所的最適解を探索することができることを示した．

第4章では，遺伝子を探索空間と同じ型でコーディングし複数種類のテンプレート画像を同時に探索するためのID を新たに組込んだIA を提案した．提案手法を書架画像中の書籍画像探索に応用した実験の結果，探索空間と遺伝子型の空間を一致させ，探索空間における抗体の分布に則した交叉手法を適用することで，バイナリ型で表現された従来のIA と比較して，より多くの局所的最適解を発見できることを示した．また，テンプレートを識別するID を遺伝子として組込むことで複数種類のテンプレート画像を同時に探索でき，テンプレート毎に独立させて探索を行うよりも効率的に解探索が可能であることを示した．

第5章では，まずTSP において有効性が示されている量子系の干渉効果を模擬した干渉交叉の螺旋構造的解釈について述べた．次に，その螺旋交叉をJSP に適用して螺旋交叉の効果がTSP に特化したものではないことを示した．

第6章では，一般ユーザであってもIA を容易に利用できる環境構築を目的としたImmune 言語を提案した．Java 言語に `immune` キーワードを追加定義したImmune 言語を用いることでコーディングの煩雑さを軽減でき，また，Java 言語の知識さえあればIA を容易に取り扱

うことができることを示した.

以上を総じて，IA の高速化と高性能化，および Immune 言語による IA の実装支援システムは，大規模化・複雑化した最適化問題における解探索精度の向上に寄与できたものと考える.

参考文献

[大内 03] 大内 東, 山本 雅人, 川村 秀憲, 柴 肇一, 高柳 俊明, 當間 愛晃, 遠藤 聡志：生命複雑系からの計算パラダイム, 森北出版 (2003)

[Cantú-Paz 98] Cantú-Paz, E.: A survey of parallel genetic algorithms, *Calculateurs Paralleles*, Vol. 10, No. 2 (1998)

[Curtis 03] Curtis, D. and Meyer, D.: Towards Quantum template matching, *Quantum-Communications and Quantum Imaging, Proc.SPIE, 5161* (2003)

[平 04] 平 英二, 高山誠悟, 内田誠一, 迫江博昭：モデル当てはめによる書棚画像解析, 電子情報通信学会論文誌 D-II, Vol. J87-D-II, No. 2, pp. 565–573 (2004)

[藤岡 02] 藤岡 弘, 中前 幸治：画像処理の基礎, 昭晃堂 (2002)

[斉藤 02] 斉藤 文彦：免疫システム型 GA を用いた正規化相関による複数画像領域探索, 電気学会論文誌 C, Vol. 122, No. 4, pp. 655–661 (2002)

[Giffer 60] Giffer, B. and Thompson, G.: Algorithms for solving production scheduling problems, *Operations Research*, Vol. 8, pp. 487–503 (1960)

[Goldberg 89] Goldberg, D.: *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley (1989)

[Han 02] Han, K.-H. and Kim, J.-H.: Quantum-inspired Evolutionary Algorithm for Class of Combinatorial Optimization, *IEEE Trans. Evolutionary Computation*, Vol. 6, No. 6, pp. 580–593 (2002)

[Han 03] Han, J.-H., K.-H.and Kim: On setting the parameters of quantum-inspired evolutionary algorithm for practical applications, *Proc. 2003 Congress on Evolutionary Computation, IEEE Press*, pp. 178–184 (2003)

[原 03] 原 崇, 飯村 伊智郎, 武田 和大, 鶴沢 偉伸, 中山 茂：インターネット・ユーザ参加型の分散並列処理のための Espace 言語の開発とその応用, 情報文化学会論文誌, Vol. 10, No. 1, pp. 11–18 (2003)

[田村 02] 田村 秀行：コンピュータ画像処理, オーム社 (2002)

[姫野 02] 姫野 雅子, 姫野 龍太郎：多峰性問題での大域的最適解と局所最適解探索のためのニッチング法, 電子情報通信学会論文誌 D-I, Vol. J85-D-I, No. 11, pp. 1015–1027 (2002)

[廣安 02] 廣安 知之, 三木 光範, 上浦 二郎：実験計画法を用いた分散遺伝的アルゴリズムのパラメータ推定, 情報処理学会論文誌: 数理モデル化と応用, Vol. 43, No. SIG-10, pp. 199–217 (2002)

[Holland 75] Holland, J.: *Adaptation in Natural and Artificial Systems*, Univ. of Michigan Press (1975)

[本田 02] 本田 勝也：フラクタル, 朝倉書店 (2002)

[Horn 94] Horn, J., Nafpliotis, N., and Goldberg, D. E.: A niched Pareto genetic algorithm for multiobjective optimization, *Proc. of the First IEEE Int. Conf. on Evolutionary Computation*, pp. 82–87 (1994)

[伊庭 99] 伊庭 斉志：進化論的計算の方法, 東京大学出版会 (1999)

[伊庭 02] 伊庭 斉志：遺伝的アルゴリズム, バイオインフォマティクス・シリーズ, 医学出版 (2002)

[飯村 05] 飯村 伊智郎, 杜 暁冬, 中山 茂：免疫アルゴリズムによる複数画像領域探索の検討, 情報処理学会論文誌, Vol. 46, No. 6, pp. 1512–1515 (2005)

[井川 05] 井川 数志, 大橋弘忠：Job-shop Scheduling Problem への Artificial Immune System の適用, 電子情報通信学会技術研究報告, Vol. 104, No. 548, pp. 31–33 (2005)

[飯村 03] 飯村 伊智郎, 池端 伸哉, 中山 茂：オブジェクト共有空間を用いた並列遺伝的アルゴリズムにおけるノアの箱舟戦略の検討, 情報知識学会誌, Vol. 13, No. 2, pp. 1–17 (2003)

[飯村 05] 飯村 伊智郎, 松岡 賢一郎, 中山 茂：1 次元トーラス網状離島モデルに基づく遺伝的局所探索における島間距離戦略の検討, 電気学会論文誌 C, Vol. 125, No. 1, pp. 84–92 (2005)

[小野 99] 小野 功, 佐藤 浩, 小林 重信：単峰性正規分布交叉 UNDX を用いた実数値 GA による関数最適化, 人工知能学会誌, Vol. 14, No. 6, pp. 1146–1155 (1999)

[石黒 97] 石黒 章夫, 近藤 敏之, 渡邊 裕司, 内川 嘉樹：免疫情報処理機構に基づく自律移動ロボットの動的行動調停システムの一強化学習法, 電気学会論文誌 C, Vol. 117-C, No. 1, pp. 42–49 (1997)

[岩川 06] 岩川 建彦, 小野 智司, 中山 茂：分散並列処理プログラミング言語 Espace の開発, システム制御情報学会論文誌, Vol. 19, No. 7, pp. 296–298 (2006)

[Kak 95] Kak, S.: Quantum Neural Computing, *Advances in Imaging and Electron Phisics*, Vol. 94, pp. 259–313 (1995)

[三井 04] 三井 和男, 大崎 純, 大森 博司, 田川 浩, 本間 俊雄：発見的最適化手法による構造のフォルムとシステム, コロナ社 (2004)

[森 93] 森 一之, 築山 誠, 福田 豊生：多様性をもつ免疫的アルゴリズムの提案と負荷割り当て問題への応用, 電気学会論文誌 C, Vol. 113, No. 10, pp. 872–878 (1993)

[森 97] 森 一之, 築山 誠, 福田 豊生：免疫アルゴリズムによる多峰性関数最適化, 電気学会論文誌 C, Vol. 117, No. 5, pp. 593–598 (1997)

[Kirkpatrick 83] Kirkpatrick, S., D., G. J. C., and , M. P., Vecchi: Optimization by simulated annealing, *Science*, Vol. 220, No. 4598, pp. 671–680 (1983)

[北野 93] 北野 宏明：遺伝的アルゴリズム, 産業図書 (1993)

[北野 95] 北野 宏明：遺伝的アルゴリズム 2, 産業図書 (1995)

[松村 99] 松村 幸輝：免疫的アルゴリズムに基づく交渉エージェント, 電気学会論文誌 C, Vol. 119-C, No. 10, pp. 1164–1174 (1999)

[Koza 92] Koza, J.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press (1992)

[熊沢 98] 熊沢 逸夫：学習とニューラルネットワーク, 森北出版 (1998)

[Menneer 95] Menneer, T. and Narayanan, A.: Quantum-inspired newral networks(QUINNS), Technical report, *Department of Computer Sience, University of Exeter, Exeter EX4 4PT, UK., Research Report329* (1995)

[澤木 00] 澤木 美奈子, 村瀬 洋, 萩田 紀博：劣化推定に基づいた辞書の自動選択による本棚画像中の文字認識, 映像情報メディア学会誌, Vol. 54, No. 6, pp. 881–886 (2000)

[森山 07] 森山 賀文, 飯村 伊智郎, 中山 茂：アントコロニー最適化法のための Ant 言語の開発, 情報文化学会誌, Vol. 14, No. 2, pp. 48–58 (2007)

[中島 97] 中島 泉, 髙橋 利忠, 吉開 泰信：シンプル免疫学, 南江堂 (1997)

[Narayanan 96] Narayanan, A. and Moore, M.: Quantum-inspired Genetic Algorithms, *Proc. IEEE Int. Conf. Evolutionary Camputation*, pp. 61–66 (1996)

[當間 00] 當間 愛晃, 遠藤 聡志, 山田 孝治：二種類の記憶機構を導入した適応的免疫アルゴリズムの提案と評価, 人工知能学会誌, Vol. 15, No. 6, pp. 1097–1106 (2000)

[Nielsen 00] Nielsen, M. and Chuang, I.: *Quantum Computation and Quantum Infomation*, Cambridge Univ. Press, Cambridge, New York (2000)

[小野 98] 小野 功, 小林 重信：Inter-machine JOX に基づく JSP の進化的解法, 人工知能学会誌, Vol. 13, No. 5, pp. 780–790 (1998)

[Ricks 03] Ricks, B. and Ventura, D.: Training Quantum Neural NetWork, *Neural Information Processing Systems* (2003)

[三宮 98] 三宮 信夫, 喜多 一, 玉置 久, 岩本 貴司：遺伝的アルゴリズムと最適化, 朝倉書店 (1998)

[Forrest 93] Forrest, S., Jovornik, B., Smith, R., and Perelson, A.: Using Genetic Algorithms to Explore pattern Recognition in the Immune System, *Evolutionary Computation*, Vol. 1, No. 3, pp. 191–211 (1993)

[中山 04] 中山 茂：量子論理ゲートでの平方根ゲートの考察, 日本計算工学会論文集, No. 20040021 (2004)

[中山 05] 中山 茂, 飯村 伊智郎, 伊藤 登志也：免疫アルゴリズムにおける量子干渉交叉法の検討, 電子情報通信学会論文誌 D-I, Vol. J88-D-I, No. 12, pp. 1795–1799 (2005)

[中山 06a] 中山 茂, 飯村 伊智郎, 松尾 翠, 前薗 正宜：遺伝的アルゴリズムにおける干渉交叉法の検討, 情報処理学会論文誌, Vol. 47, No. 8, pp. 2625–2635 (2006)

[中山 06b] 中山 茂, 前薗 正宜, 小野 智司：遺伝的プログラミングにおける螺旋交叉戦略, システム制御情報学会論文誌, Vol. 19, No. 6, pp. 262–264 (2006)

[中山 06c] 中山 茂, 今別府 孝洋, 小野 智司, 飯村 伊智郎：量子風進化的アルゴリズムにおける対交換戦略の検討, 電子情報通信学会論文誌 D, Vol. J89-D, No. 9, pp. 2134–2139 (2006)

[飯村 07] 飯村 伊智郎, 平見 亮, 森山 賀文, 中山 茂：ジョブショップスケジューリング問題での免疫アルゴリズムにおける螺旋交叉法の検討, システム制御情報学会論文誌, Vol. 20, No. 9, pp. 384–386 (2007)

[Tanese 89] Tanese, R.: Distributed Genetic Algorithms, *Proc. 3rd International Conference on Genetic Algorithms*, pp. 434–439 (1989)

[田島 00] 田島 浩一, 唐 政, 石塚 興彦, 淡野 公一：B 細胞の相互作用をもつ免疫的なネットワークによるパターン認識, 電子情報通信学会論文誌 D-II, Vol. J83-D-II, No. 2, pp. 795–804 (2000)

[長尾 91] 長尾 智晴, 安居院 猛, 中嶋 正之：書棚画像からの書籍の背文字領域抽出に関する研究, テレビジョン学会技術報告, Vol. 15, No. 42, pp. 27–32 (1991)

[長尾 02] 長尾 智晴：進化的画像処理, 昭晃堂 (2002)

[本間 03] 本間 俊雄, 加治 広之, 登坂 宣好：免疫アルゴリズムによるトラス構造の多目的最適化と解の多様性, 日本建築学会構造工学論文集, Vol. 49B, pp. 309–317 (2003)

[本間 05] 本間 俊雄, 加治 広之, 登坂 宣好：免疫アルゴリズムによる構造システムの最適化と解の多様性, 日本建築学会構造系論文集, No. 588, pp. 103–110 (2005)

[Ventura 98] Ventura, D.: Artificial Associative Memory Using Quantum Prosesses, *Proc. International Conference on Computational Intelligence and Neuroscience*, Vol. 2, pp. 218–221 (1998)

[森山 03] 森山 賀文, 飯村 伊智郎, 小野 智司, 中山 茂：抑制機構を有する免疫システム型 GA による画像探索法の研究, 電気関係学会九州支部連合大会論文集, 12–1A–05 (2003)

[森山 05] 森山 賀文, 飯村 伊智郎, 小野 智司, 中山 茂：抑制機構を有する免疫システム型遺伝的アルゴリズムによる画像探索法, 情報知識学会誌, Vol. 15, No. 3, pp. 48–58 (2005)

[森山 06] 森山 賀文, 飯村 伊智郎, 中山 茂：免疫アルゴリズムのための Immune 言語の開発, 情報文化学会誌, Vol. 13, No. 2, pp. 3–11 (2006)

[吉永 05] 吉永 孝二, 飯村 伊智郎, 中山 茂：遺伝的アルゴリズムのための Gene 言語の開発, 情報文化学会論文誌誌, Vol. 12, No. 1, pp. 18–25 (2005)

[石川 97] 石川 幸博, 大倉 充, 塩野 充, 橋本 禮治：書棚画像からの書籍の背文字領域の抽出について, 岡山理科大学紀要, Vol. 32A, pp. 163–171 (1997)

[廣谷 06] 廣谷 裕介, 小野 智司, 中山 茂：実数免疫アルゴリズムと準ニュートン法のハイブリッドによる複数解探索法の基礎的検討, 電子情報通信学会論文誌 D, Vol. J89-D, No. 1, pp. 121–128 (2006)

[大竹 92] 大竹 善二郎, 長尾 智晴, 安居院 猛, 中嶋 正之：書棚画像からの書籍の背文字領域の抽出, 日本印刷学会誌, Vol. 29, No. 1, pp. 38–45 (1992)

付録　A

ジョブショップスケジューリング問題を対象とした分散免疫アルゴリズムの検討

A.1　はじめに

最適化問題の近似解法として，生物の進化に着想を得た遺伝的アルゴリズム（Genetic Algorithm: GA）[Holland 75, Goldberg 89, 北野 93, 伊庭 02] が幅広い分野で適用されているが，GA は膨大な反復計算を必要とするため計算負荷が高いことが知られている．そのため，GA の分散並列化について多くの研究がなされてきた [Cantú-Paz 98].

GA の分散モデルの一つに島モデル [Tanese 89] があり，この島モデル型の GA のことを分散遺伝的アルゴリズム（Distributed GA: DGA）と呼ぶ．島モデルでは，個体の母集団（population）を複数のサブ母集団（subpopulation）に分割し，サブ母集団ごとに独立して遺伝的操作を作用させる．サブ母集団は島（island）とも呼ばれる．また，定期的に各サブ母集団内の一部の抗体を移住（migration）させることで，それぞれのサブ母集団が持つ個体情報を他のサブ母集団へと伝達させる．移住を行う間隔を移住間隔（migration interval）といい，移住により交換される個体数の割合を移住率（migration rate）という．DGA は粗粒度の並列化による計算時間の短縮が可能なだけでなく，分散化を行わない単一母集団の GA よりも効率的に解を探索できる [廣安 02]．並列化の粗粒度は移住間隔と移住率によって決まる．

一方で，脳神経系，遺伝適応系と並ぶ第三の生体システムとして，免疫系が注目されており，井川らの研究では，免疫系のメカニズムを取り入れた免疫型システム（Artificial Immune System: AIS）をジョブショップスケジューリング問題（Job-shop Scheduling Problem: JSP）に適用することで DGA と同程度の解を得られることが示されている [井川 05]．また，脊椎動物の持つ獲得免疫の仕組みを参考にした IA[森 97] は，自身の持つ抗体産生機構と自己調節機構により，大局的最適解を含む複数の局所的最適解が得られるという特徴を有する．森ら [森 97] や本間ら [本間 03]，飯村ら [飯村 05] の研究では，それぞれ多峰性関数の最適化問題やトラス構造の多目的最適化問題，複数画像領域探索問題に適用し，良好な結果が得られている．

このような背景の下，本研究では，従来の IA の探索能力向上と最適解への早期収束を目的

として，IA の島モデルによる分散化を行い，ベンチマーク問題の Fisher & Thompson の 10 仕事 10 機械のジョブショップスケジューリング問題（Job-shop Scheduling Problem: JSP）である ft10 を用いて，その効果を検討する．以降，島モデル型の IA を分散免疫アルゴリズム（Distributed IA: DIA）と呼ぶことにする．

A.2　提案する分散免疫アルゴリズム

DIA では，DGA 同様，母集団を N_{sp} 個のサブ母集団に分割し，移住以外の IA に関する処理はサブ母集団ごとに独立して実行される．

今回の実験では，第 5 章同様，JSP に対して従来から用いられている Inter-Machine JOX [小野 98] と，JSP の交叉法としてその有効性が確認されている螺旋交叉（Helical Crossover: HX）[Narayanan 96, 中山 05, 中山 06a, 飯村 07] を用いる．本章では，この螺旋交叉と区別するため，従来から用いられている交叉を総じて古典的交叉（Classical Crossover: CX）と呼ぶことにする．

以下，DIA の処理手順について概説する．

A.2.1　JSP における分散免疫アルゴリズムの処理手順

Step 1〔初期化〕

抗原を入力情報として認識する．JSP では，各機械上で処理すべき作業の順序とその処理時間が抗原となる．複数の抗体により形成される第 t 世代の母集団を $P(t)$ とする．初期世代（$t = 0$）の母集団 $P(0)$ には，遺伝子をランダムに決定した抗体群を用いる．抗体には，各機械について作業の順序と開始時刻を定めたガントチャートをコーディングする [飯村 07]．その後，N_v 個の抗体から成る母集団を N_{sp} 個のサブ母集団に分割する．このとき，サブ母集団内の抗体数を $N_v^{(sp_i)}$ とする[*1]．ただし，sp_i はサブ母集団名を表す（i=1, 2, . . . , N_{sp}）．

Step 2〔評価〕

抗原と抗体 v の親和度 Φ_v は式 (A.1) で定義する．ただし，M_v は抗体 v の総処理時間（makespan）である．

$$\Phi_v \;=\; 1/M_v. \tag{A.1}$$

Step 3〔終了判定〕

予め指定された進化の終了条件を満たしていれば Step 9〔進化終了〕へ，そうでなければ Step 4〔移住〕へ進む．

[*1] 本章では，サブ母集団 i を構成する抗体数を $N_v^{(sp_i)}$ と表すが，サブ母集団の番号を考慮する必要がない場合には sp_i の添え字を省略し，単に $N_v^{(sp)}$ と表す．

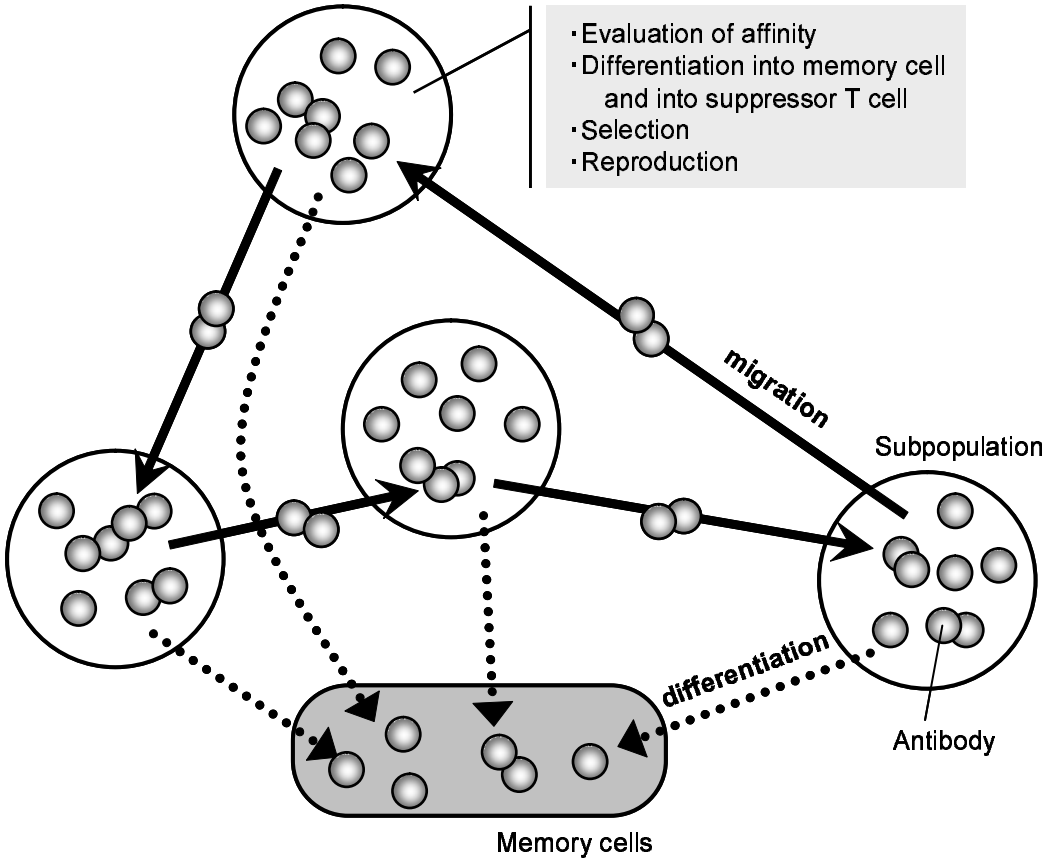


図 A.1 分散免疫アルゴリズムにおける移住

Step 4〔移住〕

移住の概念図を図 A.1 に示す．移住は I_m 世代間隔で適用する．I_m は移住間隔である．移住の方法は，まず，あるサブ母集団 i 内から移住させる $N_v^{(sp_i)} \times r_{Mg}$ 個の抗体をランダムに選ぶ．ただし，r_{Mg} は移住率である．次に，まだ移住を行っていない他のサブ母集団の中から移住先となるサブ母集団 j をランダムに決定し，選ばれたサブ母集団 i 内の抗体をサブ母集団 j へ移住させる．同様に，移住先となったサブ母集団 j 内からの抗体選出と次の移住先の決定，移住を行う．これらの処理を繰返し，すべてのサブ母集団を 1 度ずつ巡回した時点で移住処理を終了する．ただし，移住させる抗体の選出は，もともとサブ母集団内に存在する $N_v^{(sp)}$ 個の抗体のみを対象とし，新たに移住してきた抗体は対象としない．

Step 5〔濃度計算〕

まず，抗体 v と抗体 w の類似度 Ψ_{vw} を式 (A.2) で定義する．ただし，H_{vw} は抗体 v，w の各遺伝子座に共通した遺伝子の数，N_g は抗体の遺伝子数である．

$$\Psi_{vw} \quad = \quad H_{vw}/N_g. \tag{A.2}$$

次に，求めた類似度 Ψ_{vw} の総和として抗体 v の濃度 Θ_v を計算する．濃度 Θ_v は式 (2.2) で定義する．

Step 6〔分化〕

Step 5 で求めた濃度 Θ_v が閾値 T_{Π} 以上の抗体 v を記憶細胞候補 v^* とする．記憶細胞が上限数 N_m に達するまでは，候補 v^* と現時点の記憶細胞 m との最も高い類似度 Ψ_{v^*m} が閾値 $T_{\pi 2}$ 未満であるときのみ候補 v^* を記憶細胞 m に分化させる．一方，記憶細胞数が上限数 N_m に達した後は，候補 v^* との類似度 Ψ_{v^*m} が最も高い記憶細胞 m の親和度 Φ_m と候補 v^* の親和度 Φ_{v^*} を比較し，$\Phi_m < \Phi_{v^*}$ のときのみ当該記憶細胞 m を候補 v^* で更新する．ただし，DIA では，図 A.1 に示すように，サブ母集団がそれぞれ異なる記憶細胞群を保有するのではなく，最大で N_m 個の記憶細胞を全サブ母集団が共有するものとする．

次に，記憶細胞候補 v^* と同じ遺伝子を持つサプレッサー T 細胞 s を分化させ，サプレッサー T 細胞 s との類似度 Ψ_{vs} の高い（閾値 T_s 以上の）抗体 v を消滅させる．その後，消滅した抗体に代わる新しい抗体を Step 2 と同じ方法で生成し，親和度と類似度を評価する．

Step 7〔選択〕

親和度 Φ_v の低い抗体 v から $N_v^{(sp)}/2$ 個を淘汰する．なお，ここで淘汰された抗体分は，次の Step 8 で補充される．

次に，抗体 v に対して次世代に残る期待値 E_v を計算する．期待値 E_v は，式（2.4）で定義する．

Step 8〔生殖〕

本実験では，第 5 章同様，交叉法に JSP に対して従来から用いられている Inter-Machine JOX [小野 98] と，DNA の螺旋構造に由来する螺旋交叉（Helical Crossover: HX）[中山 06b, 飯村 07] を用る．以下，螺旋交叉との対比のために，古典的 IA における抗体産生のオペレータである Inter-Machine JOX を古典的交叉（CX）と呼ぶことにする．古典的交叉率 r_{CX} と螺旋交叉率 r_{HX}（$r_{CX} + r_{HX} = 1.0$）によって決まる割合で各交叉法を適用し，新たな抗体を生成する．

古典的交叉による新たな抗体の生成手法は，Step 7 で淘汰されずに残った $N_v^{(sp)}/2$ 個の抗体の中から，期待値 E_v に比例する確率分布を用いて重複を許して $(N_v^{(sp)}/2) \times r_{CX}$ 組の親抗体のペアを選び，それぞれの親抗体のペアに古典的交叉を適用することで $(N_v^{(sp)}/2) \times r_{CX}$ 個の子抗体を産生する．つまり，1 回の交叉で生成された 2 個の子抗体のうちどちらか一方をランダムに採用することで 1 組の親抗体のペアから 1 つの子抗体を生成し，交叉を $(N_v^{(sp)}/2) \times r_{CX}$ 回繰返すことで $(N_v^{(sp)}/2) \times r_{CX}$ 個の子抗体を産生する．

螺旋交叉による新たな抗体の生成手法は，Step 7 で淘汰されずに残った $N_v^{(sp)}/2$ 個の抗体の中から，重複を許さずランダムに $(N_v^{(sp)}/2) \times r_{HX}$ 個の親抗体を選び，親抗体の遺伝子を並べ，螺旋交叉を適用することで $(N_v^{(sp)}/2) \times r_{HX}$ 個の子抗体を産生する．つまり，一度の螺旋交叉で，交叉が適用される親抗体と同数の子抗体が産生される．

表 A.1 実験で用いたパラメータ値

Parameters	Values used
Number of all antibodies N_v	800
Maximum number of memory cells N_m	10
Maximum number of generations g_{max}	10,000
CX ratio r_{CX}	0.5
Number of antibodies for CX n_{CX}	$\frac{N_v^{(sp)}}{2} \times r_{CX}$
HX ratio r_{HX}	0.5
Number of antibodies for HX n_{HX}	$\frac{N_v^{(sp)}}{2} \times r_{HX}$
Mutation rate r_M	1%
Threshold T_Π	0.7
Threshold $T_{\pi 1}$, $T_{\pi 2}$, T_s, T_m	0.8
Number of subupopulations N_{sp}	1, 2, 3, 4, 5, 6, 7, 8
Number of subupopulation's antibodies $N_v^{(sp)}$	$\frac{800}{N_v^{(sp)}}$
Migration interval I_m	20, 50, 100, 200, 400, 1,000, 2,000, 5,000, 10,000
Migration rate R_m	0.5

次に，両交叉で産生された合計 $N_v^{(sp)}/2$ 個の子抗体に突然変異のオペレータを突然変異率 r_{Mt} で作用させ，Step 7 で淘汰された分の抗体を補充する．その後，Step 2 へ戻る．

Step 9〔進化終了〕

予め指定された条件を満足したため，進化を終了する．

A.3　ジョブショップスケジューリング問題への適用実験

A.3.1　実験内容

前節で述べた DIA の効果を確認するため，数値実験を行った．JSP のベンチマーク問題である Fisher & Thompson の 10 仕事 10 機械の ft10（最適総所要時間は 930）を対象に，古典的交叉率 r_{CX} と螺旋交叉率 r_{HX} を $(r_{CX}, r_{HX}) = (0.5, 0.5)$ とした DIA を用いて，それぞれ 50 回試行したときの最適解探索率と平均総所要時間，平均探索世代数を評価した．ただし，平均総所要時間には，各試行で得られた記憶細胞群のうち最良解の総所要時間を全試行で平均した値を用い，また平均探索世代数には最適解を発見したケースのみを対象とした．移住率は予備実験により $R_m = 0.5$ と決定した．つまり，今回の実験ではサブ母集団中の半数の抗体を移住させるものとした．実験で使用したパラメータ値を表 A.1 に示す．

表 A.2 移住間隔 i_{Mg} に関する実験結果

Items evaluated	Migration interval i_{Mg}								
	20	50	100	200	400	1,000	2,000	5,000	10,000
Discovery rate of the optimal solution [%]	72	82	98	96	100	98	100	98	96
Average makespan	933.2	931.7	930.2	930.2	930.0	930.1	930.0	930.1	930.2
Average number of generations for search	770.2	910.9	1133.0	1133.0	1682.2	1559.8	1611.6	1915.1	2227.6

表 A.3 サブ母集団数 N_{sp} に関する実験結果

Items evaluated	Number of subpopulations N_{sp}							
	1[*2]	2	3	4	5	6	7	8
Discovery rate of the optimal solution [%]	90	80	80	98	98	96	86	54
Average makespan	931.0	932.1	931.9	930.2	930.2	930.2	930.9	933.1
Average number of generations for search	2207.0	669.8	724.1	1133.0	1623.2	2628.3	3698.7	4172.3

A.3.2　移住間隔に関する実験結果および考察

まず，サブ母集団数を $N_{sp} = 4$ とし，移住間隔を $I_m = \{20, 50, 100, 200, 400, 1{,}000, 2{,}000, 5{,}000, 10{,}000\}$ と変化させた実験を行った．結果を表 A.2 に示す．$I_m \geq 100$ のとき最適解探索率がほぼ 100% となることが分かる．しかし，移住を行わず単に各サブ母集団を同時並行に IA を実行した場合，つまり移住間隔と終了条件が等しい場合（$I_m = 10,000$）も 96% と高い最適解探索率が得られた．これは，移住を行わず，単に全抗体をサブ母集団数で均等に分割した IA を同時並行に実行したものとほぼ同一であり，島モデルによる解探索が効果的に働いているとはいえない．そこで，最適解の平均探索世代数を比較すると，最適解を発見した場合のみを対象とし算出したものであるが，移住間隔 i_{Mg} が狭いほど，より早い世代数で最適解が得られることが分かる．また，最適解探索率が 96% 以上の $i_{Mg} = \{100, 200, 400, 1{,}000, 2{,}000, 5{,}000, 10{,}000\}$ のときのみを対象とすると，移住を行わないものと同様の $i_{Mg} = 10,000$ とした DIA は，最短平均探索世代数が得られた $i_{Mg} = 100$ のときと比較すると，ほぼ同等の探索性能を得るために約 2 倍もの世代数を必要とすることが分かる．このとき，$I_m = 10,000$ とした移住を行わない DIA と，単一母集団の IA との相違点は記憶細胞の実装方法にある．表 A.2 より，DIA は，移住間隔を狭めるほどより優れた抗体情報が他のサブ母集団へと伝わる確率が高くなるため短い世代数で最適解を得ることができる．一方で，移住間隔が狭すぎると，各サブ母集団の過剰な多様化により，最適解への収束の妨げとなると考える．

[*2]Simple IA と同様.

A.3.3　サブ母集団数に関する実験結および考察

次に，先の実験より移住間隔を $I_m = 100$ とし，サブ母集団数を $N_{sp} = \{1, 2, 3, 4, 5, 6, 7, 8\}$ と変化させた実験を行った．結果を表 A.3 に示す．この結果より，単一母集団（$N_{sp} = 1$）の IA，つまり文献 [飯村 07] の母集団サイズ N_v を 800 個に増やし，古典的交叉率 r_{CX} と螺旋交叉率 r_{HX} を $(r_{CX}, r_{HX}) = (0.5, 0.5)$ へと変更した IA は 90% の最適解発見率を得ていることが分かる．一方，母集団を複数の母集団に分割した DIA は，サブ母集団数が 4，5 のとき 98% ，サブ母集団数が 6 のとき 96% と単一母集団の IA よりも高い最適解探索率を得ることができた．しかしながら，サブ母集団数 N_{sp} を過剰に増加させると解探索能力が低下することが分かる．これは，各サブ母集団の抗体数 N_v^{sp} 減少により，局所的な探索が十分行われないまま濃度の高まった抗体が記憶細胞へ分化することが原因であると考えられる．また，DIA（$N_{sp} \geq 2$）ではサブ母集団数が少ないほど最適解の平均探索世代数が少ないことが分かる．移住による抗体情報の交換は，サブ母集団数が少ないほど母集団全体へ情報が伝達し易いため，サブ母集団数が少ないほど早く優れた抗体の情報が伝達し，より短い世代数で最適解を得られたと考えられる．

ちなみに，母集団サイズ N_v を 3,000 に増やした実験を行った結果，100% の最適解探索率を得ることができた．また，そのときの平均探索世代数は約 426.6 世代であった．ただし，先の実験結果より，母集団サイズ N_v を 800 とした場合，サブ母集団数を 4 としサブ母集団の抗体数 N_v^{sp} が 200 のときより短い世代数で高い最適解探索率が得られたため，母集団サイズ N_v を 3,000 に増やした場合も，同様に，各サブ母集団の抗体数 N_v^{sp} が 200 となるよう，サブ母集団数 N_{sp} を 15 とした．

A.4　おわりに

島モデルによる分散化を IA に組込むことにより，実験で利用した ft10 に対して，最適解発見率の向上と最適解の平均探索世代数の短縮が確認できた．

今後は，提案する DIA を JSP の他の問題に適用し，より詳細に分析し探索性能の向上を図っていきたい．

付録 B

6.4節におけるImmune言語によるプログラム記述例

B.1 KPのためのImmune言語プログラム記述例

以下に，KP を解くプログラムを Immune 言語で記述した例を示す.

```
public class IA_bin {
  static int nGenes = 20;
  static int nAntibodies = 50;
  static public void main(String ar[])throws Exception{
    IA_bin ia_bin = new IA_bin();
    //IA 処理開始メソッドの呼び出し
    ia_bin.start(nGenes, nAntibodies);
  }
  //前提条件変数の定義
  int luggage[][] = {{50, 10}, {30, 40}, ... , {120, 100}, {100, 70}};
  //親和度メソッドの定義
  public double setAffinity(boolean[][] cName, int nCell) {
    double affinity = 0.0d;
    int weight = 0;
    int value = 0;
    int limit = 550;
    for(int g = 0; g < nGenes; g++) {
      if(cName[nCell][g]) {
        weight += luggage[g][0];
        value += luggage[g][1];
      }
    }
    affinity = value;
    if(weight > limit) affinity = affinity / weight;
    return affinity;
  }
  //類似度計算メソッドの定義
  public double setSimilarity(boolean[][] cName1, int nCell1,
                              boolean[][] cName2, int nCell2) {
```

```
    double similarity = 0.0d;
    int commonLug = 0;
    for(int g = 0; g < nGenes; g++) {
      if(cName1[nCell1][g] == cName2[nCell2][g]) commonLug++;
    }
    similarity = (double) commonLug / nGenes;
    return similarity;
  }
  //immune 構文による必要情報の指定
  immune start(boolean, setAffinity, setSimilarity, luggage);
}
```

B.2　TSP のための Immune 言語プログラム記述例

以下に，TSP を解くプログラムを Immune 言語で記述した例を示す.

```
public class IA_int {
  static int nGenes = 51;
  static int nAntibodies = 100;
  static public void main(String ar[])throws Exception{
    IA_int ia_int = new IA_int();
    //IA 処理開始メソッドの呼び出し
    ia_int.start(nGenes, nAntibodies);
  }
  //前提条件変数の定義
  double city[][] = {{37, 52}, {49, 49}, ... , {56, 37}, {30, 40}};
  //親和度メソッドの定義
  public double setAffinity(int[][] cName, int nCell) {
    double affinity = 0.0d;
    double len = 0;
    for(int g = 0; g < nGenes; g++) {
      double dx;
      double dy;
      int gNext = g + 1;
      if(gNext == nGenes) gNext = 0;
      dx = city[cName[nCell][g]][0] - city[cName[nCell][gNext]][0];
      dy = city[cName[nCell][g]][1] - city[cName[nCell][gNext]][1];
      len += Math.sqrt(Math.pow(dx, 2) + Math.pow(dy, 2));
    }
    affinity = 1.0d / len;
    return affinity;
  }
  //類似度計算メソッドの定義
  public double setSimilarity(int[][] cName1, int nCell1,
                              int[][] cName2, int nCell2) {
    double similarity = 0.0d;
    int commonPath = 0;
    int commonPath_Rev = 0;
    for(int g = 0; g < nGenes; g++) {
      int gNext1 = g + 1;
```

```
      if((g + 1) == nGenes) gNext1 = 0;
      int gSame = 0;
      while(cName1[nCell1][g] != cName2[nCell2][gSame]) gSame++;
      int gNext2 = gSame + 1;
      if(gNext2 == nGenes) gNext2 = 0;
      int gNext2_Rev = gSame - 1;
      if(gNext2_Rev < 0) gNext2_Rev = nGenes - 1;
      if(cName1[nCell1][gNext1] == cName2[nCell2][gNext2]) commonPath++;
      if(cName1[nCell1][gNext1] == cName2[nCell2][gNext2_Rev]) commonPath_Rev++;
    }
    if(commonPath < commonPath_Rev) commonPath = commonPath_Rev;
    similarity = (double) commonPath / nGenes;
    return similarity;
  }
  //immune 構文による必要情報の指定
  immune start(int, setAffinity, setSimilarity, city);
}
```