複雑なビルディングブロック重複を持つ問題に対する交叉手法の提案

辻 美和子[†] 棟朝雅晴[†] 赤間 清[†]

遺伝的アルゴリズムによる効率的な探索のために，同一のビルディングブロック（building block, BB）を構成する遺伝子座の集合を検出する手法は多く提案されている（Heckendorn ら）．しかしながら，これらの手法から得られたリンケージ情報を利用して効果的に交叉を行う方法については，十分な検討がなされてこなかった．特に重複する BB を持つ問題では Yu ら（2005）の交叉手法のみが知られている．しかし，彼らの手法は BB の重複構造が複雑になったとき，頻繁に BB を破壊し，かつ十分な交叉パターンが得られないために，効率的に機能しない．本論文では，Yu らの手法を拡張し，BB 破壊をできるだけ抑えながら，新たな異なる探索点を与える交叉手法を提案する．提案される手法は，コンテクスト依存交叉（Context Dependent Crossover，CDC）と呼ばれ，与えられた親個体組の値を調査したうえで，交換する遺伝子座を決定する．CDC は，リンケージ同定手法と併用されることで，重複する BB を持つ問題を探索する強力なアルゴリズムを提供する．また，提案手法の性能を確認するために，重複の複雑さが制御可能なテスト関数を設計する．

A Proposal of Crossover Method for Complex Building Blocks Overlapping

Miwako Tsuji,[†] Masaharu Munetomo[†] and Kiyoshi Akama[†]

In order to realize effective genetic algorithms, there have been several techniques to identify linkage sets of loci to form a building block (BB) (Heckendorn, et al.). By contrast, the way to realize effective crossover from the linkage information given by such techniques has not been studied enough. Especially for problems with overlapping BBs, a crossover method proposed by Yu, et al. (2005) is the first and only known research. However it cannot perform well for problems with complexly overlapping BBs due to BB disruptions and insufficient variety of crossover sites. In this paper, we propose a crossover method which examines values of given parental strings to determine which variables are exchanged to produce new and different strings without increasing BB disruptions as much as possible. Because the proposed method considers the context of parental strings, it is called context dependent crossover (CDC). Combining a scalable linkage identification technique and the CDC, an effective algorithm for problems with overlapping BBs is provided. Moreover, to test the proposed method, we design test functions with controllable complexity of overlaps.

1. はじめに

ビルディングブロック仮説[7] によれば，遺伝的アルゴリズム（Genetic Algorithm，GA）は優れた部分列であるビルディングブロック（BB）を組み合わせることで解空間を探索するアルゴリズムである．このメカニズムと現実の多くの問題は複数の部分問題へと分解可能であろうという直観の一致により，GA は注目を集めてきた[3]．しかし，このような問題において同一の部分解を構成するような遺伝子座どうしを単純な1点交叉などを用いて組合せ可能な短い部分列として符号化することは，しばしば困難であった．以降では，BB という用語を，各部分問題に対して有望な部分解を構成するような，互いに依存関係を持つ遺伝子の集合に対して用いる．このような遺伝子の集合は交叉においてまとめて扱われる必要がある．効果的な BB 交換を促進するために，同一の BB を構成するような遺伝子座の集合—リンケージ集合—を検出するさまざまな手法が提案されてきた[6),8),11),16),17)]．

対照的に，得られた依存関係の情報を利用して BB を交換する方法に関しては十分な研究がなされてこなかった．特に，BB が重複する問題では，正しく依存関係が得られたか，あるいは何らかの方法で問題構造

† 北海道大学情報基盤センター大規模計算システム研究部門
Division of Large-Scale Computational Systems, Information Initiative Center, Hokkaido University

が既知である場合さえも，BB を適切に組み合わせることは困難である．にもかかわらず，そのような問題に関しては，Yu ら[20] による手法が知られるのみだった．しかし，彼らの手法は，BB が複雑に重複するとき，頻繁に BB を破壊し，また限られた遺伝子座のみしか交換できない．本論文では，Yu らの手法を拡張し，重複が複雑な問題においてもより少ない BB 破壊でより多くの BB 組合せを提供する手法を提案する．さらに，重複の複雑さが制御可能なテスト関数を設計し，提案した手法をテストする．

次章で本論文で対象とする重複のある加法的分解可能関数について述べ，重複する BB を持つ問題を定義する．3 章において既存のリンケージ同定と交叉手法について議論し，4 章で新たな手法を提案する．5 章でテスト関数を設計，実験結果を示し，6 章で結論を述べる．

2. 加法的分解可能関数

部分列の組合せによる探索は，人間の技術革新と似た側面を持っており，前述のように，実際の多くの問題構造とも一致すると考えられている[3]．これらの観測から，GA の最適化の対象となる関数として，加法的分解可能関数と呼ばれる低次の部分関数の和からなる問題モデルが考えられてきた．本論文では，重複のある加法的分解可能関数を考える．この関数においては，各部分関数の適応度の高い解が BB と考えられる．

実際の問題においては，複数の BB が互いに重複することが考えられる．本章では，本論文における重複する部分問題を持つ問題を定義する．長さ l のストリング $\boldsymbol{s}$ を遺伝子座の列 $\boldsymbol{s} = s_1 s_2 \cdots s_i \cdots s_l$ とする．添字は遺伝子座の番号を表す．ストリング $\boldsymbol{s}$ の適応度は，

$$f(\boldsymbol{s}) = \sum_{j=1}^{m} f_j(\boldsymbol{s}_{\boldsymbol{v}_j}), \tag{1}$$

である．上式で，m は部分関数の数，f_j は j 番目の部分関数，$\boldsymbol{s}_{\boldsymbol{v}_j}$ はその部分解である．$\boldsymbol{v}_j$ は遺伝子座番号のベクトルであり，部分解 $\boldsymbol{s}_{\boldsymbol{v}_j}$ を定義する．たとえば，$\boldsymbol{v}_j = (1,3,7,5)$ ならば，$\boldsymbol{s}_{\boldsymbol{v}_j} = s_1 s_3 s_7 s_5$ である．2 つのベクトルはすべての対応する要素が等しいときのみ，同一の部分列を指定する．V_j を $\boldsymbol{v}_j$ に含まれる遺伝子座番号の集合とする．ここで，$V_j = \{1,2,3,4\}$ かつ $V_{j'} = \{4,3,2,1\}$ ならば，$V_j = V_{j'}$ であるが，$\boldsymbol{v}_j = (1,2,3,4)$ かつ $\boldsymbol{v}_{j'} = (4,3,2,1)$ ならば，$\boldsymbol{s}_{\boldsymbol{v}_j} \neq \boldsymbol{s}_{\boldsymbol{v}_{j'}}$ であり，$s_1 s_2 s_3 s_4 \neq s_4 s_3 s_2 s_1$ である．

式 (1) は，$m = 1$ かつ部分列 $\boldsymbol{s}_{\boldsymbol{v}_1}$ を $(1,2,\cdots,l)$ で定義することで，すべての関数を表現できる．

本論文では，集合 V_j をリンケージ集合，同一のリンケージ集合内部の遺伝子座どうしの相互依存関係をリンケージと呼び，部分関数の解候補をビルディングブロック（BB）と呼ぶ．2 つの異なる部分関数を定義する 2 つの遺伝子座集合に共通の要素があるとき，つまり $V_j \cap V_{j'} \neq \emptyset$（$j \neq j'$）のとき，それらの部分関数は重複しているという．

$\overline{V}_j$（$j = 1,2,\cdots$）をリンクしていると推定された遺伝子座の集合とする．リンケージ同定[6),8),11),16),17)] の目的は，$\{\overline{V}_1, \overline{V}_2, \cdots\} \approx \{V_1, V_2, \cdots\}$ なるリンケージ集合を推定することである．

一般的な交叉オペレータはすべての遺伝子座を 2 つの大きな集合に分割し，一方を交換する．以下では，この集合を交叉集合 $\mathcal{V}$ と呼ぶ．

3. 背　　景

3.1 リンケージ同定と交叉の手法

GA の探索の課程において，式 (1) で示すような部分問題が騙し性を持つ場合や「干し草の針」のような場合には，低次の部分列の組合せや局所探索などによって BB を得ることは困難である．当然，実際の問題においては，各部分問題がこのような困難性を持つとは限らず，組合せや局所探索などで新たな優れた BB が得られる場合が多くある．しかし，BB 内部の適応度地形は事前には分からないことから，本論文では各部分問題が最も探索困難で GA のオペレータによる BB の進化が不可能であるような場合を想定する．そのような問題では，十分な個体数を持つ初期化において生成された BB を，適切に保持・交換する必要がある．

BB 破壊を防ぎ，効率的に探索を行うためにさまざまな手法が研究されてきた．遺伝子の値の摂動による適応度の変化量を利用して，陽にリンケージと呼ばれる互いに依存する遺伝子座を検出する手法や[6),8),11),16),17)]，適応度の高いストリングの分布を推定して確率モデルを構築する分布推定アルゴリズム（EDA）[9),14)]，徐々にリンケージを構築し交叉を行う手法[4),15)] などである．本研究では，陽にリンケージが得られた場合に関して，得られたリンケージ情報を利用して重複する BB を効果的に組み合わせる手法を提案した．提案した手法はリンケージ同定手法により得られたリンケージ情報だけでなく，専門知識などによってリンケージがあらかじめ分かっている場合にも用いることができる．

1 章で述べたように，特に陽にリンケージ集合を構成する手法においては，多くの場合で部分問題が重複しないことを仮定し，実際の BB 交換は BB ごとの一

様交叉[10),21)] を用いて実行されてきた．あるいは，比較的弱い依存関係を除去[12)]，比較的強い依存関係を探索[19)] するなどして，検出された依存関係情報から重複のないリンケージ集合を構成する手法がとられてきた．

しかしながら，GA の現実の問題の応用においては，排他的な集合のみで問題構造を表現することは，しばしば不可能であると考えられる．そのような問題においては，仮に問題構造が既知であったとしても，適切な BB 交換を行うことは困難である．例として**図 1** のように BB どうしが重複する問題を考える．遺伝子座 4 と 5 が BB_1 と BB_2 の両方に属している．仮に，BB_1 のみを交換すれば，BB_2 は破壊され，BB_2 のみを交換すれば，BB_1，BB_3 は破壊される．しかし，BB 交換を行わなければ，新たな探索点を得ることができない．このような問題でどのように交叉を行うべきかは，ごく少数の研究しかなされてこなかった．

3.2 既存手法とその問題点

重複する BB の交換に関するほぼ唯一の研究として，Yu ら[20)] のグラフを用いる交叉手法が知られている．彼らは，不正確なリンケージと収束の関係を理論的に調査し，

- 関連する変数を検出できなかった
- 関連しない変数をリンケージとした

場合に関して，前者のほうが GA の性能に大きく影響を与えるという結論から，グラフを用いてできるだけ少ない BB 破壊の交叉を探索する**図 2** に示すアルゴリズムを提案した．まず，何らかの手法でリンケージ集合を得る．次に，そのノードがリンケージ集合を，枝が両端の 2 ノードの間の重複関係を示すグラフを構築する．**図 3** は図 1 で示された問題に対するグラフを示す．続いて，グラフから 2 ノードを無作為に選択し，これらの 2 ノードが別々の部分グラフに属するように，グラフを 2 つの部分グラフに分割する．このとき，部分グラフの間の枝の数（枝のカット数）は最小でなければならない．

重複構造が単純なとき，この交叉手法は，少ない BB 破壊でさまざまな交叉のパターンを与えることができる．例として**図 4** にあるような環状の重複構造を考える．この関数においては，各部分関数は表現型レベルで隣接する部分関数のみと重複する．図 4 は，仮に二重丸の 2 ノードが選択されたとき，これらを最小のカット数で別々の部分グラフに分かつようなグラフ分割を示す．図において，白もしくは黒のノードのみが交換される．BB の重複が単純なとき，図で示したように同一の (n_1, n_2) 組に対してさえ，多くの交叉集合が存在する．加えて，(n_1, n_2) は交叉ごとに異なるため，交叉手法は多くの組合せを試すことができる．

しかし，枝のカット数は BB 破壊の数とつねに等しくなるとは限らないことから，最小カットから示唆さ

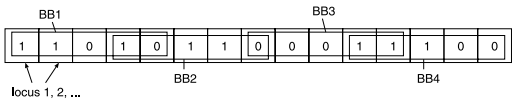


図 1 重複する BB の例
Fig. 1 An example of overlapping BBs.

(1) ノードがリンケージ集合を，枝が両端のノードの重複を示すグラフ $G = (N, E)$ を構築する．
(2) 交叉オペレータごとに：2 つのノード n_1，n_2 を無作為に選択する．グラフ G を 2 つの部分グラフ $G_1 = (N_1, E_1), G_2 = (N_2, E_2)$ に分割する．ただし，$n_1 \in N_1$, $n_2 \in N_2$ かつ $|E| - |E_1| - |E_2|$ が最小となるようにする．
(3) 交叉集合 $\mathcal{V} = \bigcup_{\overline{V}_j \in N_1} \overline{V}_j$ とし，$\mathcal{V}$ に存在する遺伝子座を交換する．

図 2 既存の交叉手法のアルゴリズム[20)]
Fig. 2 The algorithm of the existing crossover method [20)].

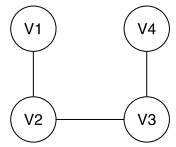


図 3 図 1 における例の既存手法でのグラフ
Fig. 3 A graph in the existing method. This represents the string in Fig. 1.

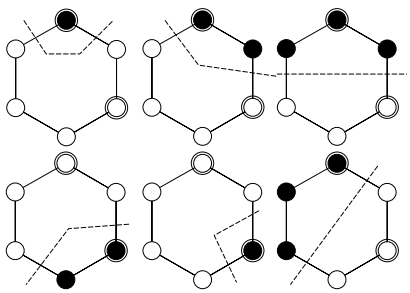


図 4 既存手法により得られる交叉の例．等価な交叉を与えるグラフ分割は除く
Fig. 4 Result from the existing crossover method. The divisions which result in an equivalent crossover are removed.

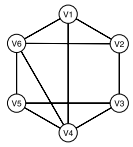


図 5 複雑な重複の例
Fig. 5 An example of complex overlaps.

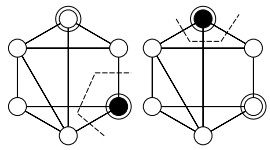


図 6 図 5 に対する V_1，V_3 に対する交叉
Fig. 6 Crossovers for the problem in Fig. 5.

れる組合せが，つねに最小の BB 破壊の交叉を実現するわけではない．

また，BB がより複雑に重複するとき，カット数を最小とするようなグラフ分割パターンは制約される．例として，**図 5** のような比較的複雑な重複構造を持つ問題を考える．図 4 と同じ 2 ノードに対するグラフは，**図 6** である．実際の問題においては，ビルディングブロックが図 4 のように規則的に重複することはほとんどなく，枝の集中により既存の手法ではほとんど BB 交換がなされることのない領域や，枝が疎であり頻繁に交叉点となる領域が出現することが考えられる．

次章では，Yu らの手法を拡張し，より少ない BB 破壊でよりさまざまな交叉パターンを試みることができる交叉手法を提案する．

4. 複雑に重複する BB を交換するための交叉手法

4.1 親個体のコンテクストに依存した交叉

BB 破壊を最小に保ちながら，交叉パターンの多様性を維持するための手法として，**図 7** に示す交叉手法を提案する．既存の手法では，すべての世代のすべての親個体組に対してつねに単一のグラフ G から最良の交叉を探索していた．提案する手法では，各親個体組に対してグラフを再構築する．交叉のためのグラフ分割を決定する前に，与えられた親個体組において同一の BB を示すノードはグラフから除かれる．また，与えられた親個体組において BB 破壊を起こす可能性のない重複と対応する枝もグラフから除かれる．このように提案手法は親個体の値（コンテクスト）に依存して最良の BB 交換を探索するため，以降ではこれをコンテクスト依存交叉（Context Dependent Crossover，CDC）と呼ぶ．

(1) ノードがリンケージ集合 $\overline{V}_j$ を，枝が両端のノードの重複を示すグラフ $G = (N, E)$ を構築する．
(2) 交叉オペレータごとに親個体組 $\boldsymbol{s} = s_1 s_2 \cdots s_i \cdots s_l$, $\boldsymbol{t} = t_1 t_2 \cdots t_i \cdots t_l$ に対して
  (a) 同一の BB を持つノードを除く．すなわち，$\boldsymbol{s}_{\overline{\boldsymbol{v}}_j} = \boldsymbol{t}_{\overline{\boldsymbol{v}}_j}$ なるノード $\overline{V}_j$ を除く．
  (b) 両端のノードが別々の子個体に引き継がれても BB が破壊されない枝を除く．すなわち，以下の条件が満足されるとき，ノード $\overline{V}_j$，$\overline{V}_{j'}$ 間の枝を除く．
(3) 2 つのノード n_1，n_2 を無作為に選択する．グラフ G を 2 つの部分グラフ $G_1 = (N_1, E_1)$，$G_2 = (N_2, E_2)$ に分割する．ただし，$n_1 \in N_1$，$n_2 \in N_2$ かつ $|E| - |E_1| - |E_2|$ が最小となるようにする．
(4) 交叉集合 $\mathcal{V} = \bigcup_{\overline{V}_j \in N_1} \overline{V}_j$ とし，$\mathcal{V}$ に存在する遺伝子座を交換する．

図 7 CDC のアルゴリズム
Fig. 7 The algorithm of the context dependent crossover.

EDA や Salman らの手法[15] においても，親個体や個体集団の値が意味を持つ．個体集団の値の分布を調査する EDA に対して，提案手法はあくまでも 2 親個体のみの局所的なコンテクストを用いている．また，Salman らの手法[15] が交叉の結果に対して遺伝子座の同時交換の確率を更新する手法である一方，提案手法は交叉の前に親個体の値を用いて交叉の結果を左右するグラフの再構築を行っている．

図 9 は **図 8** で示した親個体組に対する CDC の例を示す．まず，対応する部分列（BB）が 2 親個体で同一のノードは除かれる（図 9 左）．このような BB は交換されようとされまいと，得られる子個体に影響を与えないからである．この操作は，親個体がまったく同一でない限り，それらと異なる子個体が得られることを保障する．図の例では，V_1 の上の BB が同一であるため，ノード V_1 は除かれる．

次に，枝の除去が検討される．ある枝に関して，その両端のノードが示す 2 つの BB が別々の子個体に受け継がれても，実際には BB 破壊が起こらない場合，その枝は除去される（図 9 中央）．

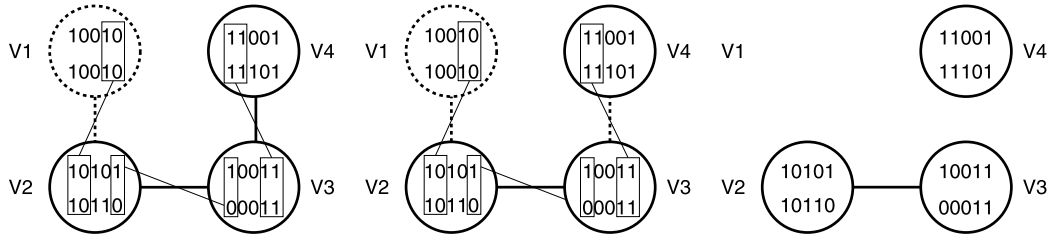


図 9 コンテクスト依存交叉の例．左から，同一の BB を示すノードを除く，BB 破壊が起こらない枝を除く，得られたグラフ
Fig. 9 Example of context dependent crossover (CDC). left: remove same BBs, middle: remove edges where BB disruption does not occur, right: resulting graph.

図 8 親個体例
Fig. 8 Parents.

図の例では，ノード V_3，V_4 間の枝が除かれる．親個体組

100101010011001 with BBs 10011 and 11001

100101100011101 with BBs 00011 and 11101,

に対して，V_3 もしくは V_4 のみを交換したときに得られる個体は

100101000011001 with BBs 00011 and 11001

100101110011101 with BBs 10011 and 11101,

もしくは

100101010011101 with BBs 10011 and 11101

100101100011001 with BBs 00011 and 11001,

であるからである．

その後，得られたグラフ（図 9 右）が枝のカット数が最小になるように 2 分割される．図 9 右のグラフは，BB 破壊なしに行える交叉があることを示唆しており，提案手法ではこの交叉が採用される．一方で，既存の手法は以下のいずれかの交換を行う：

$\{V_1, V_2, V_3|V_4\}$,
$\{V_1, V_2|V_3, V_4\}$（1 つの枝をカットする）
$\{V_1|V_2, V_3, V_4\}$（新たな個体を与えない）．

さらに，CDC は図 5 で示したような，比較的複雑な問題に対しても，さまざまな交叉パターンを与えることができる．これは，提案手法は各交叉ごとに枝とノードの除去を行ってグラフを再構築しているために，結果として親個体ごとに異なるグラフを用いて交叉を行っているからである．このように多様な交叉を用いることで，イノベーション時間—個体群にそれまで存在したよりも優れた解が発見されるまでの時間—の短縮が期待できる．

このように親個体の値に依存する交叉としては，Booker により提案された reduced surrogates crossover がある[1]．この交叉は，親個体から互いに異なる対立遺伝子のみを抜き出した個体を生成し，それらに対して交叉を適用，得られた個体をもとの個体に写像する．これにより，個体群の多様性を保持して，早期収束を防ぐことができる．reduced surrogates crossover が遺伝子座ごとの値を調査して交叉を実行したのに対して，提案手法は BB ごとの値と BB 破壊の可能性を調査して交叉を実行する．

4.2 ニッチング

ビルディングブロックが複雑に重複するとき，1 つの遺伝子座に依存する遺伝子座の数が多くなるため，すべての V_j を完全に検出することは困難である．また，重複する部分関数の最適解が矛盾する場合など，さまざまな BB 候補の組合せを試みることが必要となる．そのような多様な BB 候補を保持する手法として，ニッチングを導入する．我々の実験では，子個体をあるサイズの親個体の部分集団から最もよく似た個体と比較する手法である限定トーナメント選択（Restricted Tournament Replacement，RTR）[5] が用いられた．

5. 実　　験

5.1 テスト関数

Yu らは隣接する部分関数と 2 ビットずつ重複して環状をなすテスト関数を用いた．各部分関数 j を構成する変数は **図 10** のように，以下のように決定される：

$$\boldsymbol{v}_j = (3(j-1)+1 \bmod l, 3(j-1)+2 \bmod l, \cdots, 3(j-1)+5 \bmod l).$$

ゆえに，関数は

$$f(\boldsymbol{s}) = g(s_1 s_2 s_3 s_4 s_5) + g(s_4 s_5 s_6 s_7 s_8) + \cdots + g(s_{l-2} s_{l-1} s_l s_1 s_2). \quad (2)$$

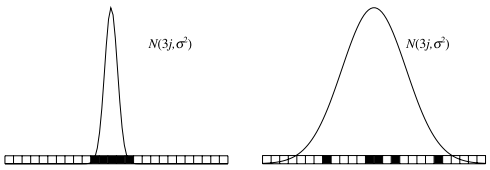


図 10 σ^2 が小さいとき（左）と大きいとき（右）で部分関数を構成する遺伝子座
Fig. 10 A sub-function with small (left) and large (right) σ^2.

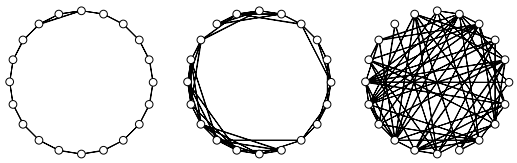


図 11 $l = 60$ の確率的環状重複関数．左から，$\sigma^2 = 1$，25，100 の場合．ノードは部分関数（BB）を，枝はその重複を示す
Fig. 11 Functions with stochastic circularly overlapping sub-functions. $l = 60$, $\sigma^2 = 1$ (left), $l = 60$, $\sigma^2 = 25$ (middle) and $l = 60$, $\sigma^2 = 100$ (right). Nodes are linkage sets and edges are overlapping relations between nodes.

となり，隣接する部分関数が互いに 2 ビットの重複を有する．

本実験では，この関数を拡張して，確率的環状重複関数を設計する．確率的環状重複関数においては，系統立てた環状から無作為なものまで，重複の複雑さを制御することが可能である．j 番目の部分関数に属する遺伝子座は次式で決定される：

$$\boldsymbol{v}_j = (N(3j,\sigma^2) \bmod l, N(3j,\sigma^2) \bmod l, \cdots, N(3j,\sigma^2) \bmod l). \quad (3)$$

ただし，$N(\mu,\sigma^2)$ は平均が μ，分散が σ^2 の正規分布である．正規分布によりサンプリングされた遺伝子座がすでに $\boldsymbol{v}_j$ に存在する場合には，新たな遺伝子座が選びなおされる．分散 σ^2 を変化させることで，問題の重複の複雑さを制御することができる．重複の長さは μ の間隔を調整することで制御可能である．**図 11** に $l = 60$ としての σ^2 の値を 1，25，100 と変化させたときの関数の重複構造を示す．

5.2 リンケージ集合が既知の場合の実験

トラップ関数

提案手法の交叉性能のみを調査するために，正しいリンケージ集合が既知であるとして実験を行った．比較対象として Yu らの既存手法，2 点交叉，1 点交叉，一様交叉を用いた．

まず，5.1 節で設計した確率的環状重複関数において各部分関数として 5 ビットトラップ関数を用いた実験を行った．この 5 ビットのトラップ関数は，Yu らの実験[20] でも用いられ，次式で定義される：

$$\mathrm{trap5}(\boldsymbol{s}_{\boldsymbol{v}_j}) = \begin{cases} \frac{4-u}{5}, & u = 0,1,2,3,4 \\ 1, & u = 5 \end{cases} \quad (4)$$

上式で，u は部分列 $\boldsymbol{s}_{\boldsymbol{v}_j}$ に存在する 1 の数を示す．上式においては，最適解が 11111 である一方で準最適解である 00000 への引き込み領域が存在し，より低次数の部分列の組合せや突然変異などの局所探索にとって困難な関数であることが知られている[2]．

本実験では，$l = 60$，90，120 の確率的環状重複関数を用い，$\sigma^2 = 1$，4，25，100 の 4 段階で変化させた．各テスト関数で，それぞれの交叉手法による 30 回の独立な試行を行い，大域的最適解に到達した割合（%）とそのときに必要とされた世代数を調査する．GA における計算機コストは，解の評価にシミュレーションや人間の評価を用いることも多い実際の問題においては，適応度評価が最も多くを占めると考えられる．本実験では同一の個体数を用いているため，各試行の世代数を比較することで，適応度評価コスト，すなわち計算機コストを比較することができる．個体群サイズは $\sigma^2 = 1$，4，25，100 のとき，それぞれ 500，600，1,200，2,000 とした．世代数が 200 を超えても最適解が発見されない場合，その実行は失敗とした．突然変異などによる新たな BB の供給が困難な場合にも適切な BB の保持交換が可能かどうかをテストするために，突然変異は用いられない．

表 1 に最適解を得られた割合とそのときに必要とされた適応度評価回数を示す．確率的環状重複関数においては，σ が小さいときには，関連する遺伝子が比較的近隣に存在する．このようなときには，1 点交叉や 2 点交叉でも解を得ることができる．特に，2 点交叉は，環状に重複する BB において，環の無作為な点から別の無作為な点までを入れ換える操作に相当するため，無作為な交叉点から特定の点（ストリングの両端）までを入れ換える 1 点交叉よりも高い性能を示している．しかし σ が大きくなると，同一の BB を構成する遺伝子座どうしが離れて存在するようになるために，これらの手法では効率的な探索が困難になる．一方で，トラップ関数は，準最適解へ引き込むような低次の組合せや局所探索に困難な騙し性を持つため，リンケージをまったく考慮しない一様交叉では解を得ることができなかった．提案手法と Yu らの既存手法では，このような BB も効率的に交換することができるため，1 点交叉や 2 点交叉と比較して優れた性能を示している．特に，提案手法では，小さい σ の値に対してさえ，Yu らの既存手法よりも高い割合で最適

表 1 提案手法，既存手法，1 点交叉，2 点交叉，一様交叉の GA においてニッチングを用いた場合に最適解に到達した割合（%）とそのときの世代数の平均．表で%は最適解に到達した割合を，#はそのときの世代数の平均

Table 1 The % of runs which obtain optimal solutions and the average number (#) of generations required to obtain the optimal solutions in GAs with proposed method, existing method, and 1point-, 2point-, uniform-crossovers.

	$\sigma^2=1$		$\sigma^2=4$		$\sigma^2=25$		$\sigma^2=100$		$\sigma^2=1$		$\sigma^2=4$		$\sigma^2=25$		$\sigma^2=100$	
l	%	#	%	#	%	#	%	#	%	#	%	#	%	#	%	#
	提案手法								既存手法							
60	100	16.2	100	16.7	100	17.9	100	21.6	100	24.4	90.0	34.3	93.3	62.1	100	53.5
90	93.3	24.6	90.0	24.3	100	25.5	96.7	45.4	86.7	37.5	46.7	55.6	36.7	117.9	46.7	149.7
120	86.6	37.9	86.7	32.2	90.0	33.3	73.3	56.0	43.3	53.6	6.67	74.0	13.3	146	0.0	nan
	2 点交叉								1 点交叉							
60	100	39.7	100	46.7	53.3	108	6.67	168	96.7	42.3	90.0	76.6	0.17	129	0	nan
90	100	58.4	90.0	76.6	23.3	126	0.0	nan	46.7	68.6	43.3	76.9	0.0	nan	0.0	nan
120	70.0	73.8	53.3	85.2	0.0	nan	0.0	nan	36.7	84.5	3.33	91.0	0.0	nan	0.0	nan
	一様交叉															
60	0.0	nan	0.0	nan	0.0	nan	0.0	nan								
90	0.0	nan	0.0	nan	0.0	nan	0.0	nan								
120	0.0	nan	0.0	nan	0.0	nan	0.0	nan								

表 2 提案手法（CDC）と既存手法でニッチングなしの場合に最適解に到達した%とそのときの世代数の平均

Table 2 Proposed method (CDC) and existing method, without niching.

	$\sigma^2=1$		$\sigma^2=4$		$\sigma^2=25$		$\sigma^2=100$		$\sigma^2=1$		$\sigma^2=4$		$\sigma^2=25$		$\sigma^2=100$	
l	%	#	%	#	%	#	%	#	%	#	%	#	%	#	%	#
	提案手法								既存手法							
60	83.3	18.56	90.0	18.41	0.0	nan	0.0	nan	40.0	26.3	3.3	33.0	0.0	nan	0.0	nan
90	56.7	27.47	43.3	26.15	0.0	nan	0.0	nan	16.7	40.2	3.3	47.0	0.0	nan	0.0	nan
120	6.7	29.50	16.7	35.00	6.7	41.5	0.0	nan	0.0	nan	0.0	nan	0.0	nan	0.0	nan

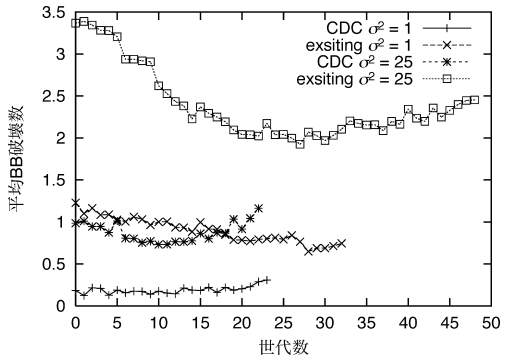


図 12 BB 破壊数．横軸は世代数，縦軸は各世代における交叉後との BB 破壊の数の平均を示す

Fig. 12 The number of BB disruptions.

解を発見しており，その差は σ が大きくなるにつれて大きくなっている．

図 12 は，$l=90$，$\sigma^2=1$ および $l=90$，$\sigma^2=25$ の確率的環状重複関数に関して，最適解が検出されるまでの各世代で実際に破壊された BB の数の平均を示す．既存手法は提案手法と比較して多くの BB を破壊していることが分かる．

また，提案手法と既存手法におけるニッチングの効果を調査するために，さきほどと同様のテスト関数・条件でニッチングのみを除いた場合の実験を行った．ニッチングを用いた置き換えに代えて，親個体と子個体をすべて適応度でソートし上位個体のみを次世代に残す置き換えを行った．この結果を **表 2** に示す．表から，比較すると，提案手法のほうが高い割合で最適解を得ることが可能であったものの，どちらの手法においても，ニッチングを用いない場合にはニッチングを用いる場合と比較して，性能の低下が見られた．特に，$\sigma^2=100$ の場合では，ニッチングを用いる場合と同様の個体数ではどちらの手法も最適解を得ることができなかった．これらは，重複構造が複雑になるにつれて交叉点が限定され，多様な子個体の生成が困難になることから，十分な探索が行われる前に個体群が早期収束したためだと考えられる．特に既存手法においては，比較的重複の複雑さの小さい $\sigma^2=4$ の場合でも，低い割合でしか最適解に到達していない．

各部分関数で最適解が矛盾する場合

重複する部分関数において部分最適解はつねに同じではない．そのような場合の探索性能をテストするために，重複する部分関数で隣接する部分関数どうしの最適解が矛盾する場合に関する実験を行った．式 (4) で示したトラップ関数，およびそれを反転させたトラッ

表 3 最適解が矛盾する場合における実験結果
Table 3 Results for functions with inconsistent optimal-sub-solutions.

	提案手法		既存手法	
l	%	#	%	#
60	100	19.4	100	24.3
90	100	30.3	93.3	38.5
120	96.7	41.3	96.7	56.6

プ関数

$$\overline{\mathrm{trap5}}(\boldsymbol{s}_{\boldsymbol{v}_j}) = \begin{cases} \frac{u-1}{5}, & u = 1,2,3,4,5 \\ 1, & u = 0 \end{cases} \quad (5)$$

が交互に連なる関数を用いた．従来の trap5() の最適解が 11111 である一方，$\overline{\mathrm{trap5}}$() の最適解が 00000 あり，これらの部分関数が交互に配される．このような場合で，最適解が既知なのは規則的な重複構造をとるときのみであるので，本実験では規則的な環状重複関数を考えた．個体数はすべての場合で 500 とし，さきほどの実験と同様にして，ニッチングを用いて，各 30 回の試行を行い最適解が得られた割合とそのときの世代数の平均について調査した．

結果を **表 3** に示す．本実験においては，最適解は 111..111 もしくは 000..000 である．これは，連続する部分関数において，最適解と準最適解を交互にとることが大域的最適解となることを示している．よって，最適解が 1 つのみで，しかも反対方向への騙し性を持つ単純なトラップ関数の和と比較して，どちらの手法でも高い割合で最適解を得ている．また，このとき，提案手法のほうが高速に最適解を発見している．

NK-Landscape

最適解が未知の場合として，NK-Landscape 関数を用いた実験を行った．NK-Landscape 関数の各部分関数は，式 (3) における確率環状トラップ関数において部分関数を構成する変数の選択方法を，

$$\boldsymbol{v}_j = (j, N(j,\infty) \bmod l, \cdots, N(j,\infty) \bmod l)$$
$$\boldsymbol{v}_{j+1} = (j+1, N(j+1,\infty) \bmod l, \cdots, N(j+1,\infty) \bmod l)$$

のように変更することで得られる．

本実験では，$N = l = 200$，$K = 4$ の NK-Landscape 関数を無作為に 10 個生成し，各関数において 10 回の試行を行って得られた最良の解を比較した．各部分関数を構成する変数は上式で定義し，各部分関数における各部分列の適応度は $[0,1]$ の一様乱数から与え，それぞれの手法を用いてこれらの和が最大になる解を探索した．世代数の上限は 200 とし，世代数が 200 を超えたとき，探索を終了する．個体数は 2,000 とした．これまでと同様に突然変異は用いられない．

表 4 NK-Landscape 関数における提案手法と Yu らの既存手法とで得られた最良の解
Table 4 The best solutions by the proposed and existing method for NK-landscape functions.

問題	提案手法	既存手法
0	153.77463	152.12827
1	152.32563	151.77572
2	151.71316	151.28834
3	151.26535	150.07203
4	150.39535	149.27647
5	152.70227	151.07745
6	151.70438	150.83879
7	151.53759	149.50147
8	151.08335	150.02589
9	151.66996	150.36083

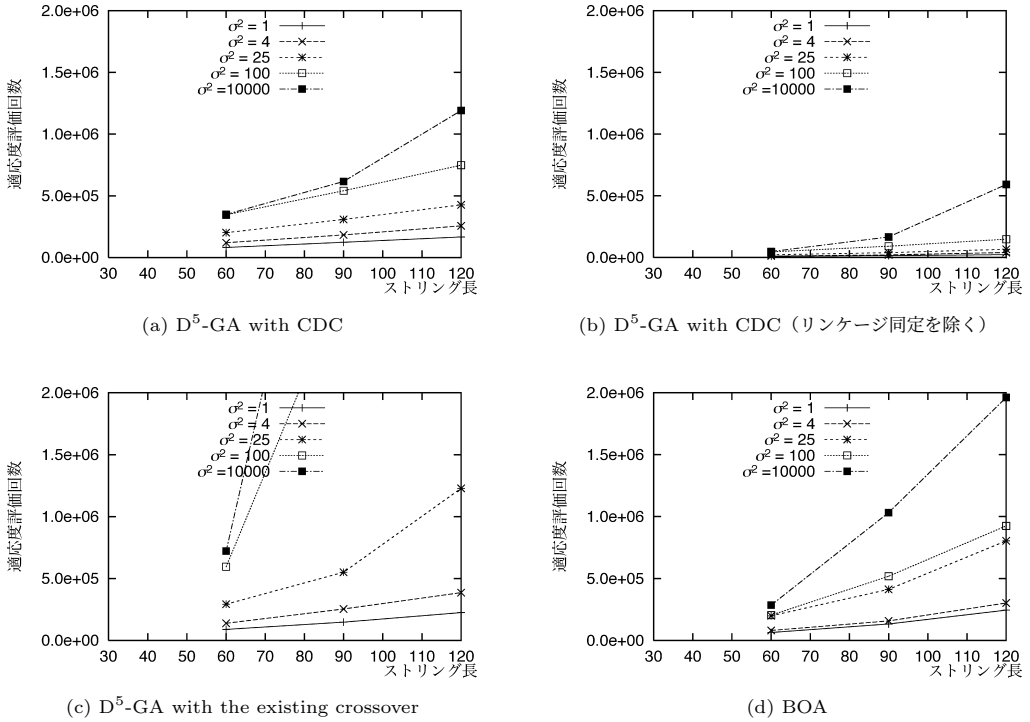
結果を **表 4** に示す．表から，すべての問題において提案手法が高い適応度を得ていることが分かる．これは，提案手法が，NK-Landscape のようにまったく無作為に重複する部分関数に対しても，多くの交叉パターンを組み合わせることが可能なためであると考えられる．

5.3 リンケージ集合が未知の場合の実験

実際の問題においては，リンケージ集合をあらかじめ知ることは難しい．ここでは，D^5 [17] を用いて依存関係を調査し，その情報を用いて確率的環状重複関数を最適化する実験を行った．

D^5 は各遺伝子座に関して，同一の部分問題を構成する遺伝子座を検出する．まず，各遺伝子座に関して，すべての個体で遺伝子座の値の摂動による適応度の差分を計算し，その大きさに従って個体を部分個体群に分類する．分類した部分個体群に対して，値が偏って分布している遺伝子座を探索することで，各遺伝子座に対して依存する遺伝子座の番号リストを与える．

リンケージ同定は適応度評価コストを必要とする．30 回の独立な試行においてすべて最適解を得られるまでに必要とされる適応度評価回数を調査した．用いた関数は $\sigma^2 = 1$，4，25，100，10000，$l = 60$，90，120 である．突然変異は使用されない．比較のために，統計的な概念を用いて互いに依存する遺伝子座を検出する手法である BOA（Bayesian Optimization Algorithm）[13] を実行した．各アルゴリズムに対して，個体群サイズは最適な解を最小の計算コストで得られるように経験的に決定された．具体的には，各アルゴリズムに関して複数の個体群サイズでの実験を行い，最も小さい計算コストを与えた結果を採用した．また，置き換えにおいて両方の手法でニッチングを用い



(a) D^5-GA with CDC

(b) D^5-GA with CDC（リンケージ同定を除く）

(c) D^5-GA with the existing crossover

(d) BOA

図 13 最適解を得るまでに必要とされた適応度評価回数
Fig. 13 The numbers of evaluations required to obtain optimal solutions in all 30 runs.

ている．

結果を**図 13** (a)–(d) に示す．(a) は提案手法を用いた D^5-GA における総適応度評価回数を示し，(b) は個体の進化に用いられた適応度評価回数のみを示す．これらの 2 つを比較すると，BB 重複が複雑になるにつれて進化に用いられる評価回数は，リンケージ同定に用いられる評価回数以上に増加することが分かる．(a)，(c)，(d) より，提案手法を用いた大きな l，σ^2 の問題に対して，D^5-GA が最も少ない適応度評価回数で最適解を与え，逆に既存手法を用いた D^5-GA は非常に大きな適応度評価コストを必要としていることが分かる．既存の研究において，BB 重複がない問題に対して，D^5-GA は，特にストリング長が BB 次数に対して大きなときや各 BB の適応度への寄与が大きく異なるとき，BOA や LINC[11] よりも高速に最適解を得ることが知られている[18),21)]．にもかかわらず，D^5 をはじめとするリンケージ同定手法でリンケージが得られたとしても，それらが重複する場合には適切に BB の組合せを行う方法が存在しなかったため，本研究においてそのための手法を提案した．

ストリング長 $l = 90$，$\sigma^2 = 100$ の場合の計算時間について**表 5** に示す．すべて Intel(R) Pentium(R) D CPU 3.00 GHz，512 M メモリの PC を用いている．図 13 (a)，(d) より，このときに BOA と提案手法を用いた D^5-GA は適応度評価回数ではほぼ同等である一方で，計算時間は BOA が大きくなっていることが分かる．これは BOA における分布推定からのモデルの構築が，高い計算機コストを必要とするためである．提案手法と既存手法を用いた D^5-GA を比較すると，リンケージ同定までの処理は同じであるためにかかる計算機時間はほぼ等しい．しかし，その後の個体の進化にかかる時間には大きな差が存在する．これは，交叉ターンの少ない既存手法による交叉で最適解を得るためには，多くの世代数を必要とし，長い世代数でもビルディングブロックが失われないようにするために，多くの個体数を必要とするためである．一方で提案手法は既存手法に加えてグラフの再構築という処理が必要となるものの，最適解を得るための必要な

表 5 総計算時間，リンケージ同定時間および進化時間．単位は秒．表で EM は Existing Method を示す
Table 5 Computation times for linkage identification, evolution and total (sec). The EM in the table is for Existing Method.

	linkage	evolution	total
D^5-GA with CDC	7.155	16.239	16.239
D^5-GA with EM	7.136	487.678	480.542
BOA	-	-	1337.277

世代数や個体数が少ないために，全体としては少ない計算機時間で解を得ている．

6. おわりに

本論文では，複雑に重複する BB を持つ問題に対する効果的な交叉手法 CDC を設計した．既存の手法は，重複が複雑になったとき，BB 破壊数が増加し交叉パターンが制約されるため効率的な探索が難しかった．提案手法 CDC は，親個体組のコンテクストを調査して交叉を決定するため，多様な子個体を BB 破壊なし（あるいはより少ない BB 破壊で）与えることができる．よって，新たなそれまでよりも優れた解を発見するまでの時間を短縮し，全体として必要とされる世代数や適応度評価回数を少なくすることが可能となる．

提案した交叉手法をリンケージ同定手法の 1 つである D^5 と組み合わせて，未知の BB が複雑に重複する問題を解くことのできるアルゴリズムを提案した．また，重複の複雑さが制御可能なテスト関数を設計し，提案手法の性能を確認した．

参考文献

1) Booker, L.: Improving search in genetic algorithms, *Genetic Algorithms and Simulated Annealing*, Davis, L. (Ed.), pp.61–73, Morgan Kaufmann (1987).
2) Deb, K. and Goldberg, D.E.: Analyzing Deception in Trap Functions, *Foundations of Genetic Algorithms 2*, Whitley, L.D. (Ed.), pp.93–108, Morgan Kaufmann Publishers (1993).
3) Goldberg, D.E.: *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*, Kluwer Academic Publishers, Norwell, MA, USA (2002).
4) Harik, G.: Learning Linkage, *Foundations of Genetic Algorithms*, Bellew, R.K. and Vose, M.D. (Eds.), pp.247–262, Morgan Kaufmann Publishers (1997).
5) Harik, G.R.: Finding Multimodal Solutions Using Restricted Tournament Selection, *Proc. 6th International Conference on Genetic Algorithms*, Eshelman, L. (Ed.), pp.24–31, Morgan Kaufmann, San Francisco, CA (1995).
6) Heckendorn, R.B. and Wright, A.H.: Efficient Linkage Discovery by Limited Probing, *Evolutionary Computation*, Vol.12, No.4, pp.517–545 (2004).
7) Holland, J.H.: *Adaptation in Natural and Artificial Systems*, University of Michigan Press (1975).
8) Kargupta, H. and Park, B.-H.: Gene Expression and Fast Construction of Distributed Evolutionary Representation, *Evolutionary Computation*, Vol.9, No.1, pp.43–69 (2001).
9) Larrañaga, P. and Lozano, J.A.: *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Kluwer Academic Publishers (2001).
10) Munetomo, M. and Goldberg, D.E.: Designing a Genetic Algorithm Using the Linkage Identification by Nonlinearity Check, Technical Report IlliGAL Report, No.98014, University of Illinois at Urbana-Champaign (1998).
11) Munetomo, M. and Goldberg, D.E.: Identifying Linkage Groups by Nonlinearity/Nonmonotonicity Detection, *Proc. 1999 Genetic and Evolutionary Computation Conference*, pp.433–440, Morgan Kaufmann Publishers (1999).
12) Munetomo, M. and Goldberg, D.E.: Linkage Identification by Non-monotonicity Detection for Overlapping Functions, *Evolutionary Computation*, Vol.7, No.4 (1999).
13) Pelikan, M., Goldberg, D.E. and Cantú-Paz, E.: BOA: The Bayesian optimization algorithm, *Proc. 1999 Genetic and Evolutionary Computation Conference*, pp.525–532, Morgan Kaufmann Publishers (1999).
14) Pelikan, M., Goldberg, D.E. and Lobo, F.G.: A Survey of Optimization by Building and Using Probabilistic Models, Technical Report IlliGAL Report, No.99018, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL (1999).
15) Salman, A.A., Mehrotra, K. and Mohan, C.K.: Adaptive Linkage Crossover, *Evolutionary Computation*, Vol.8, No.3, pp.341–370 (2000).
16) Streeter, M.J.: Upper Bounds on the Time and Space Complexity of Optimizing Additively Separable Functions, *Genetic and Evolutionary Computation — GECCO2004 Part 2, Lecture Notes in Computer Science 3103*, pp.186–197, Springer-Verlag (2004).
17) Tsuji, M., Munetomo, M. and Akama, K.: Modeling Dependencies of Loci with String Classification According to Fitness Differences, *Genetic and Evolutionary Computation — GECCO2004 Part 2, Lecture Notes in Computer Science 3103*, pp.246–257, Springer-Verlag (2004).
18) Tsuji, M., Munetomo, M. and Akama, K.: Linkage Identification by Fitness Difference Clustering, *Evolutionary Computation*, Vol.14, No.4, pp.383–409 (2006).

19) Yu, T.-L., Goldberg, D.E., Yassine, A. and Chen, Y.-P.: Genetic algorithm design inspired by organizational theory: Pilot study of a dependency structure matrix driven genetic algorithm, *Proc. Artificial Neural Networks in Engineering 2003* (*ANNIE 2003*), pp.327–332 (2003).
20) Yu, T.-L., Sastry, K. and Goldberg, D.E.: Linkage learning, overlapping building blocks, and systematic strategy for scalable recombination, *Proc. 2005 conference on Genetic and evolutionary computation*, pp.1217–1224 (2005).
21) 辻美和子，棟朝雅晴，赤間 清：適応度差分により分類された個体の分布推定を用いた遺伝的アルゴリズムの構築，MPS シンポジウム論文集—複雑系科学シンポジウム 2004，pp.157–162, 情報処理学会 (2004).

(平成 18 年 10 月 6 日受付)
(平成 18 年 12 月 11 日再受付)
(平成 19 年 1 月 25 日再々受付)
(平成 19 年 2 月 15 日採録)

辻 美和子

平成 14 年北海道大学工学部情報工学科卒業．平成 16 年同大学大学院工学研究科システム情報工学専攻修了．平成 17〜19 年日本学術振興会特別研究員．平成 19 年北海道大学大学院情報科学研究科複合情報学専攻博士後期課程修了．博士（情報科学）．進化計算および並列計算機環境下における進化計算の研究に従事．

棟朝 雅晴（正会員）

平成 8 年北海道大学大学院工学研究科情報工学専攻博士後期課程修了．同年同大学院工学研究科システム情報工学専攻助手．平成 10〜11 年イリノイ大学基礎工学科遺伝的アルゴリズム研究室客員研究員．平成 11 年北海道大学情報メディア教育研究総合センター助教授．平成 15 年同大学情報基盤センター助教授（大規模計算システム研究部門）．平成 19 年同准教授．博士（工学）．進化計算，大規模計算システム，分散システムに関する研究に従事．IEEE 会員．

赤間 清（正会員）

昭和 48 年東京工業大学工学部制御工学科卒業．昭和 50 年同大学大学院修士課程修了．昭和 54 年同大学院博士課程単位取得退学．同年同大学助手．昭和 56 年北海道大学文学部講師．平成元年同大学工学部助教授．平成 11 年同大学情報メディア教育研究総合センター教授．平成 15 年同大学情報基盤センター教授（大規模計算システム研究部門）．工学博士．知識処理，等価変換に基づく問題解決，プログラム自動生成の研究に従事．人工知能学会会員．