

## Hybrid Genetic Algorithm (HGA) による グラフの2分割問題の最適化

稲吉宏明 (\*), 秋山泰 (\*\*), 古谷立美 (\*), 星野力 (\*\*\*)

(\*) 電総研, (\*\*) 京都大学, (\*\*\*) 筑波大学

本報告では、局所探索（山登り法）とGAを融合したアルゴリズムの有効性をランダムグラフの2分割問題を例題として示す。本手法は「散在する複数の鋭い峰で構成される多峰の探索空間」において有効であり、従来のGAの“blind search”のために鋭い山を登れないという欠点を補うものである。と同時に、GAのもつ大域探索能力を継承しているため、局所探索の単純な反復のみよりも効率的に最適解に到達できることが例題で示される。なお、付録として2分割問題の探索空間の隣接構造、および組合せ問題に適用可能な4種の山登り法が述べられる。

### Optimization of Graph-Bisection by HGA (Hybrid Genetic Algorithm)

Hiroaki Inayoshi(\*), Yutaka Akiyama(\*\*), Tatumi Furuya(\*), Tsutomu Hoshino(\*\*\*)

(\*)ElectroTechnical-Lab. ,(\*\*) Univ. Kyoto, (\*\*\*) Univ.Tsukuba.

The effectiveness of the hybrid between some local-search-method, e.g. hill-climb-method, and GA is demonstrated by Bisection Problem for random-graph. This hybrid method is effective for those search-spaces which have several scattered acute-peaks and makes up for the traditional-GA's incapability of climbing acute-peaks due to its "blindness" in search. This hybrid method inherits the capability of global-searching from GA, and this capability makes the hybrid method more efficient than the repeated hill-climbing method. The efficiency is shown by an example of graph-bisection-problem. In appendix, (A) the neighboring structure in the search-space of bisection-problem and (B) four hill-climbing methods, which could be applied to combinatorial problems, are represented.

## 1 はじめに

散在する複数の「鋭い」峰で構成される多峰の探索空間を考える。(ただし、ここでいう「鋭い」峰とは、ひとつの山に着目した時に、山を構成している諸地点の高さの頻度分布が、高い地点ほど希少になっているような峰を意味する。)単純な山登り法を用いてこの空間内の最高地点に到達するためには、試行錯誤的に複数の山を登ることが必要である。他方、多峰の探索空間向きとされているGA(Genetic Algorithm)でも、“blind search”である、即ち探索空間の勾配情報を全く利用しないため、上記の探索空間では「鋭い」山を登ることができない。

本報告では、両者(山登りとGA)を融合したアルゴリズム—HGA(Hybrid GA)とよぶ—により、上記の探索空間でも効率的探索が可能となることを、グラフの2分割問題を例題として示す。

## 2 グラフの2分割問題について

一般に「グラフの $k$ 分割問題」は「 $N$ 個のノードで構成されるグラフにおいて、ノードを $n(=N/k)$ 個ずつの $k$ グループに分割する際に、異なるグループ間の2ノードをつなぐ辺の重みつき和(=コスト)が最小となるような分割の方法(組合せ)を求める問題」として定義され、組み合わせ最適化問題の一種である。 $k$ 分割問題の応用例としては、「電子回路の設計等において $N$ 個の要素で構成される回路を、 $k$ 枚のボード上で実現させようというときに、どのような組合せで要素を $k$ 分割すれば、ボード間の配線を最小化できるか」といった問題等が挙げられる。

以下に報告する実験においては、200ノードが約1/10の接続確率でランダムに接続されたグラフを100ノードずつの2集合に分割する問題を対象とした。(ただし各辺には指数分布に従う重み=辺のコストを割り当てる。)  $C(200,100) = 9.1 \times 10^{58}$ 通りの2分割方法が可能な探索空間である。グラフの2分割問題は付録Aに示されるような探索空間の隣接構造を持つ。

図1に探索空間のコスト分布の一例を示す。右の曲線がランダムサンプリングした場合で、左の曲線群が数種の山登り法(付録B参照)およびK&L法(文献[KL])とよばれるヒューリスティックス(付録C参照)により得られた低コスト値の分布である。右の曲線の左の裾が、左の曲線群とはなれていることから、「山が鋭い」ことがわかる。

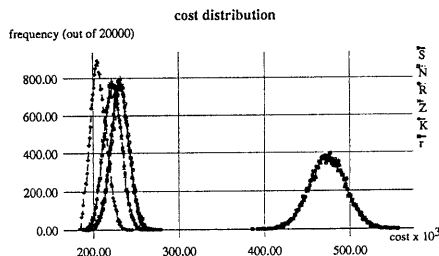


図1: ランダムサンプリング(右)および山登り法およびK&L法(左)によるコスト値の分布

## 3 GA および HGA について

### 3.1 GA について

Genetic Algorithm(遺伝的アルゴリズム; GAと略す)は、「生物集団における、遺伝、交配、突然変異、自然淘汰といった現象により、より環境に適応した個体が形成されていくプロセス」を模倣するようなアルゴリズムの一般的な枠組をさし、探索手法の一種である。GAが通常の探索手法と大きく異なっている点のひとつとして、「GAは多点探索を行なう」という点が挙げられる。即ち、通常の探索手法においては、「探索空間内における1点に着目して、その地点での勾配等のデータを基に、次の探索地点を決定するプロセスを反復する」という戦略による「一点探索」が行なわれるのに対し、GAにおいては、「探索空間内の複数箇所に探索地点をばらまき、それら複数の探索地点の評価関数値の分布を基にして次の(複数の)探索地点を決定する」という戦略が用いられる。

以下では、GA研究の源となったともいえるHollandの文献[ANAS]に示された“reproductive plan”および、本研究でその考えを採り入れた文献[Whitley]の“rank-based GA”の2つを順に説明する。

### 3.2 reproductive plan について

まず、前提として、

- (1) 探索すべき空間(例えば $n$ 次元立方体内の空間であれば、 $a_i < x_i < b_i$ ;  $i = 1 \sim n$ )および、
- (2) この空間内の要素(=探索空間内の1地点; GAでは一般に「個体」と呼ばれる):  $X_k$  に対し、その「適応度」とよばれる値を決定するある非負関数:  $\mu_k = \mu(X_k)$

の2つが与えられているとする。

このとき標準的 reproductive plan では、以下の手順で適応度  $\mu$  が最大となる個体(空間内の1地点)を探索する。

1. 探索空間内から  $M$  個体 ( $\{X_k\}; k = 1 \sim M$ ) をランダムに選ぶ。→  $M$  個体の集団(population)を構成。
2.  $\mu$  の最大値に到達または時間切れとなるまで次の3~5のステップを反復する。
3. 各個体の適応度:  $\mu_k = \mu(X_k)$  を計算する。
4. 相対適応度 ( $\equiv \mu_k / \sum_{i=1}^M \mu_i$ ) に比例した確率分布で  $M$  個体中の1個体を選ぶ。...selection process.
5. 4で選ばれた個体に遺伝的オペレータ(genetic operator)を適用し、できた個体を集団内の任意個体と置き替える。...substitution process.

「4の selection processで選ばれる」ことは、その選ばれた個体に対応する探索空間内の地点の近傍が、次の探索地点の候補となることを意味し、「5の substitution processで選ばれる」ことは、選ばれた個体に対応する探索空間内の地点の情報が、 $M$  個体の集団から消失することを意味する。

次に、前記 reproductive plan の第5ステップで現れる遺伝的オペレータの例として、(1)cross\_over (2)mutation (3) inversion の3つを説明する。ただし以下のオペレータの説明においては、簡単のため遺伝子座の

個数を  $L$  とし、各遺伝子座の採り得る値は、どの座においても 2 値 (0/1 で表現) とする。すなわち任意の遺伝型が「長さ  $L$  のビットパターン」として表されている場合を考える。2 値以外の場合への拡張は、かなり容易にできる。

1. cross\_over (交配): 前記の第 4 ステップにおいて選ばれた個体に対し、もう一度第 4 ステップを行なって、合わせて 2 個体を準備する。これら 2 個体は「両親 (parents)」とよばれる。図 2 に交差方法の 3 通りの例を示す。

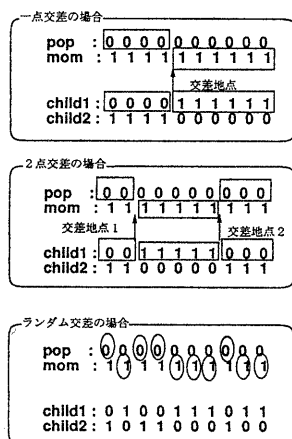


図 2: 交差方法の 3 通りの例

2. mutation (突然変異):  $L$  個のビット位置それぞれにおいて、与えられた突然変異確率でビット値を反転する。(反転するビット数は必ずしも 1 つとは限らない。)
3. inversion (反転): 交配方法として一点交差を用いる場合、2 つの遺伝子座に着目したときに、それらの距離が離れているほど交差により分割される確率が高い。そこで導入されたのがこのオペレータであり、「遺伝子座の並び順序を変更」させることにより、2 つの遺伝子座間の距離を変化させる。このオペレータの単純な実現方法として、1 ~  $L-1$  の整数を 2 地点ランダムに選び、その区間内の遺伝子座の並び順序を反転させるという方法が、文献 [ANAS] に示されている。

上記 3 つの遺伝的オペレータ中、交配および突然変異のみ用いるアルゴリズムは、SGA (standard または simple GA) とよばれている。

### 3.3 rank-based GA について

SGA (standard または simple GA) を利用する多くの GA 研究者を悩ませた問題のひとつに、「premature convergence」(未熟収束) とよばれる問題があった。これは、探索がまだ十分行われていない時期に、集団が探索空間内のある領域に集中してしまう現象のことである。この原因のひとつとして、「突出個体」(“super individual”) の存在が挙げられる。すなわち、集団

内に「他の個体よりも突出 (傑出) して適応度の高い個体」が存在すると、前記の *selection process* の説明から推察されるように、この「突出個体」近傍に次世代の探索が集中してしまうのである。

この問題の対応策として Baker により、「個体のランキング」を用いる方法が示された (文献 [Baker])。すなわち、「相対適応度に比例して次の探索候補地点を選ぶ」方法に替えて、「個体のランキングを基にして次の探索候補地点を選ぶ」方法が提案された。

その後、Whitley により、Baker の示した “premature convergence” を防ぐ方法としてのみならず、「scaling problem」の解決方法としても rank-based GA が有望であることが主張された (文献 [Whitley])。ここで、“scaling problem” とは、ちょうど “premature convergence” と反対の現象、すなわち、「どの個体も他の個体と同程度の適応度を持つ状態」において、人為的に淘汰圧を高めるためには、いかなる方法で適応度の「スケーリング変換」を行えばよいかという問題である。この問題の解決策の一例としては、文献 [Gref] に示されたような「サイズ  $W$  の “scaling window” を用いる方法」が、挙げられる。この方法では、「 $W$  世代前までの全個体中の適応度の最小値 ( $\equiv \mu_{\min}(W)$ )」を基にしたスケーリング変換が行なわれる。すなわち、現在の適応度を  $\mu$  としたとき、これを  $\mu - \mu_{\min}(W)$  で変換したのちに、相対適応度が計算される。しかし、 $W$  の値としていくつが最適か? という問題が残りの、あまり良い解決策であるとは思えない。

Whitley の主張にもあるとおり、「適応度を定める絶対的な基準」が存在するわけではなく、GA の研究者が主観的に「評価関数を基にして適応度を対応させている」のが現実である。故に、*selection process* において、この非客観的な適応度にこだわる必要はなく、ランキングで十分であるというのが彼の主張であり、筆者も賛同している。

さらに、GA に常に付きまとう問題である、“exploration or exploitation” すなわち、「さらに広範囲の探索を続けるべきか、あるいは、現地点をより深く探索すべきか」という、淘汰圧に関する問題に関しても rank-based GA は、良い指針となっている。(淘汰圧が弱ければ exploration が、強ければ exploitation が行なわれる。)

以上の理由から、本研究では rank-based GA を採用している。

rank-based GA における *selection process* は、図 3 および図 4 に示されるような「ルーレット」を基にして行なわれる。両図とも、上段が SGA、下段が、rank-based GA に対応し、簡単のため、集団サイズ ( $\equiv$  集団を構成する個体数) は 4 としている。

図 3 においては、個体 C の適応度が、他個体に比べて突出しており、4 個体の相対適応度を円グラフ、すなわち「ルーレット」としてしめせば、図中右上のルーレットとなる。すなわち、*selection process* において個体 C の選ばれる確率は、 $1/2$  以上となる。故に、この “super individual” C により、探索プロセスが支配されてしまう可能性が高い。

他方、同図下の rank-based GA においては、相対適応度でなく、「集団内での適応度の降順 (又は昇順) に並べたときの順序」を基にして、「固定形」のルーレット位置に、順に個体が配置される。ここで「固定形」のルーレットとは、SGA のルーレットにおいては、相対適応度の大きさに応じて、「ルーレットの境界が変動」するのに対して、「ルーレットの境界が変動しない」

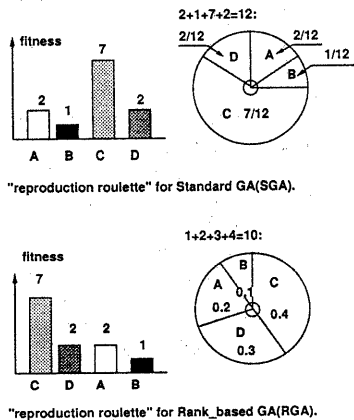


図 3: rank-based GA における「ルーレット」(1):su-per individual

ルーレットを意味している。「ルーレットの境界」は、集団サイズ  $M$  で一意的に定められ、「適応度の昇順に並べたときの順序」が  $i$  番目の個体は、

$$i / \sum_{k=1}^M k = i / ((M+1)M/2)$$

の確率で selection process において選ばれる。

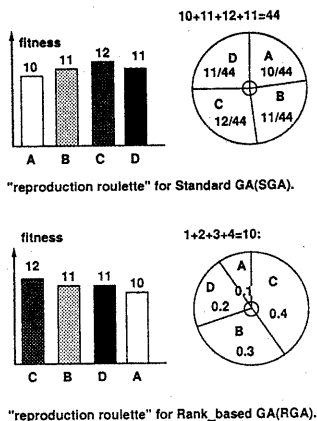


図 4: rank-based GA における「ルーレット」(2):scaling problem

図 4 においては、個体間で適応度にほとんど差がない場合、すなわち “scaling problem” が起こる場合のルーレットが示されている。同図上のルーレットでは、4 個体がほとんど等確率に選ばれるが、同図下のルーレットでは、ランクの高い個体ほど高い確率で選ばれ、淘汰圧が働くことがわかる。

### 3.4 交配手順について

交配は、適応度に比例した確率で選ばれる 2 個体の遺伝型をもとに、以下の手順で行なう。図 5 参照。

- 2 個体の遺伝型間のハミング距離が  $(N/2)$  以下となるように、必要ならば、一方のビットパターンを 0/1 を反転させる。
- ハミング距離が  $(N/2)$  以下となった 2 つのビットパターンに対し、2 個体のビット値が一致しているビット位置については、その値をそのまま offspring (子) のビット値とする。
- 2 個体のビット値が不一致のビット位置について、不一致のビット数が  $2L$  個あれば、0-1 で異なる  $L$  個のビット位置のリスト  $X$  と、1-0 で異なる  $L$  個のビット位置のリスト  $Y$  の 2 つを作成する。
- リスト  $X$  中のビット位置について、パラメータとして与えられる確率に従い、順に 0/1 のいずれかの値を決め、その値を offspring の対応するビット位置のビット値とする。
- (4) で、 $L$  個の内  $l$  個のビット位置で値 1 を採った場合、リスト  $Y$  から  $(L-l)$  個のビット位置をランダムに選びだし、offspring のそれらのビット位置は、値 1 とし、残されたビット位置については、値 0 とする。

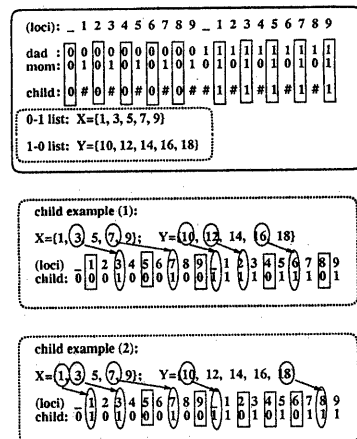


図 5: 2 分割問題用の交配手順

### 3.5 HGA について

HGA では、以下の手順で探索を行なう。(1): 解候補 (= 個体) の集団をつくる。(2): 集団内の各個体に、他個体とは独立に局所探索を行なわせる。(3):(2) で得られたピーク値の集団に遺伝的オペレータを適用する。(4):(2)&(3) を繰り返す。

### 4 実験およびその結果

2 節で設定された探索空間に対し、GA のみの場合と、HGA の場合の探索性能を本節で示す。

#### 4.1 Genetic Algorithm の探索性能

総探索地点数の上限を 10,000 地点とした RGA 法 (Rank-based\_GA) による探索の結果を図 6 に示す。同図の 3 つの曲線は 3 通りの集団サイズ (50,100,300) の場合に対応している。同図から、

1. 集団サイズが大きいくほど、より低いコスト値に到達できている点。
2. にもかかわらず、図 1 と比較して明らかなように、到達コスト値に関してはランダムウォーク程度の性能で、山登り法や K & L 法に全く及んでいない点。

の 2 点がわかる。

参考のために RGA 法 (Rank-based\_GA) による探索において、集団が収束するまでに探索された地点数の分布状況を図 7 に示す。同図の 3 つの曲線は前図同様、3 通りの集団サイズ (50,100,300) に対応している。同図から、

1. 集団サイズが大きいくほど、より多くの探索地点を収束するまで探索している点。(集団サイズ=50 のとき、平均 772 地点; 集団サイズ=100 のとき、平均 2089 地点を収束までに探索している。)
2. 集団サイズ=300 の場合、総探索地点数の上限として設定された値 (10,000 地点) の探索を行なっても、大部分の試行で未収束 (300 試行中半分以上) である点。

の 2 点がわかる。

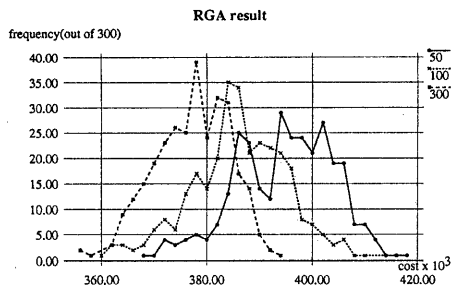


図 6: RGA 探索により到達したコスト値の頻度分布

#### 4.2 HGA の探索性能

(1) 「K & L 法を単純に 500 試行反復し、500 試行中の最小コストを求めた場合」と (2) 「500 試行 (以内) の K & L 法を HGA のもとで行なった場合」の 2 通りの探索戦略で得られたコスト値の頻度分布を図 8 に示す。同図から明らかなように、HGA の導入により、より高頻度に最小と思われる解に到達できている。図 9 には、参考として、同探索空間における K & L 法で得られた解のコスト値とハミング距離の関係を示す。(ハミング距離は、コストの最小値を与える 2 分割状態を基準としている。) なおこれらの図は、乱数の種を変化させることで生成した 8 通りの異なる探索空間中で、最も「探索が困難な」ケースである。

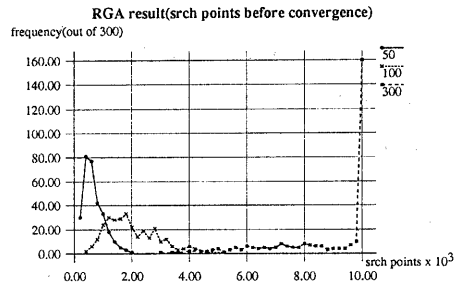


図 7: RGA 探索により探索された地点数の頻度分布

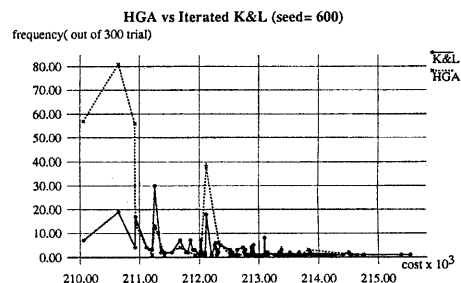


図 8: K & L 法の反復 vs. HGA の到達コスト値の分布比較

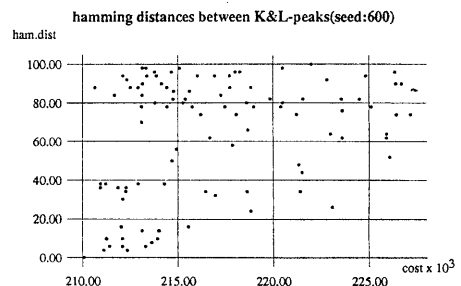


図 9: K & L 法で得られた解のコスト値とハミング距離の関係

## 5 むすび

「散在する複数の鋭い峰で構成される多峰の探索空間」においては、山登りなどの局所探索を各個体に行なわせ、大域探索である GA をこれらの個体集団に適用する、即ち、局所探索と GA を融合させることにより、従来の GA のみや・局所探索の単純な反復のみよりも効率的に最適解に到達できることを示した。

## 参考文献

- [ANAS] Holland, J.H.: "Adaptation in Neural and Artificial System" Univ. Michigan Press (1975).
- [Baker] J. Baker: "Adaptive selection methods for genetic algorithms", ICGA'85 pp.101-111.(1989).
- [Gref] Grefenstette, J.: "Optimization of Control Parameters for Genetic Algorithms", IEEE Trans. on Systems, Man, and Cybernetics, SMC-16(1) (1986) pp.122-128.
- [KL] Kernighan, B. & Lin, S.; Bell Syst. Tech. J. vol.49, pp.291-307 (Feb1970);
- [Whitley] Whitley, Darrell: "The GENITOR Algorithm and Selection Pressure: Why Rank-based Allocation of Reproductive Trials is best", ICGA'89 pp.116-121.(1989).

## 付録

### A 探索空間の隣接構造

探索空間における隣接構造の例として、順に4次元、6次元、8次元の3通りの場合の隣接構造を図10～図12に示す。

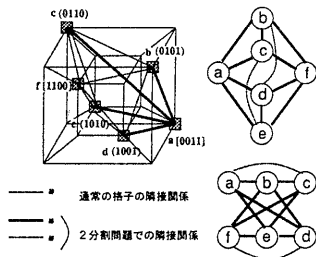


図 10: 探索空間の隣接構造 (4次元の場合)

図10は、2進4次元空間における、通常の隣接関係(=4次元0/1格子…図中細線)と、2分割問題での隣接関係(…図中太線および点線)を示し、線でつながる2地点(ノード)が上記2通りの各々の隣接関係を表している。同図右の2つのグラフは、同図左における空間地点をノードに、ノード間の隣接関係を辺に対応させ、それらのノード&辺をグラフで表現している。これらのグラフは図的表現が異なるが、実体は同一のグラフである。) 図11および図12は、図10の右図同様の「探索地点間の隣接関係のグラフ」を各々6次元および8次元の場合について示している。(ただし図中、煩雑さを避けるため、一部の隣接関係は省略されている。)

「探索地点間の隣接関係のグラフ」は、以下のように説明できる。(図12の場合参照)

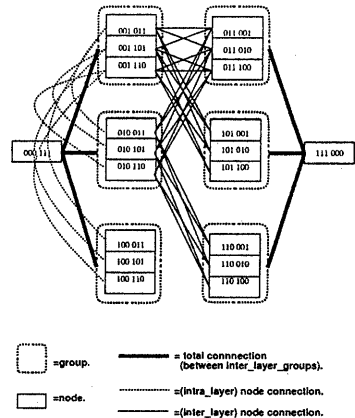


図 11: 探索空間の隣接構造 (6次元の場合)

まず、 $(2n$ 次元)の探索空間内において空間地点を2進表現したときのビットパターンが $(0,0,0,...,0, 1,1,1,...,1)$ という「連続 $n$ 個の0と連続 $n$ 個の1」のビットパターンとなるような探索空間内の一地点に着目する。

この地点を基準として(図中の左端ノード)、

- 基準と2 bit異なる地点の集合(図中の左から2列目)、
- 基準と4 bit異なる地点の集合(図中の左から3列目)、
- 基準と6 bit異なる地点の集合(図中の左から4列目)、...
- 基準と $2(n-1)$  bit異なる地点の集合(図中の右から2列目)、
- 基準と $2n$  bit異なる地点の集合(図中の右端ノード)。

という順で列挙し、隣接する地点(ノード)間を辺でつなぐと「探索地点間の隣接関係のグラフ」が形成される。ノードを上記手順で配置した時、グラフの辺は「隣合う列に属する要素の間、又は同じ列内の要素間」のいずれかのみが存在し、「隣合わない列に属する要素の間」には辺が存在しない(=対応する空間地点間は隣接関係にない)。

上記で着目した以外の一般的な地点に関しても、以下に示されるように同様の隣接関係のグラフが形成される: 着目する一般的な地点を2進表現したときのビットパターンが $(b_u, b_l)$ であるとする。 $(b_u$ および $b_l$ は、各々上位 $n$ 次元のビットパターンおよび下位 $n$ 次元のビットパターンとする。) ビットパターンの反転(=0を1に、1を0に置き換える演算)をダッシュ記号(')で示すと、 $(b_u, b_l')$ とのxor(排他的論理和)をとることにより、今着目している地点は $(0,0,0,...,0, 1,1,1,...,1)$ という「連続 $n$ 個の0と連続 $n$ 個の1」のビットパターンに変換される。 $(b_u, b_l')$ とのxor(排他的論理和)を空間内の全地点でとることにより、一般的な地点に関しても、前記同様の隣接関係のグラフが形成されることがわかる。

### B 山登り法の探索アルゴリズム

一般に山登り法とは、連続値をとるパラメータ空間における探索の場合、ある探索地点において(各パラメータに関する偏微分をとることにより)その地点まわりの勾配情報を求め、最も急勾配な方向に探索地点を移動させる、というプロセスの反復により、勾配ゼロの極大地点(最小値の探索問題では、極小地点)を見つけ出すアルゴリズムを指す。

しかしながら、2分割問題の探索空間においては、あるノードに着目したとき、ノードの採り得る状態は、「グループAに属している」または「グループBに属している」の2状態のいずれかであるため、各探索地点での勾配情報は得られない。

そこで、本探索空間のような離散的な状態空間での山登り法では、以下の手順で探索を行なう。

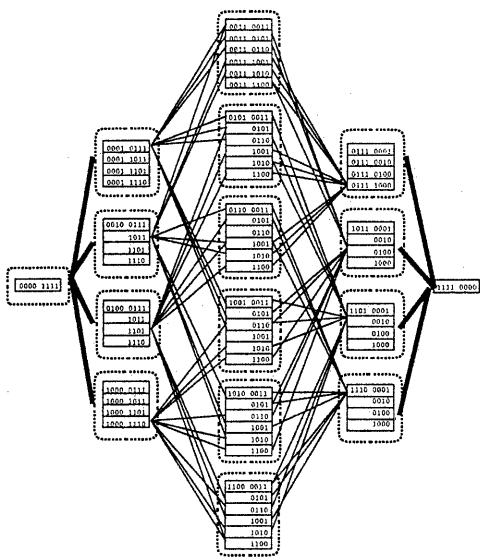


図 12: 探索空間の隣接構造 (8 次元の場合)

- step 0) 離散的な状態空間の各地点に、「隣接地点<sup>1</sup>」を定義する。
- step 1) 探索の始点を決定 (通常はランダムに探索空間から選出) し、この地点を  $x$  とする。さらにこの地点の関数値  $f(x)$  を求め、 $f_{max} = f(x)$  とする。
- step 2)  $x$  に対する隣接地点 ( $x'$  で表す) の関数値  $f(x')$  を求め、それらの関数値の最大値を  $f'_{max} \equiv \max_{\{x'\}} f(x')$  (ただし、 $f'_{max} \equiv f(y)$ ;  $y \in \{x'\}$ ) とする。
- step 3)  $f_{max} \leq f'_{max}$  ならば、

$$f_{max} \leftarrow f'_{max} (\dots \text{最大値の更新});$$

$$x \leftarrow y (\dots \text{探索地点の移動})$$

として step 2) を繰り返す。

$f_{max} > f'_{max}$  ならば、 $x \dots$  現在の探索地点 および  $f_{max} \dots$  現在の最大値 を探索ゴールとして探索を終了する。

2 分割問題の探索空間での山登り法は、隣接状態を以下のよう定義して、上記 step 1) ~ step 3) の手順で探索する。ただし、2 分割問題では探索目標が (コストの) 最小値であるため、正しく表現するならば、「山下り法」であるが、「 $f(x)$  の最小値を求める問題」を「 $-f(x)$  の最大値を求める問題」とみなせば山登りと同じに扱えるので、表現はこのままとする。

• 隣接状態の定義: ある 2 分割状態 ( $N = 2n$  個のノードが、 $n$  個ずつ、 $A, B$  の 2 グループに分割された状態) に対し、「 $A$  に属している 1 つのノード  $a \in A$  と、 $B$  に属し

<sup>1</sup>隣接地点について: 通常の  $\{0/1\}^N$  (2 進  $N$  次元) 空間においては、「ある地点  $x$  に対し、 $x$  とのハミング距離が 1 の地点、即ち  $N$  次元中 1 つの次元のみにおいて値が異なっている地点」が、「 $x$  に対する隣接地点」である。この場合、状態空間の各地点は、 $N$  個の地点と互いに隣接している。

しかしながら、2 分割問題の探索空間においては、後の記述にあるように、「ある地点  $x$  に対し、 $x$  とのハミング距離が 2 の地点、即ち  $N$  次元中 2 つの次元のみにおいて値が異なっている地点」が、「 $x$  に対する隣接地点」となり、状態空間の各地点は、 $(N/2)^2$  個の地点と互いに隣接している。

ている 1 つのノード  $b \in B$  の交換 (これをノードペア  $a \& b$  の交換とよぶ) により、もとの 2 分割状態と異なる 2 分割状態が得られる。そこで、「ある 2 分割状態に対し、1 対のノードペアの交換で得られる状態」を「もとの状態の隣接状態」とよぶことにする。

ある 2 分割状態においてノードペアの異なる選び方は、 $n^2$  通りの組合せが可能である。山登り法による解法では、「 $n^2$  通りのいずれのノードペアの組合せで交換しても交換前よりコストが高くなる」という 2 分割状態 (= コストの局所的最小値を与える状態) に到達するまで、上記のノードペアの交換による状態遷移 (探索地点の移動) を行なう。

山登り法による具体的な解法としては、「隣接状態の探索の深さ」および、「隣接状態の生成順序に関する規則」の異なる、以下に示す  $\{S, N, R, Z\}$  の 4 通りを考えた。

S = "steepest ascent" では、『ある状態で「全」隣接状態のコスト値を調べ、「最もコスト値の低くなる状態」に遷移する』というプロセスを (local) minimum に到達するまで繰り返す。

N = "next ascent" では、『ある状態から、「ある順序で」隣接状態のコスト値を調べ、「コスト値の低くなる状態」がみつかり次第、その状態に遷移する』というプロセスを (local) minimum に到達するまで繰り返す。

R = "next ascent random" では、『ある状態から、「ランダムな順序で」隣接状態のコスト値を調べ、「コスト値の低くなる状態」がみつかり次第、その状態に遷移する』というプロセスを終了条件<sup>2</sup>をみたすまで繰り返す。

Z = "next ascent (R & N)" では、R ("next ascent random") の状態遷移のプロセスを R の終了条件をみたすまで繰り返し、その最終状態を探索の始点としてひきつづき、N ("next ascent") 探索を行なう。

4 通りのうち、「隣接状態の探索の深さ」についてみると、S だけが、常に「隣接状態の全探索」を行なうのに対し、N, R, Z の 3 つは、ピーク到達時にのみ結果的に「隣接状態の全探索」を行ない、通常の探索は、良くなる状態がみつかり次第遷移するという、「部分探索」をしている。(ただし R については終了条件に関する脚注にあるとおり、「全探索」でなく、「準全探索」をピーク到達時に行なう。)

他方、「隣接状態の生成順序に関する規則」についてみると、S と N は、生成順序に関する周期的な規則を用いるのに対し、R と Z は、ランダムに隣接状態を生成する。ただし Z は「R 探索のあとで N 探索することにより、R 探索における、「隣接状態の生成もれ」のケース」すなわち探索のランダム性由来して、生成されない隣接状態が残っているにもかかわらず、探索が終了してしまうケースを防ぐ。

## C Kernighan & Lin のアルゴリズムによる解法

この解法は、従来経験的に最も良い解を見つけ出すことができる、と考えられているアルゴリズムであり、以下に示される手順により探索を行なう。(図 13 参照。)

- step 0)  $2n$  ノードのある 2 分割状態を探索の始点とする。

- step 1) 現在の 2 分割状態に対し、「交換による gain (= cost の減少量) が最大となるようなノードペア ( $a_1 \& b_1$ ;  $a_1 \in A, b_1 \in B$ )」を  $n^2$  通りの組合せの内か

<sup>2</sup>終了条件について: 上記 S と N に関しては、『ある状態で「全」隣接状態のコスト値を調べた末 自分がコスト最小である』場合を終了状態 (local minimum) とみなせるが、R に関しては、探索のランダム性から、『ある状態の「全」隣接状態を調べたかどうか』は、常に確率的問題である。そこで、人為的に、R においては、crit = 「ある状態の隣接状態の個数 (=  $N \times N$ )」回 idling (= 状態遷移しない状態) したなら、終了条件を満たした、とみなす。

ら求め、このときの gain の値を  $g_1$  として、 $a_1$  と  $b_1$  は、交換されたものとして固定<sup>3</sup>する。

- step 2) 固定されていない残りの  $2(n-1)$  ノードについて、同様に「交換による gain (= cost の減少量) が最大となるようなノードのペア ( $a_2$  &  $b_2$ ;  $a_2 \in A - \{a_1\}, b_2 \in B - \{b_1\}$ )」を  $(n-1)^2$  通りの組合せの内から求め、gain の値を  $g_2$  として、 $a_2$  と  $b_2$  も、交換されたものとして固定する。
- step 3)  $i = 3 \sim n$  まで、{} 内を反復する。  
(固定されていない残りの  $2(n-i)$  ノードについて、「交換による gain (= cost の減少量) が最大となるようなノードのペア ( $a_i$  &  $b_i$ ;  $a_i \in A - \{a_j; j = 1 \sim i\}, b_i \in B - \{b_j; j = 1 \sim i\}$ )」を  $(n-i)^2$  通りの組合せの内から求め、gain の値を  $g_i$  として、同様に  $a_i$  と  $b_i$  も、交換されたものとして固定する。  
(全ノードを  $A, B$  間で交換しても cost 値は変わらないから、当然、 $\sum_{i=1}^n g_i = 0$  である。)
- step 4) 以上のプロセスで得られた gain の値:  $\{g_i; i = 1 \sim n\}$  を基に、 $\sum_{i=1}^k g_i = G_k$  を最大とするような  $k$  の値  $K$  を選ぶ。 $(G \equiv \max_k G_k)$  このとき  $G > 0$  であるならば、 $\{a_i; i = 1 \sim K\}$  と  $\{b_i; i = 1 \sim K\}$  を交換したものを新たな 2 分割状態の始点として、上記の step 1 ~ 3 のプロセスを繰り返す。 $G < 0$  であるならば、現在の 2 分割状態を探索ゴールとし、探索を終了する。

上記 step 1 ~ 4 までの 1 回のプロセスを以後「KL バス (path)」とよぶ。K&L 法による探索では、この「KL バス (path)」を通常数回反復することにより、解が求められる。

状態 (0) は、step 0) における「仮の 2 分割状態」(initial state) であり、リスト表現すれば、

上段 = {a,b,c,d,e}、下段 = {A,B,C,D,E}

である。この状態において、上段のノード 1 つと下段のノード 1 つの  $5 \times 5 = 25$  通りの組合せのうち、交換による gain が最大となる組合せを求める。(図では a と A の交換で、最大 gain  $g_1$  としている。) 求められたノードペアを交換した状態が状態 (1) で、a と A が交換されたのち固定される。(固定されたノードは、図中点線で囲まれている。) 次に状態 (1) において、固定されていない (= 図中実線で囲まれている) ノードについて、 $4 \times 4 = 16$  通りの組合せから交換による gain が最大となる組合せを求める。(図では b と B の交換で、最大 gain  $g_2$  としている。)

状態 (1) からさらに b と B を交換 & 固定した状態が状態 (2) である。状態 (2) において、可能な  $3 \times 3 = 9$  通りの交換組合せ中、最大 gain  $g_3$  を与えるノードペア (図では c & C としている) を交換 & 固定すると、状態 (3) となる。

同様のプロセスの反復により、1 回の「KL バス」の最終状態として、図の状態 (5) に到達する。状態 (5) は「状態 (0) の上下段を入れ換えた状態」に対応している。

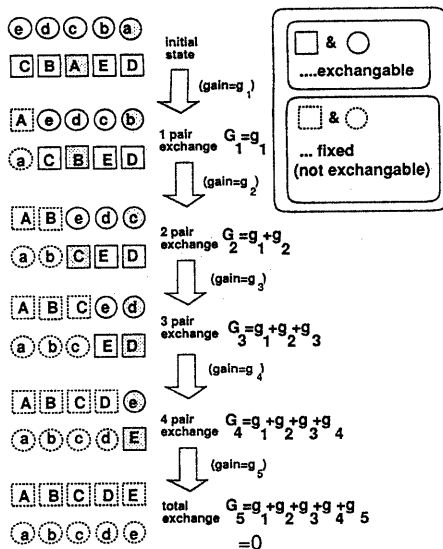


図 13: K & L 法の説明図

上記 step 0 ~ 3 までのプロセスを図示したものが図 13 である。同図は {a,b,c,d,e,A,B,C,D,E} の 10 ノードの 2 分割問題を例としている。図中の (0),(1),...,(5) は、この場合の「KL バス」1 回で通過する状態を示し、各々の状態は 2 分割された 10 ノードを「上段の集合と下段の集合」として表現している。

<sup>3</sup>ここでいう固定とは、「その後の探索において、交換の対象から除外すること」即ち、そのノード (ペア) は、2 分割により属することとなるグループが (一時的に) 確定したことを意味する。