

Stackelberg型施設配置問題への 遺伝的アルゴリズムの応用

大 角 盛 広

概 要

一定の距離以上離れた施設は利用しないという性質を持つ需要点が離散的に分布する平面上の市場において、先手後手の区別のある競合する2企業が、需要点から獲得できる購買力の最大化を目的として、施設の配置を決定する問題を考察する。先手は、後手が直後に参入してくることを考慮した上で、最終的な自分の利得が最大になるように配置を決定せねばならない。先手の最適配置を求める問題をCentroid問題、後手の最適配置を求める問題をMedianoid問題として定式化し、近似解を遺伝的アルゴリズムにより求める方法を提案するとともに、厳密解を求めるアルゴリズムと比較する数値実験の結果を示した。

キーワード: 施設配置問題, 非ゼロ和ゲーム, 遺伝的アルゴリズム

1 はじめに

本論文では、先手後手の区別のあるStackelberg型の競合施設配置問題について考察する。2者による競合施設配置問題は、直線上の市場における配置競争を論じたHotelling [1]のモデルから考えられ始めた。Hakimi [2]は1983年にネットワーク上でこの問題を考察し、次の2つの問題を定式化した。既存施設に対抗する最適配置を求めるMedianoid問題と、後から競合者が最適配置を行うことを考慮に入れた上で自己の施設の最適配置を求めるCentroid問題である。Centroid問題の解はStackelberg型の解となるが、計算量はNP-hardとなることが示されている。

遺伝的アルゴリズム (Genetic Algorithm, 以下GA) は、Hollandによって1975年に提案された最適化の手法である。これは、選択淘汰・突然変異という生物進化の原理を応用した最適化のためのアルゴリズムであり、確率的探索に学習の要素を入れたものである。すでに一般の施設配置問題への応用もあるが [3][4][5][6], 競合施設配置問題

への応用はまだ少ない。特に、本論文のように先手後手の区別のある競合施設配置問題への応用はまだ見られない。

本論文では、Stackelberg 型の配置問題のうち、従来列挙法でしか解けなかった問題について、GA を用いて近似解を求め、厳密解との比較でその性能を示す。

2 モデルと定式化

N 個の需要点 (顧客) が分布する平面市場に、2 企業 X, Y が順に 1 つずつ施設を配置する問題を扱う。各企業の目的は、需要点から獲得できる購買力の合計の最大化である。顧客の選好に関して、以下の仮定を設ける。

1. 限界距離 R_i 以内の施設は利用し、 R_i より遠い施設は利用しない
2. 限界距離 R_i 以内に複数の施設がある場合は、それらを均等に利用する

なお、 R_i は顧客ごとに異なる限界距離である。

顧客の施設利用圏に限界距離がない場合には、ゼロ和ゲームになるので、先手後手ともに相手の獲得する購買力を減らすことが自己の利得の最大化になる。限界距離がある場合には、最初に出店する先手企業が、その場面で最も多く取れる場所に置かず、後手企業に取り分を残しておいた方が最終的に利得を多くできる場合があり、そのような具体例は、大角・塩出 他 [8] で述べられている。

需要点の位置を P_i 、その購買力を W_i とする。任意の位置 X, Y の間の距離は距離関数 $d(X, Y)$ で表されるとする。このとき、 P_i の利用圏 D_i とその境界 B_i は、

$$D_i = \{Q \mid d(Q, P_i) \leq R_i\}, \quad B_i = \{Q \mid d(Q, P_i) = R_i\}$$

と表せる。一方、先手 X の施設の位置を X 、後手 Y の施設の位置を Y とし、 X, Y 側から需要点の位置をみると、 X が需要点を取り込める勢力圏 $D(X)$ とその境界 $B(X)$ は次のように表せる。 Y についても同様である。

$$D(X) = \{Q \mid d(Q, X) \leq R_i\}, \quad B(X) = \{Q \mid d(Q, X) = R_i\}$$

X, Y の得る利得は、需要点から得る購買力の合計に等しいと考え、 X, Y が配置されたときの X, Y の利得 G_X, G_Y を、

$$\begin{aligned} G_X(X, Y) &= \frac{1}{2} \sum_{P_i \in D(X) \cap D(Y)} W_i + \sum_{P_i \in D(X) \cap \overline{D(Y)}} W_i \\ G_Y(X, Y) &= \frac{1}{2} \sum_{P_i \in D(X) \cap D(Y)} W_i + \sum_{P_i \in \overline{D(X)} \cap D(Y)} W_i \end{aligned}$$

で表す。

このとき、 X に対する Y の最適反応戦略 R_Y は次のように表される。

$$R_Y(X) = \{Y^* \mid G_Y(X, Y^*) = \max_Y G_Y(X, Y)\}$$

X に対抗して Y がただひとつの戦略を採用するとすれば、次の最適反応戦略集合 D_Y が得られる。

$$D_Y = \{(X, Y) \mid X \in \mathcal{R}^2, Y \in R_Y(X)\}$$

ここで、 X^* と Y^* が次の関係を満たすとき、Stackelberg 均衡にあるという。

$$G_X(X^*, Y^*) = \max_{(X, Y) \in D_Y} G_X(X, Y), \quad Y^* \in R_Y(X^*)$$

Y^* を求める問題を Medianoid 問題といい、 X^* を求める問題を Centroid 問題という。一般に、Medianoid 問題よりも Centroid 問題を解く方が難しい。なぜなら、 X は、 Y があとから利己的に最適配置を行うことを考えに入れた上で、自身の最適配置を決めなければならないからである。

なお、この Stackelberg 解 X^* は、あとから Y がどんな位置に配置した場合でも X に最低限 $G_X(X^*, Y^*)$ だけの利得を保証する、というものではない。なぜならば、これは Y が最適反応戦略 $R_Y(X^*)$ を採ることを前提にして成り立つ解であり、もし Y がそうした合理的な戦略を採らず Y 自身にとって不利な判断基準を採用したり、ランダムに配置したりする場合には、 X は $G_X(X^*, Y^*)$ よりも少ない利得しか得られない場合があるからである。

本論文では、距離関数として直角距離を用いることにする。すなわち、 $X = (x_1, x_2)$ と $Y = (y_1, y_2)$ の距離 $d(X, Y)$ は

$$d(X, Y) = |x_1 - y_1| + |x_2 - y_2|$$

で表されるものとする。このとき、 B_i や $D(X)$ は視覚的には 45° 傾いた正方形となる。直角距離は、移動方向が 2 方向に制限される場合のモデルであり、碁盤目状に道が広がる京都やニューヨーク・マンハッタン地区などのような都市部では、ユークリッド距離よりも良い近似を与えることが知られている。

3 遺伝的アルゴリズムの適用

GA の概要は、以下の通りである。

- Step 1. 初期個体 (individual) 集団の生成：乱数を用いて最初の世代を生成
- Step 2. 適応度 (fitness) の評価：各個体について問題への適応度を評価
- Step 3. 選択 (selection)：適応度に応じて交配できる親個体を選択
- Step 4. 交叉 (crossover)：交配で子孫を残す際に交叉率 C で遺伝子の交叉が発生
- Step 5. 突然変異 (mutation)：子孫の遺伝子の一部が突然変異率 M の割合で変化
- Step 6. 子孫を次の世代の親として Step 2 へ

Step 2～5 の1回の繰り返しが1世代で、この世代交代を繰り返しつつ、各世代の最も適応度の高い個体の遺伝子を問題の解とする。GA は、この基本的な考え方以外は自由度が大きい。以下、本論文で扱う配置問題を GA で扱えるように設計する。

3.1 遺伝子コーディング

表現型 (phenotype) によって規定される問題を問題空間、遺伝子型 (genotype) によって規定される空間を GA 空間と呼ぶ。問題空間から GA 空間へどのように対応付けるかを遺伝子コーディングと呼ぶ。

本論文では、座標値を表す表現型として 16 ビット符号無し整数を用いる。これは 0 から 65535 までの整数を表すことができるので、1メートル単位での精度の問題および解を考えるならば、65.5km 四方の領域をカバーできる。これは面積にして大阪府の約 2.3 倍、ニューヨーク・マンハッタン地区の約 49 倍で、都市部での配置を問題にする場合には、十分な広さである。

平面上の1点を1個体として、各個体は x 座標と y 座標用に 16 ビット固定長の2つの遺伝子を持つものとする。最初に述べた 16 ビット符号無し整数を、以下に述べる Gray コードを用いて遺伝子型に変換する。

Gray コード

コンピュータ内部の2進数による符号無し整数表現では、隣り合った数値の間で異なるビットの数、すなわちハミング距離 (Hamming distance) は一定ではない。例えば 10 進数で隣り合う 1 と 2 と 3 は、2 進数で 2 桁で書くとそれぞれ 01, 10, 11 であり、1 と 2 ではハミング距離 2、2 と 3 ではハミング距離 1 で、等しくない。GA では一般的にビット単位で遺伝子を変化させるので、ある値と隣接する値のハミング距離が常に 1 であるという性質を持つ Gray コードを用いる方が望ましいと考えられている。

2 進数から Gray コードとの相互変換を行うには、次の関係を用いればよい。すなわ

ち、ビット長 ℓ の 2 進数 b を最上位桁から順に $b_{\ell-1}, b_{\ell-2}, \dots, b_1, b_0$ とし、同じくビット長 ℓ の Gray コード g を最上位桁から順に $g_{\ell-1}, g_{\ell-2}, \dots, g_1, g_0$ とすると、

$$g_k = \begin{cases} b_{\ell-1} & , k = \ell - 1 \\ b_{k+1} \oplus b_k & , k \leq \ell - 2 \end{cases}$$

である。ただし、 \oplus は排他的論理和を表す。

なお、Gray コードは、隣接する値のハミング距離が 1 であることを保証するだけで、1 ビットの変化が元の値に及ぼす影響は考慮されていない。そのため、1 ビットの変化が元の値に及ぼす影響の平均が一定以下になるような均質コーディング [9] と呼ばれるコーディング方法も提唱されている。その他、コーディングに関しては浜田穂積による万能数値表現法 (URR: Universal Representation of Real numbers) も興味深い。

本論文の数値実験における実装では、 x, y 座標の表現型であるバイナリ値をそのまま遺伝子型の値としておき、交叉や突然変異の処理を行う直前に、Gray コードに変換している。このようにしておけば、交叉や突然変異が起った場合にのみ表現型と遺伝子型との相互変換をすれば済む。交叉も突然変異も毎回起こるものではないから、少しでも処理時間が短くなることが期待できる。

3.2 適応度と選択方法

ある世代の個体集団において、個体すべての適応度の評価を行い、それをもとに交配することのできる個体を選択する。適応度に基づく選択方法はいくつか考案されているが、本論文ではルーレットルールを用いる。すなわち、 P 個の個体集団の中のある個体の適応度を f_i で表すとき、その個体を選択される確率 p_i は、

$$p_i = \frac{f_i}{\sum_{j=1}^P f_j}$$

で与えられるものとする。

このルーレットルールを用いる際、適応度 f_i の値をそのまま選択確率に反映させるのではなく、スケーリング (scaling) 関数を適用した値を使うことも多い。しかし、本モデルでは、利得関数の値は $0 \leq G_X, G_Y \leq \sum_{i=1}^N W_i$ の範囲に収まり、値を補正する必要性も薄いため、スケーリング関数を通さず G_X, G_Y の値をそのまま f_i として用いて選択確率を計算するものとする。

なお、本論文の実装では、全世代を通して個体数は一定とする。すなわち、1 回の交配で 2 つの親から 2 つの子が生まれて次の世代に移行する。従って、1 世代あたりの交

配は $P/2$ 回起きることになり、そのたびにルーレットルールで親が選択される。適応度の高い個体は、1回の世代交代の間に複数の交配に選択される可能性がある一方で、適応度が0の個体は子孫を残せず1世代で死滅する。

エリート保存戦略

ある世代の個体集団の中で最も高い適応度を持つ個体を、必ず選択して交配させるのがエリート保存戦略である。一般のエリート保存戦略では、エリート遺伝子は後述する交叉や突然変異からも免れ、遺伝子そのままにコピーされる。この戦略を採用すると、その世代の最良の解が交叉や突然変異によって破壊されないという利点がある。しかし一方で、エリート遺伝子が集団中に急速に広がり局所解から抜けにくくなる可能性もある。

本論文のモデルでは、局所解を与える領域が多数あることが見込まれるため、実装にあたっては、エリート個体は必ず選択されるが非エリート同様に交叉や突然変異の影響も受けるという、現実の生物進化に近づけた弱いエリート保存戦略というものを考え、用いることにした。この戦略の有効性は、4.2節の数値実験結果で示される。

3.3 交叉と突然変異

選択された親個体の遺伝子を子個体の遺伝子にコピーする際、通常は親の遺伝子をそのまま子の遺伝子にコピーする。ただし、確率 C で親同士の遺伝子の交叉が起こり、その結果が子にコピーされる。また、子の遺伝子には確率 M で突然変異が起こる。これがGAの特徴であり、最適な C, M の値は数値実験を通して経験的に決めていくのが一般的である。なお、交叉と突然変異は排他的に起こるわけではない。

交叉

長さ ℓ の固定長のビット列で表現された遺伝子 a, b があり、最上位桁から順に $a_{\ell-1}, a_{\ell-2}, \dots, a_1, a_0$ および $a_{\ell-1}, a_{\ell-2}, \dots, a_1, a_0$ とする。このとき、乱数で決めた2つの整数 i, j ($0 \leq i \leq j < \ell$) を使い、

$$a_k \text{ と } b_k \text{ の値を交換} \quad \text{for all } k, i \leq k \leq j$$

の操作を行うことが2点交叉である。 i だけを決めて $j = \ell - 1$ とする場合を1点交叉と呼ぶ。本論文では、2点交叉を用いる。

1つの個体は x, y 座標それぞれについて遺伝子を持っているので、これを1本の染色体にまとめてから交叉を行うことも考えられるが、スキーマタ (schemata) の保存の観点から、本論文ではそれぞれ別々の遺伝子として扱い、確率 C でいずれか一方のみ交叉を行わせている。

突然変異

長さ ℓ の固定長のビット列で表現された遺伝子 a があり、最上位桁から順に $a_{\ell-1}, a_{\ell-2}, \dots, a_1, a_0$ であるとする。このとき、乱数で決めた整数 i ($0 \leq i < \ell$) を使い

$$a_i = \begin{cases} 0 & , a_i = 1 \\ 1 & , a_i = 0 \end{cases}$$

とするのが最も一般的な突然変異であり、本論文でもこれを採用する。その他、ある範囲を変化させる摂動や、要素を入れ替える逆位・転座や、遺伝子長を変化させるものなどが考えられている。

4 厳密解法とGAによる解の探索

Medianoid問題とCentroid問題をGAを使って実際に解き、実行時間と解の近似度について検討する。性能評価のために、厳密解を求める列挙法との比較を行う。

数値実験用のプログラムは、列挙法によるものもGAによるものも、C言語を用いて実装した。開発及び実行環境は以下の通りである。

コンパイラ	gcc 4.3.4
OS	KNOPPIX 6.2 (Debian GNU/Linux)
CPU	ThinkPad T60p, 2GB RAM
	Intel Core 2 Duo (T7600) 2.33GHz

なお、コンパイル時には特別な最適化オプションは付けていない。

数値実験用のデータとしては、直角距離で直径10kmの市場に、利用圏の半径 $R_i = 1\text{km}$ の需要点が離散的に存在し、それぞれの需要点は1以上100以下の実数の購買力 W_i を持つものとして、 N 個の需要点のデータを乱数で生成した。

4.1 列挙法による厳密解の探索

すべての利用圏の境界 B_i によって分割された平面上のすべての領域に D_1, D_2, \dots とラベルを付けると, D_i 内で獲得可能な購買力の総和 $S_i = \sum_{k: D_i \cap D_k \neq \phi} W_k$ は一定である。従って, 前処理として D_i 内の1点を解候補として列挙し, その中から最適解を探索する。実装上は, D_i の x, y 座標値が最も小さい頂点を解候補の代表点として列挙するが, 以下, 新たなラベルは付けず, D_i を領域の意味でも代表点の頂点の意味でも用いる。

直角距離を用いた場合, B_i の形は正方形であり, 正方形の重なりを判定し列挙するには Imai [7] で示されている $O(N \log N)$ のアルゴリズムが利用できる。列挙後, S_i の降順に D_i を整列し, 添え字を付け直す。すなわち, 最大の購買力 S_1 を持つ領域が D_1 であるとする。このとき, 厳密解 X^*, Y^* は以下のアルゴリズムで求められる。

Step 1. $i \leftarrow 1, m \leftarrow 0$

Step 2. $X \leftarrow D_i$ とする。 $P_k \in D(X)$ である P_k について $W_k \leftarrow \frac{1}{2}W_k$ とする。
 $G_Y(X, D_j)$ の最大値を与える D_j を線形探索し, $Y^* \leftarrow D_j$ とする。

Step 3. すべての W_i を初期値に戻す。 $P_k \in D(Y^*)$ である P_k について $W_k \leftarrow \frac{1}{2}W_k$ とする。
 $G_X(X, Y^*) > m$ ならば $m \leftarrow G_X(X, Y^*)$ とし, $X^* \leftarrow D_i$ とする。

Step 4. $G_X(X, Y^*) > \frac{1}{2}S_1$ ならば $i \leftarrow i+1$ として **Step 2** へ, そうでなければ終了。

X, Y とも最悪でも D_1 に置いて S_1 を半分ずつ分け合うという解があるので, $S_1/2$ 未満しか取れない領域が解になることはありえない。従って, 解の探索において, これを下界として探索を打ち切ることができる。

列挙法の実行時間

このアルゴリズムの計算量は, 領域 D_i の個数に依存するので, D_i の個数について考察する。1つの閉曲線は平面を2つの領域に分割するから, 一般に N 個の閉曲線は平面を最大 2^N 個の領域に分割する。しかし, 本論文の場合は閉曲線は直角距離での円に限定されているため, N 個目の円が追加されたとき, 既存の $N-1$ 個の円それぞれと高々2回ずつしか交わらず $2(N-1)$ 個の領域が新たに増えるにすぎない。従って, 領域数のオーダーは $O(N^2)$ である。

Medianoid 問題を解くには, **Step 2** にあるように領域 D_i を線形探索するので, 領域の数に比例した計算手間 $O(N^2)$ がかかる。また, Centorid 問題については, ある X に対して Y の最適反応戦略 Y^* が決まって初めて利得 $G_X(X, Y^*)$ が定まるが, それを

各領域 D_i について繰り返して最大値を探索するため、 $O(N^2 \times N^2) = O(N^4)$ の計算時間がかかる。

表 1: 需要点数 N と領域数・実行時間の増え方

需要点数 N	領域数	秒
10	15	0
50	298	0
100	1237	1
150	2837	10
200	5122	60
250	8152	223
300	11421	790
350	15408	1877
400	20286	4309

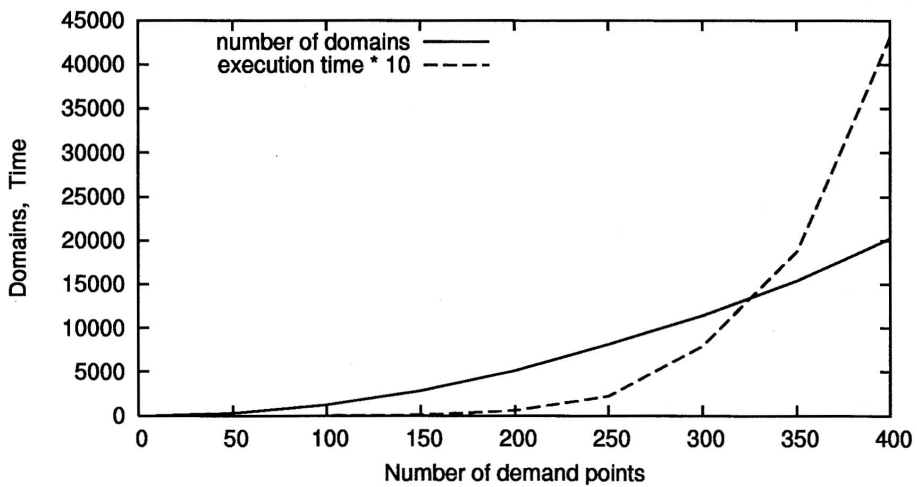


図 1: 需要点数 N と領域数・実行時間の増え方

表 1 は、需要点の数 N を 10～400 まで増やした場合の領域数の増え方と Centroid 問題を解くまでの実行時間の増え方を実測したもので、図 1 はそれをグラフ化したもの

である。ただし、図1では、縦軸の縮尺を合わせるため、実行時間を10倍した数値でグラフを描いている。また、実行時間0秒とは、1秒未満であるとの意味である。

需要点400点で実行時間は約72分かかっている。実行時間を最小二乗法により x^4 で近似すると $1.51 \times 10^{-7}x^4$ となり、1000点では約42時間、2000点で約28日かかると推定される。

4.2 Medianoid 問題への GA の適用

ある X に対する Medianoid 問題に、実際に GA を適用して解いた結果を示す。以下では、特に断らない限り、需要点数 $N = 100$ 、世代数 $G = 200$ として、それ以外のパラメータを次のように変えて数値実験を行っている。

個体数 P	10, 50
交叉率 C	0.2, 0.02
突然変異率 M	0.2, 0.02

ただし、今回はパターンが増えすぎないように $C = M$ として4パターンでの数値実験を行った。

GA では乱数を多用するが、疑似乱数については、gcc の標準関数で移植性が高い `drand48()` を用いた。これは、64ビットの線形合同法で演算を行い、そのうち48ビットを使用するので $2^{48} =$ 約280兆の周期を持つ。今回の実験での `drand48()` の呼び出し回数は、需要点1個のデータの生成に x, y 座標値と購買力 W_i で3回、GAでの1個体1世代につき交叉有無の判定に1回、交叉の位置決定に2回、突然変異有無の判定に1回、突然変異の位置決定に1回である。後述するように Centroid 問題の計算手間から、 G 世代目の最良の解を得るまでに $3N + 5 \cdot P^2 \cdot G^2$ の `drand48()` の呼び出しが必要である。今回の数値実験では、 $N = 100, P = 50, G = 200$ で30回繰り返すので、`drand48()` の呼び出しは約150億回であるから、280兆の周期は十分な長さである。

なお、同じく標準関数の `rand()` は周期が約20億で使えず、`random()` は乱数生成時間が遅いので今回のような用途にはやや不向きである。

図2は、乱数により P_i の位置や W_i 、初期集団などの初期条件を変えて、200世代までの数値実験を30回ずつ繰り返した平均での、解の適応度の推移を表すグラフである。横軸が世代であり、縦軸がその世代の最高の適応度を持つ個体の適応度である。ただし、適応度については、 G_Y の値そのままではなく、列挙法で求めた厳密解の適応度を1として正規化してある。

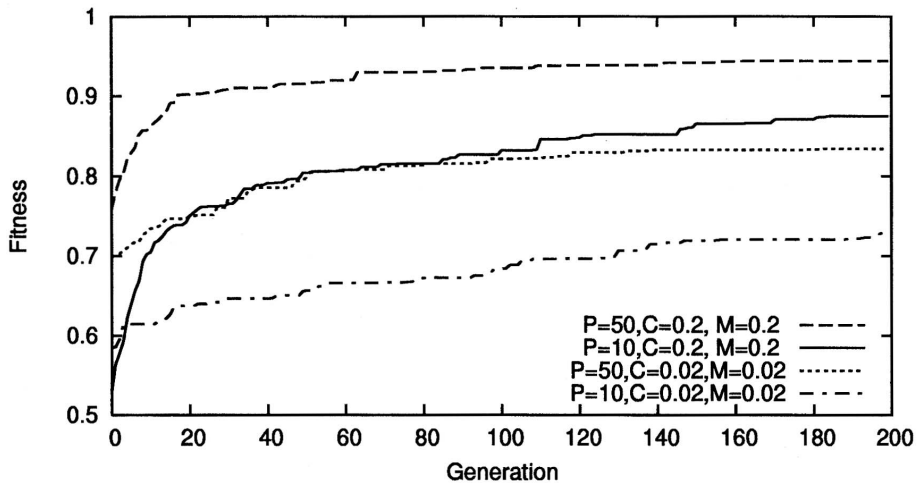


図 2: Medianoid 問題における解の適応度の推移 (30 回の平均)

解の近似度

図 2 から読み取れるように, $C = 0.2, M = 0.2$ の場合については, 最初の 20 世代あたりまでの解の改善が急速であり, 個体数 $P = 50$ ではこの時点で平均的に厳密解の 90 パーセント以上の近似解が得られた。個体数 $P = 10$ でも, 200 世代では平均的に厳密解の 87 パーセント以上の近似解が得られており, 適用対象によっては実用的な範囲と思われる。

実行時間

ある 1 世代の 1 個体について, N 個の需要点との距離の計算が必要になるので, 全体の計算時間のオーダーは $O(N \cdot P \cdot G)$ である。どのパターンにおいても 200 世代目までの実行時間は 1 秒未満であったため, 表は省略する。

パラメータと解のふるまい

図 2 のグラフだけでは, 世代が進むに従って適応度がなめらかに単調に増加するように読み取れてしまうが, これは 30 回の平均をグラフ化してあるためであり, 1 回ずつの実験結果を見ると別の様相が見て取れる。平均をとると消えてしまう情報を見るため, また, C と M がどのように解の推移に影響を及ぼすか見るため, C と M を大きく変えて, 任意に選んだ 3 回ずつの実験結果を個別に図 3~8 に示す。

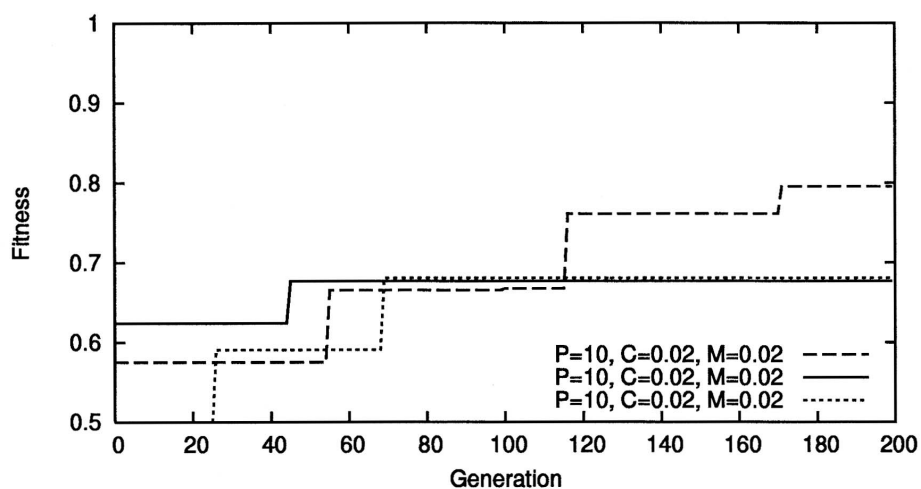


図 3: $P=10, C=0.02, M=0.02$ で任意に選んだ 3 回のグラフ

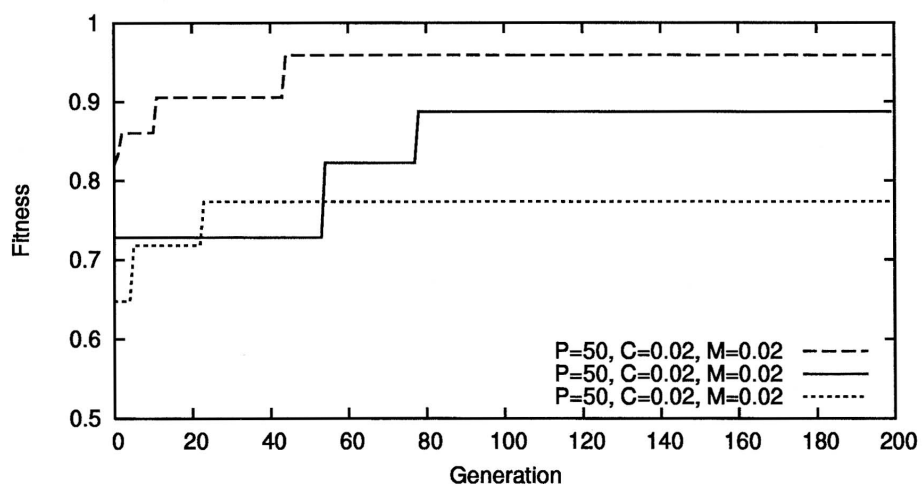


図 4: $P=50, C=0.02, M=0.02$ で任意に選んだ 3 回のグラフ

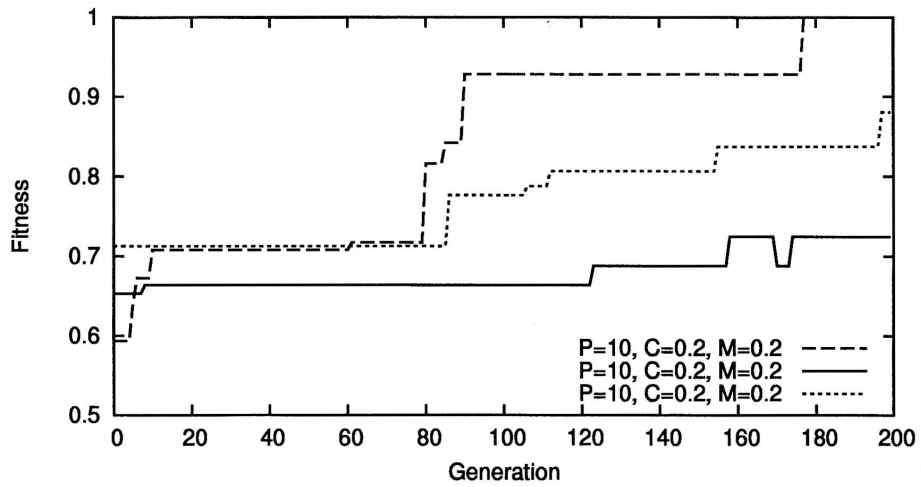


図 5: $P=10$, $C=0.2$, $M=0.2$ で任意に選んだ 3 回のグラフ

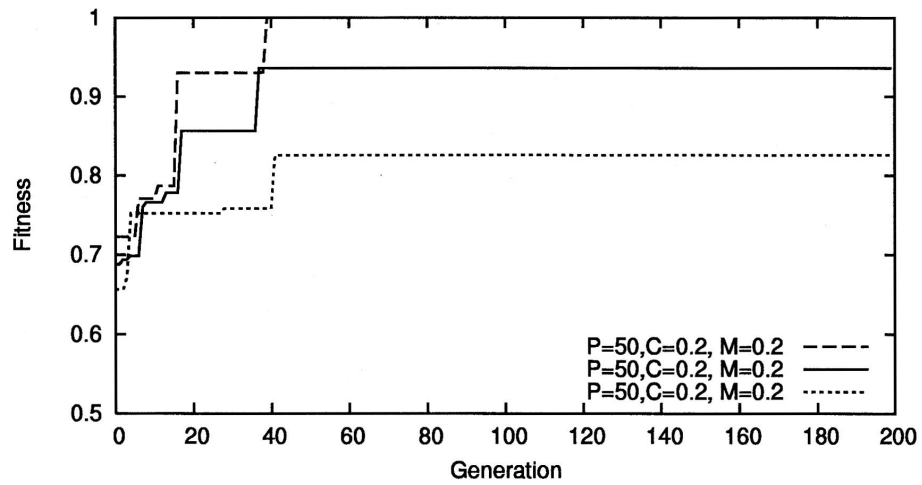


図 6: $P=50$, $C=0.2$, $M=0.2$ で任意に選んだ 3 回のグラフ

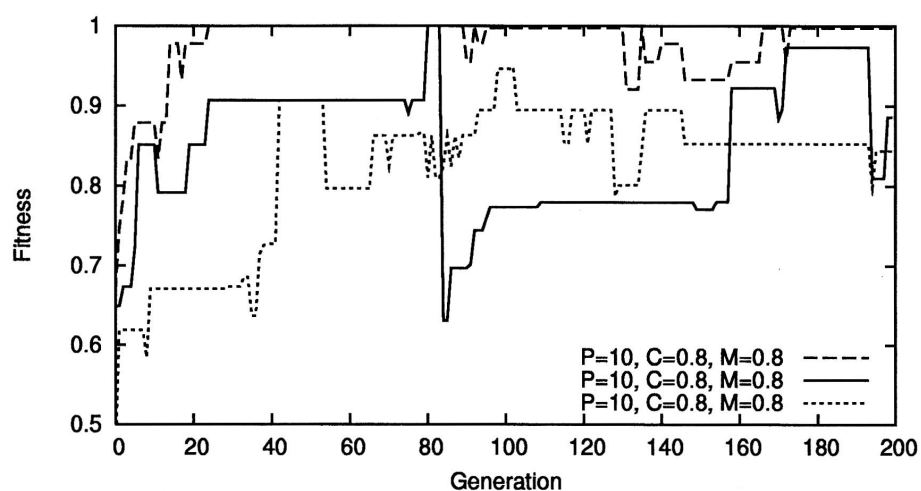


図 7: $P=10$, $C=0.8$, $M=0.8$ で任意に選んだ 3 回のグラフ

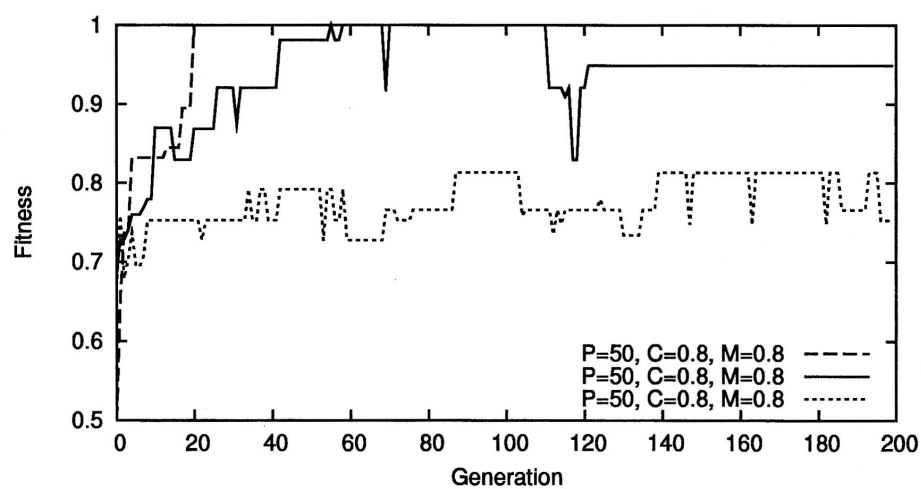


図 8: $P=50$, $C=0.8$, $M=0.8$ で任意に選んだ 3 回のグラフ

図3と図4は、 $C = 0.02, M = 0.02$ と低率の場合の適応度の推移を示している。他の場合よりも解の改善速度がゆるやかであることが見て取れる。なお、階段関数状になっているのは、ある D_i に含まれるか否かで G_Y が離散的に変化するため、平均を取ると図2のようになる。

図5と図6は、 $C = 0.2, M = 0.2$ のグラフであり、 $C = 0.02, M = 0.02$ よりも解の改善速度が向上していることがわかる。ただし、図5の170世代あたりに見られるように、適応度が減少する場合も出てきている。これは、直前の世代で最大適応度を持っていた遺伝子がすべて交叉や突然変異を受け、しかもそれがより良い方ではなく悪い方に働いた例である。一般的な強いエリート保存戦略を採用するとこのような減少は生じないが、図9の実験結果で示されるように、平均的には本論文で考えた弱いエリート保存戦略の方が性能が良い。

図7と図8は、 $C = 0.8, M = 0.8$ のグラフである。他の場合より早い世代で適応度1に達することもあるものの、高確率での交叉と突然変異により、解が安定しないことがわかる。

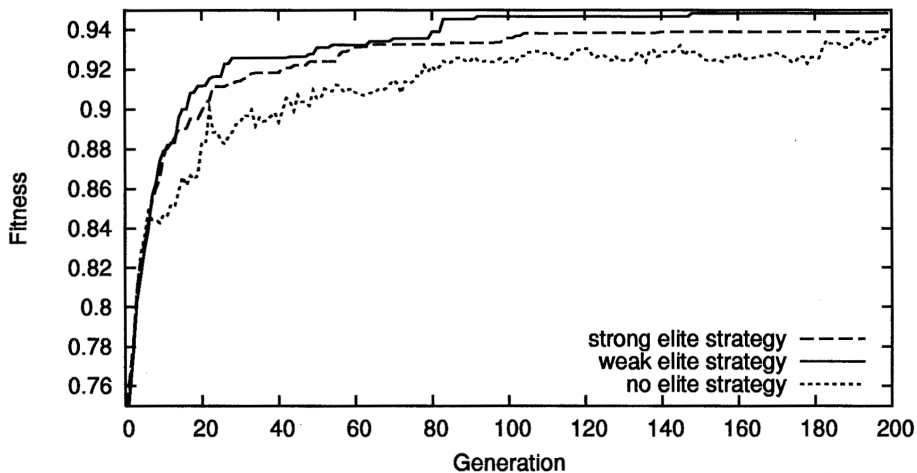


図9: エリート保存戦略の違いによる解の適応度の推移 (30回の平均)

図3～8から読み取れることをまとめると、定性的に次のことがいえる。

1. 個体数の多い方が、適応度が高い状態で推移する
2. 交叉率や突然変異率の高い方が解の改善が速い

3. 交叉率や突然変異率が高すぎると解が安定しない

4.3 Centroid 問題への GA の適用

Centroid 問題に GA を適用する際、 X, Y の最適配置を決めれば良いのだから、それぞれの座標を引数に持つ4次元の関数を使って GA を解けばよい、 X, Y の位置を連結した染色体にまとめて GA で解けばよい、と思えるかもしれないが、以下の事実を考えると、それではうまくいかない。

1. X と Y の目的関数が異なる。すなわち、 G_X の最大化と G_Y の最大化とは異なる。非ゼロ和ゲームであるから、別々に考えなければならない。
2. 先手 X のある1個体に対して、後手 Y の N 個の個体について G 世代まで適応度を計算しなければならない。

従って、本論文の実装では、Stacelberg 問題の定義通り、GA を二重に用いている。すなわち、 X の最適化を担当する GA 手続きは G_X を呼び出し、 G_X は Y の最適化を担当する GA 手続きを呼び出し、その GA 手続きは G_Y を呼び出す。先手 X のある1個体の適応度は、後手 Y がそれに対して GA で求めた適応度最高の場所に置いたとき初めて確定するので、この呼び出し方が定義通りである。実装にあたっては GA 手続きの中に目的関数とデータを保存するスタックを設けて、複数回の再帰呼び出しが可能であるようにした。

図10は、Medianoid 問題におけるグラフと同様、乱数により P_i の位置や W_i 、初期集団などの初期条件を変えて200世代までの数値実験を30回ずつ繰り返した平均での、解の適応度の推移を表すグラフである。Centroid 問題の場合は、近似解法である GA を二重に使うため、Medianoid 問題の場合と比較して GA の性能がかなり悪くなると予想されたが、実際に図2のグラフと比べると、 $P = 10$ の場合の解の改善速度が比較的遅いこと以外には、ほとんど差はない。

解の近似度

図10から読み取れるように、 $C = 0.2, M = 0.2$ の場合については、最初の20世代あたりまでの解の改善が急速であり、個体数 $P = 50$ ではこの時点で平均的に厳密解の90パーセント以上の近似解が得られている。個体数 $P = 10$ でも、200世代では平均的に厳密解の89パーセント以上の近似解が得られており、適用対象によっては実用的な範囲と思われる。

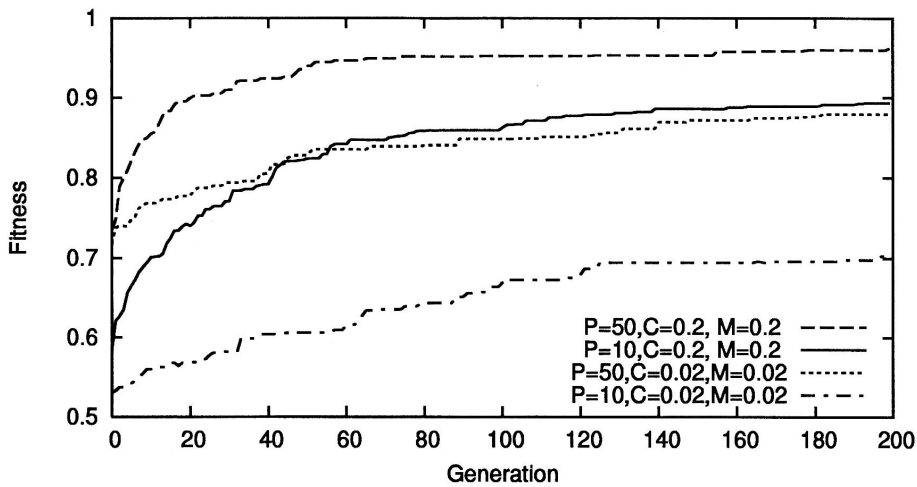


図 10: Centroid 問題における解の適応度の推移 (30 回の平均)

実行時間

すでに示したように GA で Medianoid 問題を解く場合の計算量は $O(N \cdot P \cdot G)$ で、Centroid 問題を解くためには、それを先手の全個体 P 個と全世代 G について行わねばならないので、計算量は

$$O(P \cdot G \cdot (N \cdot P \cdot G)) = O(N \cdot P^2 \cdot G^2)$$

となる。

$N = 50$ で固定し P, G を変えて実行時間を計測した結果を表 2 に示す。

また、 $P = 50, G = 100$ で固定し N を変えて実行時間を計測した結果を表 3 に示し、そのグラフを図 11 に示す。ただし、列挙法を用いた場合の実行時間の増え方が $O(N^4)$ で急激であり、グラフにすると見づらくなるため、図 11 のグラフは 300 世代までとした。GA で Centroid 問題を解く場合の実行時間の増え方が N に対して線形であることが、数値実験の結果からも読み取れる。 $P = 50$ の場合の最小二乗法での近似は、 $2.4N - 150$ である。

表 3 と図 11 から、需要点 225 点前後で GA と列挙法の実行時間が逆転し、以後差が拡大していることがわかる。列挙法では需要点 2000 点の場合に解を得るまでに約 28 日かかるかと推定されるが、GA では $P = 50, G = 200$ で平均的に厳密解の 97 パーセント程度の解が 80 分程度で得られると推定される。

表 2: Centroid 問題に GA を適用したときの実行時間 (秒) (N=50)

世代数	P=10	P=50
10	0	1
50	0	16
100	1	61
150	3	136
200	6	242
250	9	375
300	12	542
350	16	740
400	22	960

なお、表 3 の数値が表 1 と異なるのは、乱数系列の違いによるものである。

表 3: Centroid 問題の解を得るまでの実行時間 (秒) (GA は 50 個体, 100 世代)

需要点数	GA	厳密解
10	49	0
50	61	0
100	76	0
150	90	11
200	104	36
250	118	179
300	132	739
350	146	1726
400	161	3040

Centroid 問題を GA で解いた場合の特徴

図 10 は 30 回の数値実験の平均を取ったものである。表面に出ていないが、個別の実験で出現する特徴的な結果を図 12 に示す。これら 3 本のグラフではいずれも、GA による解として厳密解の適応度 1 を上回る適応度の解が出ている世代がある。これは、非ゼロ和かつ Stackelberg 型の競合、という条件がもたらすものである。すなわち、後

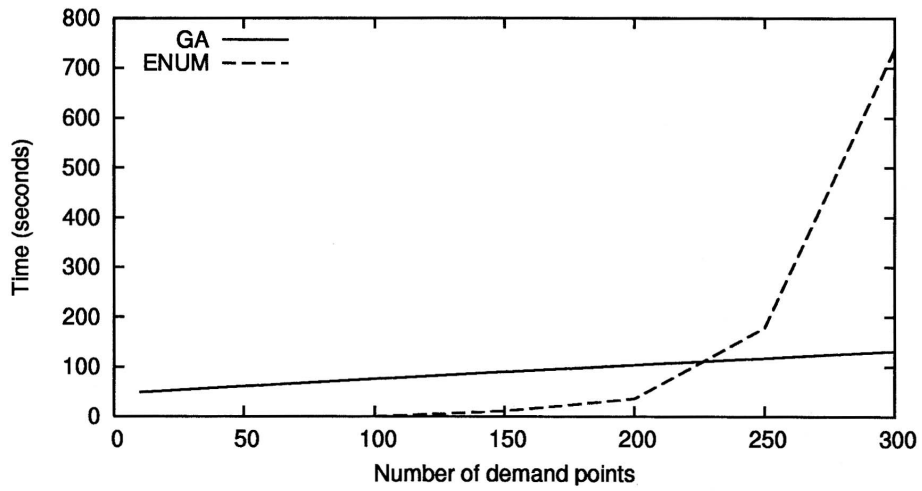


図 11: GA と列挙法 (ENUM) による実行時間の比較

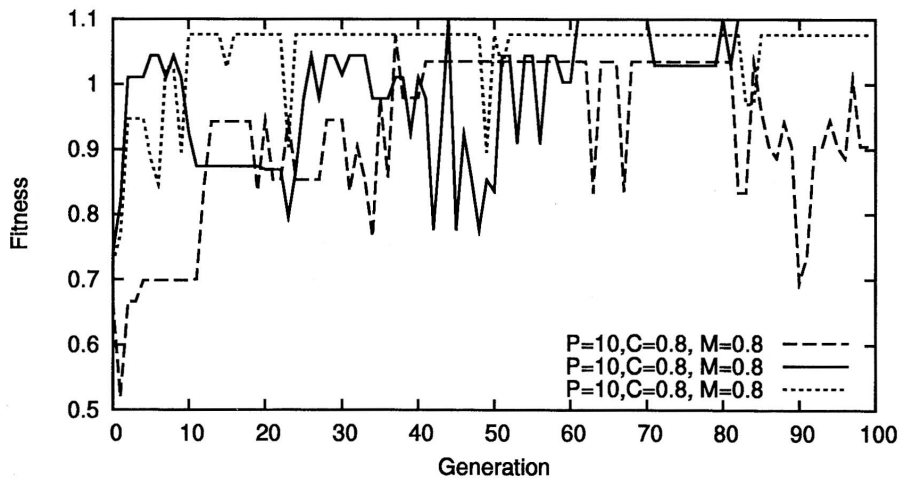


図 12: Centroid 問題で厳密解を超える解が出ている例

手が採るはずの配置戦略もまた GA を用いた近似解 \tilde{Y}^* であるので、 \tilde{Y}^* を仮定する先手は、いわば将来に対する読みが甘い、ということを意味し、後手が真の最適戦略である厳密解を採った場合には先手側が獲得不可能になるはずの利得をも取れるものと見積もって解を出すことになるのである。

ただし、図 12 の例は、個体数 $P = 10$ 、交叉率 $C = 0.8$ 、突然変異率 $M = 0.8$ 、世代数 $G = 100$ という極端なパラメータの組み合わせで出現しているものであり、同パラメータで 400 回の試行を行なったとき、厳密回を上回る解が一度でも出現したのはこれらの例を含めて 7 例であった。なお、 $P = 50$ 、 $C = 0.2$ 、 $M = 0.2$ 、 $G = 100$ というパラメータでは、1000 回の数値実験中で一度も出現することはなかった。

5 おわりに

本論文では、先手後手の区別のある Stackelberg 型の競合施設配置問題に GA を応用して解き、数値実験の結果を示した。先手の最適配置を求める Centroid 問題においては、需要点の個数が数千点のオーダーになると厳密解を求める時間が日数単位になる一方で、GA による近似解法では厳密解の 95 パーセント以上の解が実行時間内に得られることを示した。

先手後手の区別のある競合配置問題の場合、理論上は一般に先手の方が有利であるが、GA による近似解を得るまでの時間を考えると、時間制約が厳しい場合には、得られる近似解の質という点において後手の方が有利な状況もありえると思われる。

厳密解を得るための列挙法で用いた正方形の重なり列挙の前処理は、直角距離では利用圏の境界が 2 方向に制限されるという性質を利用している。従って、距離がユークリッド距離や大円距離である場合には、厳密解を得るにはまったく別の手法を用いねばならない。しかし、GA を用いる場合は、新たな手法を考えることなく、ただ距離関数を差し替えるだけで近似解を得ることができる。また、需要が距離の増加とともに逓減するモデルや施設の位置により異なる設置コストを考慮するモデルにも、利得関数の差し替えのみで対応できる。こうしたモデルの厳密解を求めることは容易ではないため、近似解で十分な場合には大幅な労力の節約となる。

今後の課題としては、交叉や突然変異の変動がもたらす結果の統計的な解析、および遺伝子のコーディング方法と選択方法の改良が挙げられる。

なお、本研究は、神戸学院大学 2009 年度研究助成 C を得たものである。

参考文献

- [1] H.Hotelling, 1929, “Stability in Competition”, The Economic Journal, Vol.30 , pp. 41-57.
- [2] S.L.Hakimi, 1983, “On Locating New Facilities in a Competitive Environment”, European Journal of Operational Research, Vol.12, pp. 29-35.
- [3] Haldun Aytug and Cem Saydamb, 2002, “Solving Large-scale Maximum Expected Covering Location Problems by Genetic Algorithms: A Comparative Study”, European Journal of Operational Research, Volume 141, Issue 3, pp.480-494.
- [4] Zorica Stanimirovic, Jozef Kratica and Djordje Dugosija, 2006, “Genetic Algorithms for Solving the Discrete Ordered Median Problem”, European Journal of Operational Research, Volume 182, pp.083-1001.
- [5] M.Bischoff and K.klamroth, 2006, “An Efficient Solution Method for Weber Problems with Barriers Based on Genetic Algorithms”, European Journal of Operational Research, Volume 177, pp.22-41.
- [6] Lili Yang, Bryan F. Jones and Shuang-Hua Yang, 2007, “A Fuzzy Multi-objective Programming for Optimization of Fire Station Locations through Genetic Algorithms”, European Journal of Operational Research, pp.903-915.
- [7] Hiroshi Imai, 1982, ”Finding Connected Components of An Intersection Graph of Squares in The Euclidean Plane”, Information Processing Letters, Vol.15, p. 125-128.
- [8] 大角盛広, 塩出省吾, 石井博昭, 寺岡義伸, 1997, 「施設の競合配置問題」, 日本生産管理学会論文誌 Vo.4, No.2, pp.153-160.
- [9] 長尾智晴, 「遺伝的アルゴリズムによる数値最適化のための均質コーディング」, 1997, 電子情報通信学会論文誌, Vol.J80-DII, No.1, pp.56-62