数理計画問題 —メタヒューリスティクス解法の基礎—

Fundamental Method of Meta-Heuristics in Mathematical Programming/Masashi FURUKAWA

北海道情報大学 古川正志

1. はじめに

設計、制御、生産管理等では、最適化を行う必要がある多くの場面に遭遇する。こうした最適化の問題は、数理計画問題 (Mathematical Programming, MP) と呼ばれ、古くはオペレーションズ・リサーチを中心に発展してきたが、今は人工知能をはじめとして、広い分野で研究が進んでいる。

MP は、数理的なアプローチが中心であるが、非線形や離散的な問題では計算の量が膨大になるため、より効率的な方法を求めるようになり、ヒューリスティクス (発見的方法, Heuristics) が発展してきた。特に、自然界の最適化を模倣したヒューリスティクスは、特定の分野のみならず広い分野で適用可能なために、メタヒューリスティクス (Meta-Heuristics, MH) と呼ばれる。よく知られたメタヒューリスティクス¹⁾には、遺伝的アルゴリズム (Genetic Algorithm, GA)、粒子群最適化法 (Particle Swarm Optimization, PSO)、シミュレーテッドアニーリング (Simulated Annealing)、タブーサーチ (Tabu Search, TS)、アントコロニー最適化法 (Ant Colony Optimization, ACO)、シミュレーテッド進化法 (Simulated Evolution, SimE) 等が挙げられる。ここでは、GA を中心に扱うが、機会があれば、他の方法についても述べたい。

2. 最適化問題の表現と留意点

2.1 最適化問題の定式化表現

どんな最適化問題もそれを解くためには、必ず定式化を行う必要がある。定式化を行うには、解の形を最初に約束する必要がある。今、問題の解を x とする。 x は設計変数やパラメータの集まりのベクトルや特定の順序を表現する。ここで、解 x によって定まる評価関数 $f(x)$ を設定する。評価関数は設計の良さやパラメータによって決まるシステムの良さを表現している。評価関数が数式で書けないときは、変数 x を定め、シミュレーションにより求めるのも可能である。多くの問題は、 $f(x)$ を最大化か最小化する解 x を求める問題になる。次に変数 x の制約条件式を設定する。制約条件とは、変数の範囲や制約変数間の関係性を意味し、複数あるのが普通である。したがって、制約条件式は $g_i(x)$ ($i=1, 2, \dots, m$) と添字を付けて表現する。シミュレーションで評価関数を定める場合は、シミュレーション

の中に含めることもできる。設計変数 x 、評価関数 $f(x)$ 、制約条件式 $g_i(x) \geq 0$ ($i=1, 2, \dots, m$) が定まると、最適化問題は、次のように書ける。

最小化 (x について) $f(x)$

制約条件 $g_i(x) \geq 0$ ($i=1, 2, \dots, m$)

一般には、これを

$$\min_x f(x)$$

subject to

$$g_i(x) \geq 0 (i=1, 2, \dots, m) \quad (1)$$

のように書く。min は minimize の略である。最大化の場合は、maximize の略である max を用いる。

2.2 制約条件の取り扱い

MH の方法の多くは、制約条件を取り込んだ最適化の定式化を用いる。問題 (1) は、以下のように簡単に制約条件のない問題に変換する。

$$\min_x f(x) + \sum_i P_i \quad (2)$$

ただし、

$$P_i = \begin{cases} 0 & : g_i(x) \geq 0 \\ \text{a large value} & : \text{otherwise} \end{cases}$$

である。 P_i は、制約条件を満たせば 0、そうでなければ大きな値とする。MP の勾配法に基づいた方法は微分を必要としたために、制約の扱いに気を使うことが多いが、ここでは制約があっても簡単に問題 (2) の形式で解くことにする。解 x が整数か実数かは MP では大きな問題となるが、MH では整数の方が簡単な場合もあり、あまり考えないこととする。

2.3 単峰性か多峰性

北海道の中心には大雪山と呼ばれる山がある。大雪山は実は連峰で、多くのピークをもつ山々 (多峰性) から成立している。どの山に登るかは、登山口による。富士山は単峰 (性) である。これと同じように評価関数の景観は、その多くは多峰性であり、一番高い峰の頂上を見つけるのが大きな問題となる。数学ではこれらの峰を極値という。一番高い峰の頂上は大域的最適値、それ以外の峰の頂上は局所的最適値と呼ばれる (図 1)。伝統的な MP と MH の異なる点は、多峰性の中で最も高い峰を見つける工夫がなされていることである。MH でも局所的最適値に陥ることがあり、問題によっては気をつける必要がある。

L：局所解 G：大域解

図1 大域的最大値と局所的最大値

2.4 計算量

厳密ではないが、ある問題が n 個の変数で定まり、与えられたときに、以下のものであるとする。

- (1) 解を与えると、その解が制約条件を満たしているかどうかを簡単に調べられる
- (2) 解を一つ一つ単純に調べていくと、その計算時間(量)が指数関数的に増加する

このような問題を困難な問題 (Hard Problem, HP) あるいは手に負えない問題 (Intractable Problem) と呼ぶ。計算量は、 n が増加すると C^n に比例して増加する。 C は、ある定数である。これに対して、計算時間が n^c に比例して増加する問題もある。後者の問題の集まりは計算量が多項式オーダーなので P クラス、前者の問題の集まりは NP クラスと呼ばれる。 P と NP は Polynomial Time および Nondeterministic Polynomial time の略からきている。 NP クラスと同等かそれ以上に難しい問題の集まりを、 NP 困難 (NP -hard) なクラスという。MH は NP または NP 困難なクラスに属する問題に対して、可能な時間で有効 (実行可能) 解を与える点で優越性を備えている。

3. 遺伝的アルゴリズム

3.1 生物の遺伝子表現

生物の進化の過程をアルゴリズムに実現した MH は、人工知能の分野で発展し、John Holland と David B. Fogel が先駆者である。その後、これらの多くを改良した方法が進展し、一連のこうしたアルゴリズムを進化計算群と呼んでいる。ここでは、Holland が始めた最も単純な遺伝的アルゴリズム (Genetic Algorithm, GA) について述べる。

生物の細胞には染色体があり、染色体はタンパク質に巻き付いた DNA でできている。この DNA は 20 種類のアミノ酸の並びから構成され、アミノ酸を作るのがヌクレオチドである。ヌクレオチドは、A, G, C, T と 4 種類があり、アミノ酸は、4 種類の中から 3 種類が並んで作られている。実際はヌクレオチドの並びは、 $4 \times 4 \times 4 = 64$ 種類ある。このアミノ酸の並びがタンパク質を、ひいては生物の個体を表現する。

コンピュータは 2 値表現を取り扱うので、上記の関係を模倣すると、アミノ酸は 6 ビットで表現できる。

DNA のアミノ酸の並びを遺伝子表現、アミノ酸からタンパク質を経て作られていく生物の個体を形質表現とこ

では呼ぶ。遺伝子表現から形質表現には、遺伝子を翻訳する機能が必要で、自然界ではこの翻訳が行われていると考えられている。

3.2 環境への適応と進化

各個体は個体群を生成する。この個体は、環境に適応しながら子孫を残していく。子孫が変わることを世代交代という。世代交代は、個体群の中で最も環境に適応した個体が、最も多く子孫を残していく。しかしながら、祖先の環境適応力が継続して受け継がれるとすると、環境の適応には限界が生じる。また、環境の変動によっては生き残れなくなる。このため、自然界では突然変異を起こすことにより、適応力の多様性を補う。この突然変異は、個体のプリミティブな部分である A, G, C, T の変化であると考えられている。これをビット表現であれば 0 から 1, あるいは 1 から 0 へとする。また、有性動物では 2 個体間の染色体が減数分裂で組み合わせられるから、このとき、遺伝子乗り換え (交差) が起こる可能性がある。これも遺伝子の突然変異である。

一般に、突然変異を起こした個体は、環境の適応力が弱い、時々既存の遺伝子をもつ個体より適応力が強い優勢な個体が現れる。この優勢な個体は環境への適応度が大きいので、子孫を多く残し、他の個体を淘汰し、個体の進化が起きる。

3.3 コンピュータアルゴリズム

上記の過程を以下のようにアルゴリズムにする。(1) n 個の初期個体を遺伝子で作成する (遺伝子型表現)、(2) 遺伝子を形質表現に翻訳する、(3) 翻訳された形質 (機能) を使用して、環境適応度を計算する。適応度を計算する関数を作成する場合に、これを適応関数という、(4) 得られた環境適応度に応じて、同じ遺伝子をもつ子孫の生成 (再生) を行い、 n 個の個体数になるように淘汰を行う、(5) 指定した割合で、個体をランダムに選択し、遺伝子型表現で突然変異を実施し、元に戻す、(6) 指定した割合でランダムに対の個体を選択し、二つの遺伝子を交差させ、新しい対の個体を元に戻す、(7) 指定した世代交代数になれば終了し、そうでなければ (2) へ戻る。

4. 生産計画問題への適用

4.1 生産計画問題

ここでは簡単な生産計画問題¹⁾を取り扱う。ある工場では 2 台の機械 A, B で 2 種類の製品 1, 2 を生産する。各機械での製品 1 個当たりの製造時間と利用可能時間は表 1 で与えられるとする。製品 1, 2 の 1 個当たりの利益は 60 千円と 50 千円とする。利益を最大にするような各製品の製造個数 x_1 と x_2 を求める。

この問題は、以下のように表現できる。

$$\begin{aligned} \max_x f(x) &= 60x_1 + 50x_2 \\ \text{subject to} \\ 2x_1 + 4x_2 &\leq 80 \end{aligned}$$

表 1 各機械の加工時間と利用可能時間

製 品	機械 A	機械 B
1	2 時間	3 時間
2	4 時間	2 時間
利用可能時間	80 時間	60 時間

表 2 計画問題の初期値の例（未修整の適応値）

個体	遺伝子型表現		形質型表現		適応値
i	G_1	G_2	x_1	x_2	$f(x)$
0	01000101	10110110	10.823	28.549	-17923.137
1	10111110	00000111	29.803	1.0980	-8156.862
2	01110101	11010100	18.352	33.254	-17236.078
3	00010101	11010000	3.294	32.627	-18170.980
4	10100010	00011111	25.412	4.863	-8232.157
5	01110000	00110111	17.569	8.627	-8514.509
6	01001110	10000111	12.235	21.176	-18207.058
7	11111111	01110010	40.0	17.882	-16705.882
8	01001011	01010110	1.765	13.490	-8619.608
9	10111100	00000111	29.490	1.098	-8175.686

$$3x_1 + 2x_2 \leq 60$$

$$x_1 \geq 0, x_2 \geq 0 \quad (3)$$

x_1 と x_2 は製品 1 と 2 の製造個数を表す。これらは、必ず 0 以上の大きさになる。 $f(x)$ は、利益を表し、制約条件は機械 A と B の利用可能時間の制約を表す。

4.2 遺伝的アルゴリズムの適用（実数）

以下のようにアルゴリズムを適用する。

適応関数の設計 問題を制約条件を取り込んだ問題に変換すれば、

$$\max_x f(x) = 60x_1 + 50x_2 + P_1 + P_2 \quad (4)$$

となる。ただし、

$$P_1 = \begin{cases} 0: 2x_1 + 4x_2 \leq 80 \\ -10^4: \text{otherwise} \end{cases}, P_2 = \begin{cases} 0: 3x_1 + 2x_2 \leq 60 \\ -10^4: \text{otherwise} \end{cases}$$

である。この問題では、最大化の問題を取り扱うので、制約条件を満たさないときの P_1 と P_2 の値は、十分に小さな値とする。これが環境への適応度を計算する適応関数となる。

遺伝子の設計 個体の総数を n とする。変数 x_1 と x_2 に対応する個体 i の遺伝子を G_{i1} と G_{i2} とし、それぞれ 8 ビットで 2 値（0 と 1）表現する。また、個体 i の遺伝子は、これらを結合して $G_i = G_{i1} \| G_{i2}$ と表現する。

翻訳器と形質表現 形質表現を変数 x_1 と x_2 とする。 G_1 と G_2 は、0 と 1 の記号列であり、この記号列をそのまま 2 進数とみなす。2 進数に変換したものを B_1 、 B_2 とする。変数 x_1 と x_2 の範囲は、それぞれ $[0, 20]$ 、 $[0, 20]$ であることは制約条件から分かるが、ここでは大きめにとってそれぞれ $[0, 40]$ とする。 B_1 と B_2 の範囲は 8 ビットで考えると $[0, 255]$ となるから、遺伝子から形質への翻訳は、 $x_1 = 40B_1/255$ および $x_2 = 40B_2/255$ で実行する。

初期化 個体 i の遺伝子 G_i は、乱数を使用して G_{i1} と G_{i2} に 8 ビット発生し、これらを結合して作成する。これを n

表 3 最終的に得られた解

個体	遺伝子型表現		形質型表現		適応値
i	G_1	G_2	x_1	x_2	$f(x)$
0	00111111	01011111	9.882	14.902	1338.039
1	00111111	01011111	9.882	14.902	1338.039
2	00111111	01011111	9.882	14.902	1338.039
3	00111111	01011111	9.882	14.902	1338.039
4	00111111	01011111	9.882	14.902	1338.039
5	00111111	01011111	9.882	14.902	1338.039
6	00111111	01011111	9.882	14.902	1338.039
7	00111111	01011111	9.882	14.902	1338.039
8	00101111	01010110	7.372	14.902	1187.450
9	00101111	01011111	9.882	14.902	1338.039

個体行う。

適応関数の計算 適応関数の計算は式 (4) で行う。直接、式 (4) を用いると、個体の子数の計算が面倒なので、適応関数を次のように修正する。

$$f(x) = 60x_1 + 50x_2 + P_1 + P_2 - f_{\min} \quad (5)$$

ここで、 $f_{\min}$ は計算した適応関数の最小のものである。最も適応度の悪い個体の関数値は 0 になる。

淘汰 子の個体数を計算する。子の個体数の計算方法はいくつかあるが、ここでは適応度に比例して子を生成する。子 i の個体数の計算には以下の計算式を用いる。

$$n_i = \frac{f_i}{\sum_i f_i} n + 1 \quad (6)$$

n_i は子 i の個体数、 f_i はその修正した適応関数値である。1 を加えたのは、どの個体も必ず子を 1 個体もてるようにするためである。

次に子の個体数を、総数を n 個に限定して生成する。これは、適応関数値を大きい順にソーティングし、その順番に子の個体を個体数に基づいて作成していく。個体数の総和が n 個になると子の生成を終了する。この生成により、適応度の小さな個体は除去されていく。

突然変異 個体数に指定した割合でランダムに個体を選択する。次いで、その個体の遺伝子の位置をランダムに決定し、1 ならば 0 に、0 ならば 1 にして元の個体群に戻す。

交差 個体数に指定した割合で対となる 2 つの個体をランダムに選択する。次いで個体の遺伝子の位置をランダムに決定し、その位置以降の遺伝子を交換して新しい個体を 2 つ作り、これらを元の個体群に戻す。その後、指定世代数であれば終了、そうでなければ適応関数の計算と淘汰に戻る。

計算例 初期解を表 2、得られた解を表 3 に示す。個体数 $n=10$ 、突然変異率 10%、交差率 20% で計算している。ほぼ 2000 世代で、正解を得る。この問題では $x_1=10$ 、 $x_2=15$ が正解となる。うまくいかないときは、個体数を 30 くらいにすればよい。200 世代くらいで正解を得る。

4.3 遺伝的アルゴリズムの適用（整数）

実は 4.2 で述べた方法は、実数問題を取り扱う場合について述べている。この計画問題では、変数 x_1 と x_2 が整数

でないという意味をもたない。このような問題は、従来は整数計画問題と呼ばれ、解くのが非常に難しいものであった。しかし、GA では解の表現を整数と限定すればよいので、簡単になる。上記の遺伝子型表現と形質型表現を以下のように変更する。

遺伝子型表現と形質型表現 変数 x_1 と x_2 を $[0, 40]$ に限定することは先にも述べた。整数のみを取り扱うのであれば、これを2進数で直接表現すればよいことが分かる。変数の最大値が40であり、2進数で表すビット数は6ビットで十分である。変数 x_1 と x_2 を6ビットで表現したものを遺伝子型表現とする。

変数 x_1 と x_2 の遺伝子を2進数に翻訳した値を、先ほどと同じく B_1 , B_2 とする。このとき、解 x_1 と x_2 は、単に $x_1 = B_1$ および $x_2 = B_2$ と置く。これが形質型表現となる。

後の手続きは、前とまったく同じである。 $x_1 = 10$, $x_2 = 15$ 、適応値1350が整数値として得られる。

5. 巡回セールスマン問題への適用

5.1 巡回セールスマン問題

巡回セールスマン問題 (Travelling Salesman Problem, TSP) とは、セールスマンが n 都市をただ一度のみ訪問し、最短距離 (最小時間) で出発都市に戻る問題である。簡単にいうと一筆書きの最短距離経路を求める問題である。応用としては、ドリルやワイヤーボンディングの最小加工順序を求めたり、物流トラックの最短配達経路を決定する問題がある。これは以下のように定式化する。

$$\min_x f(x) = \sum_{i=0}^n d_{i,i+1} + d_{n0} \quad (7)$$

変数 x は、ここでは、都市を回る順序を示すものとする。例えば、 $x = (20143)$ のように設定する。数字は都市に付けられた番号とする。添字の i は、変数 x に現れる順番の値とする。 $i=3$ であれば都市1を示す (都市は0から番号付けする)。 $d_{i,i+1}$ は、変数 x に現れる i 番目の都市と $i+1$ 番目の都市の距離とする。この値は行列 $D = [d_{i,i+1}]$ に格納しておく。経路がない場合には、十分大きな値を設定する。

5.2 遺伝的アルゴリズムの適用

TSP では、解の表現を行う遺伝型表現に少し工夫が必要となる。形質型表現は都市の巡回順序にする。ここでは、単純な順序表現を用いた方法を説明する。

適応関数の設計 適応関数は、式(7)で計算する総距離を使用する。

遺伝子の設計と初期化 個体の総数を n とする。遺伝子の形質型表現は、都市を巡回する順序をリストに格納し、直接用いる。例えば $x = (20143)$ は都市を2-0-1-4-3-2の順に訪問するが、これを図2のようにアドレスを付けたリストに格納する。

遺伝型表現はこのリストの値を使用することができない。それはリストのある場所から二つの個体の遺伝子交差を行うと同じ都市が現れる可能性があるためである。遺伝

アドレス I	0	1	2	3	4
都市番号	2	0	1	4	3

図2 リストに訪問都市番号を与えた実順序リスト

アドレス I	0	1	2	3	4
都市番号	0	1	2	3	4

図3 初期作業リスト

アドレス I	0	1	2	3	4
都市番号	1	3	4		

アドレス G	0	1	2	3	4
アドレス I	2	0	0		

図4 3回目の作業リストと順序リスト (作業リストの反転の部分のアドレスが順序リストに格納される)

アドレス G	0	1	2	3	4
アドレス I	2	0	0	1	0

図5 最終順序リスト (20010) が遺伝子型表現となる

型表現の初期化は次のような方法を用いる。

- 1) 最初に0から n 番地をもつリストを作り、リストの個々のデータに番地と同じ値を都市番号として与える。これを作業リストと呼ぶ。
- 2) やはり、0から n 番地をもつリストを作り、データ欄は空白にしておく。これを順序リストと呼ぶ。
- 3) $i=0$ に設定する。
- 4) 0から $n-i$ までの乱数を生成し、その値 k を順序リストのアドレス i 番地のデータとして格納する。
- 5) 作業リストの k 番地のデータを消去し、それ以降のデータを k 番地からのデータになるようにシフトする。
- 6) $i=n$ であれば、終了する。そうでなければ、 $i=i+1$ として4)へ戻る。

例の都市順序は図4~5のように格納されていく。この最終の順序リスト (図5) が遺伝型表現となる。

翻訳器と形質型表現 順序リストの翻訳は以下のように行う。

- 1) 前回と同じ作業リストを作成する。
- 2) やはり、0から n 番地をもつリストを作り、データ欄は空白にしておく。これを実順序リストと呼ぶ。
- 3) $i=0$ に設定する。
- 4) 順序リストの i 番地のデータを作業リストのアドレスとみなし、そのデータの都市番号を実順序リストのデータに格納する。
- 5) 作業リストの i 番地のデータを消去し、それ以降のデータを i 番地からのデータになるようにシフトする。
- 6) $i=n$ であれば、終了する。そうでなければ、 $i=i+1$ として4)へ戻る。

でき上がった実順序リストのデータは、図2となり形質

図6 100都市 TSP のGA による初期解

図7 100都市 TSP のGA による収束解

型表現として、都市の訪問順序となる。

適応関数の計算と淘汰 適応関数は式(7)で行うが、淘汰は、今回は以下のように行う。すなわち、適応関数値の小さいものからソートし、上位 20% の個体を下位 20% の個体と入れ替える。20% は、淘汰圧と呼ばれ、自由に設定できる。

突然変異 個体数に指定した割合でランダムに個体を選択する。次いで、順序リストのアドレス i を乱数で決め、そのデータを 0 から i までの乱数で決定する。

交差 個体数に指定した割合で対となる 2 つの個体をランダムに選択する。次いで、これらの個体の順序遺伝子のアドレスをランダムに決定し、その位置以降の遺伝子を交換して新しい個体を 2 つ作る。これらを元の個体群に戻す。

逆位 個体数に指定した割合でランダムに個体を選択する。次いで、実順序リストのアドレス i と j を乱数で決め、アドレス i と j のデータ部分の都市順序を逆順に挿入する。その後、指定世代数であれば終了、そうでなければ

適応関数の計算と淘汰に戻る。

計算例 ランダムに生成した 100 都市の計算例の初期解と得られた解を図 6, 7 に示す。これは個体数 $n=10$ 、突然変異率 20%、交差率 30%、逆位率 40% で計算している。ほぼ 18000 世代で、準最適解を得ているのが分かる（多くの場合、最適解は不明）。

6. お わ り に

メタヒューリスティクスとは何かと、その中で代表的な方法として遺伝的アルゴリズムについて述べた。メタヒューリスティクスには、ほかにも PSO や TS のような有用な方法がある。機会があればこれらについて紹介したい。

参 考 文 献

- 1) 古川正志ほか：メタヒューリスティクスとナチュラコンピュ—ティング，コロナ社，東京，(2012)。