遺伝アルゴリズムによる作業割当問題の解法*

当麻 淳治**・三宮 信夫**・川合 成治***

Genetic Algorithm Approach to an Operation Assignment Problem*

Junji TOUMA**, Nobuo SANNOMIYA** and Seiji KAWAI***

1. 緒言

限られた設備を用いて多くの作業を行う場合に、各作業を望ましい時間帯にいかに関与するかを考える必要がある。そのような問題に対して、通常設備と日程間で作業の山積み表を作成し、そのピークをくずす方法がとられている¹⁾。しかし、その場合実行可能解を得ることは容易であるが、最適な山くずし法を与えることは難しい。そこで論文では、並列機械に対する作業割り当て問題に遺伝アルゴリズム (GA)²⁾を適用して、その解法を考察する。

2. 問題の記述

N 種類の製品をある 1 種類の機械で処理するものとする。機械は並列に H 台あり、製品 i ($i=1, 2, \dots, N$) の機械での処理時間 c_i 、最早処理開始時刻 a_i 、最遅処理終了時刻 b_i が与えられており、できれば区間 (a_i, b_i) で処理したいとする。ただし、 $H < N$ および $b_i - a_i > c_i$ とし、時刻は離散量で与えられるものとする。このとき、この制約条件を満たすような製品の処理順序があれば求めよ。もしなければ、区間 (a_i, b_i) 外での処理時間が最も少なくなるような製品の処理順序を求めよ。

ここで、本問題の特徴を表すパラメータとして、全体の作業の余裕度 F および集中度 P を、それぞれ次式で定義する。

$$F = \frac{\sum_{i=1}^N c_i}{H (\max_i b_i - \min_i a_i)} \quad (1)$$

$$P = \frac{\max_{k_1 \leq k \leq k_2} z(k)}{H} \quad (2)$$

ただし、 k は離散時刻を表わし、

$$z(k) = \sum_{i=1}^N t_i(k) \quad (3)$$

$$t_i(k) = \begin{cases} 1, & a_i \leq k \leq a_i + c_i \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

$$k_1 = \min_i a_i \quad (5)$$

$$k_2 = \max_i (a_i + c_i) \quad (6)$$

3. GA の適用

この問題に GA を適用して準最適解を求める。GA で用いられる用語にしたがって、解を個体、解集合を個体群、繰り返し回数を世代数と呼ぶことにする。

3.1 アルゴリズムの概要

GA の手順は次のステップからなり、世代数が t^* になると終了する。なお、個体数は世代を通じて一定である。

Step 1 M 個の個体をランダムに生成して、初期個体群を設定する。世代を $t \leftarrow 1$ とする。

Step 2 各個体の適応度を計算して個体の淘汰を行う。

Step 3 M 個の個体をランダムに 2 個ずつ組み合わせ、確率 P_c で交叉を行って新しい個体を生成し、その生成に関与した古い個体と置き換える。

Step 4 M 個の個体の各々について、確率 P_m で突然変異を行って、得られた個体を古い個体と置き換える。

Step 5 $t = t^*$ ならば、最終世代における適応度が最大の個体を問題の準最適解として、計算終了。そうでないならば、 $t \leftarrow t+1$ として Step 2 へ戻る。

3.2 目的関数値の計算

GA における個体は、製品番号の列 $B_1, B_2, \dots, B_N$ によって表わす。個体からその目的関数値を計算するた

* 原稿受付 1994 年 5 月 18 日

** 京都工芸繊維大学 工学部 Faculty of Engineering and Design, Kyoto Institute of Technology; Goshokaidocho, Matsugasaki, Sakyo ward, Kyoto 606, JAPAN

*** 富士電機総合研究所 Fuji Electric Corporate Research and Development Ltd.; Fujimachi 1, Hino city, Tokyo 191, JAPAN

Key Words: genetic algorithm, operation assignment problem, parallel machines, relaxation of concentrated operations, scheduling.

めに、つぎの3種類の方法を考える。ここに、製品 i の実際の処理開始時刻および処理終了時刻を、それぞれ s_i および e_i とする。

【方法Ⅰ】

個体表現の製品番号順に、製品をその最早開始時刻よりも後で最も早く空く機械に割り当てる。そのアルゴリズムは、 $t(m, k_m)$ を機械 m の k_m 番目の処理の終了時刻とすれば、次のようになる。

- Step 1 $j \leftarrow 1, m = 1, 2, \dots, H$ について $k_m \leftarrow 0$, $t(m, 0) \leftarrow 0$ とする。
- Step 2 $m^0 = \arg \min_{1 \leq m \leq H} t(m, k_m)$ を求める。
- Step 3 $i \leftarrow B_j, j \leftarrow j+1$ とする。
- Step 4 $s_i \leftarrow \max(t(m^0, k_{m^0}), a_i)$, $e_i \leftarrow s_i + c_i, t(m^0, k_{m^0}+1) \leftarrow e_i$ とする。
- Step 5 $k_{m^0} \leftarrow k_{m^0}+1$ とする。 $j \leq N$ ならば Step 2 に戻る。そうでないならば終了。

【方法Ⅱ】

個体表現の製品番号順に製品を、一定の順序に定められた機械に割り当てる。このとき、それまでに割り当てられた製品の処理時間と重なる場合には、 $(s_i + e_i)/2$ の小さい順になるように、それまでに割り当てられた製品の処理時間を前後にずらして重ならないようにする。そのアルゴリズムは、 $x(m, k_m)$ および $r(m, k_m)$ をそれぞれ機械 m の k_m 番目に処理する製品の番号および処理の中間点とすれば、つぎのようになる。

- Step 1 $j \leftarrow 1, m = 1, 2, \dots, H$ について $k_m \leftarrow 0$ 。
- Step 2 $i \leftarrow B_j, s_i \leftarrow (b_i + a_i - c_i)/2, e_i \leftarrow s_i + c_i$ 。
- Step 3 B_j を機械 $m = ((j-1) \bmod H) + 1$ に割り当てる。
- Step 4 $l = 1, 2, \dots, k_m$ について $i_0 \leftarrow x(m, l)$, $r(m, l) \leftarrow (s_{i_0} + e_{i_0})/2$ 。
また、 $r(m, 0) \leftarrow -\infty, r(m, k_m+1) \leftarrow +\infty$ 。
- Step 5 $r(m, w-1) \leq (s_i + e_i)/2 \leq r(m, w)$ が成り立つ w をさがす。
- Step 6 $l = k_m, k_m-1, \dots, w$ について $x(m, l+1) \leftarrow x(m, l), x(m, w) \leftarrow i$ 。
- Step 7 $l = w+1, i_0 \leftarrow x(m, l), i_1 \leftarrow x(m, l-1)$ 。
- Step 8 $e_{i_1} > s_{i_0}$ が成り立つ間 $s_{i_0} \leftarrow e_{i_1}, e_{i_0} \leftarrow s_{i_0} + c_{i_0}$, $l \leftarrow l+1, i_0 \leftarrow x(m, l), i_1 \leftarrow x(m, l-1)$ を実行する。 $l \geq k_m+1$ ならば Step 9 に進む。
- Step 9 $l = w-1, i_0 \leftarrow x(m, l), i_1 \leftarrow x(m, l+1)$ 。
- Step 10 $s_{i_1} < e_{i_0}$ が成り立つ間 $e_{i_0} \leftarrow s_{i_1}, s_{i_0} \leftarrow e_{i_0} - c_{i_0}$, $l \leftarrow l-1, i_0 \leftarrow x(m, l), i_1 \leftarrow x(m, l+1)$ を実行する。 $l \leq 0$ ならば Step 11 に進む。

- Step 11 $k_m \leftarrow k_m+1, j \leftarrow j+1$ 。 $j \leq N$ ならば Step 2 に戻る。そうでないならば終了。

【方法Ⅲ】

方法Ⅱを以下のように変更する。すなわち、機械の割り当ての際、それぞれの機械ごとに、 B_j を割り当てることによって生ずる処理時間のずれを計算して、それが最も少ない機械に割り当てるようにする。そのアルゴリズムは、方法Ⅱの Step 3 をつぎのように書き換える。

- Step 3 機械 $m = 1, 2, \dots, H$ のそれぞれについて、 $y(m) \leftarrow 0$ とした上で方法Ⅱの Step 4~11 を実行する。このとき、Step 8 での $s_{i_0} \leftarrow e_{i_1}$ および Step 10 での $e_{i_0} \leftarrow s_{i_1}$ の実行の前に、それぞれ $y(m) \leftarrow y(m) + e_{i_1} - s_{i_0}$ および $y(m) \leftarrow y(m) + e_{i_0} - s_{i_1}$ を実行する。そして、 $y(m)$ が最も少ない機械 m^0 に割り当てる。 $m \neq m^0$ なる m については、その機械に割り当てられているすべての製品 l について s_l, e_l を元に戻す。

以上の方法により s_i および $e_i (i = 1, 2, \dots, N)$ を定めると、目的関数値は次式で与えられる。

$$Z = \sum_{i=1}^N (\max(e_i - b_i, 0) + \max(a_i - s_i, 0)) \quad (7)$$

ただし、方法Ⅰでは処理を前はずらさないで、(7)式の第2項は必ず0である。

3.3 遺伝演算子

淘汰規則は、各個体の適応度から計算される次世代での個体数の期待値を整数化して、確定的に個体群を生成する方法を用いる。各世代における各個体 $p_j (j = 1, 2, \dots, M)$ の適応度関数を次式により定める³⁾。

$$f_j = U \min_{1 \leq q \leq M} Z_q + \max_{1 \leq q \leq M} Z_q - Z_j \quad (8)$$

ここに Z_j は個体 p_j の目的関数値である。また、 U は淘汰速度を調節するための淘汰圧パラメータである。

本問題の性質上、個体表現の製品番号列において、各遺伝子の前後関係よりも各遺伝子座を保存するような交叉規則が望ましい。実際、参考文献4)では cycle crossover を 1-point および 2-point order crossover と比較して、その有効性が確認された。よって、交叉規則としては、cycle crossover を用いる。

また、突然変異規則も遺伝子の前後関係を保存したままでシフトするものよりも、ある特定の遺伝子座の遺伝子のみ置き換わるものが望ましい。よって突然変異規則は、個体上でランダムに選んだ二つの遺伝子を交換する方法を用いる。

4. 計算結果

まず、 $(N, H) = (8, 2)$ の場合はサイズが小さいの

Table 1 Computational results of average value of $\min Z$ for various values of N and H

(N, M)	$\bar{F}$	$\bar{P}$	t^*	GA			Random search		
				Method I	Method II	Method III	Method I	Method II	Method III
(16, 3)	0.872	4.267	120	1.2	3.2	3.3	8.5	6.9	5.5
(24, 4)	0.878	3.950	250	2.3	5.8	7.0	39.6	15.9	12.4
(50, 3)	0.923	5.800	800	221.8	129.4	134.7	867.8	210.8	192.6
(70, 3)	0.933	6.333	1200	571.6	279.7	299.1	2039.1	481.1	436.2
(100, 3)	0.890	6.400	2000	1857.7	790.7	844.5	5645.1	1312.8	1226.7

Notes: $\bar{F}$ and $\bar{P}$ are the average values of F and P for five numerical examples, respectively.

で、列挙法により最適解が求められる。そこで、最適解を参考にして GA のパラメータのチューニングを行った。その結果、パラメータとして個体数 $M = 100$ 、淘汰圧 $U = 0.01$ 、交叉確率 $P_c = 1.0$ および突然変異確率 $P_m = 0.001$ とした。また、世代数 t^* は、問題のサイズに応じて定めることとした。

つぎに、5 種類の (N, H) の組み合わせについて、3 種類の目的関数計算法を用いて計算した結果を Table 1 に示す。 (N, H) のそれぞれの組み合わせについて、 $F < 1$ でかつ $P > 1$ となるような 5 例ずつの事例をランダムに作成し、各事例を 2 回ずつ計算した。同表の数値は、この 10 回の計算で得られた準最適目的関数値の平均値を表わす。また、同表には GA と同じ探索回数 $Mt^* = 100t^*$ によって得られたランダム法の結果も併記した。

これより、 $N \leq 24$ の問題に対しては方法 I が最も良く、 $N \geq 50$ の問題に対しては、方法 II が最も良いことがわかる。この理由は次のように考えられる。すなわち、同表より、 N の小さい事例では $\bar{F}$ および $\bar{P}$ の値が小さく、山くずしの平滑化に余裕があり、山積みのピーク値も小さい。その場合には、方法 II および方法 III の複雑な手続きをするまでもなく、方法 I のように単純にピークを後ろにずらせばよい。

これに対して、 N の大きい事例では $\bar{F}$ および $\bar{P}$ の値が大きくなり、平滑化の余裕度が小さく、山積みのピーク値も大きい。その場合は方法 I の単純な操作には限界があり、ピークを前後に振り分けることのできる方法 II の有効性が示されている。ただし、方法 III が方法 II に劣っているのは、方法 III で用いる local search が不完全であることにほかならない。この問題では、簡単で有効な local search を見い出せなかった。

また、GA とランダム法を比較すると、すべての場合に GA の探索効率が悪れており、問題の規模が大きくなるにつれてその傾向が顕著になることがわかった。

5. 結 言

作業割り当て問題において、GA を適用してその解法を考察した。山くずしとして三つの方法を提案したが、問題の余裕度および集中度の大きい困難な問題に対しては、方法 II が最も有効であることがわかった。この方法は、解法の手続きとしては方法 III より単純である。これより、GA の機能は GA パラメータのチューニングが適切であれば、local search のような手続きを必ずしも必要としない場合があることを示している。

参 考 文 献

- 1) 関根：PERT・CPM；日科技連，pp. 59～76 (1965)
- 2) D. E. Goldberg：Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley (1989)
- 3) 飯間，三宮：遺伝アルゴリズムを用いた変形フローショップ問題の解法；システム制御情報学会論文誌，Vol. 6, No. 10, pp. 437～445 (1993)
- 4) 当麻，三宮，川合：作業割り当て問題に対する遺伝アルゴリズムの適用；第 38 回システム制御情報学会研究発表講演会講演論文集，pp. 89～90 (1994)