遺伝アルゴリズムによる製品投入順序問題の解法[†]

飯 間 等*・三 宮 信 夫*

Genetic Algorithm Approach to a Production Ordering Problem

Hitoshi IIMA* and Nobuo SANNOMIYA*

In this paper, we consider a production ordering problem for an assembly process with several constraints. One constraint is that the capacity of the buffers at the input and output sides of the process is limited. The other constraint is that the cycle time of the worker is constant.

For obtaining a good suboptimal feasible solution, the genetic algorithm is applied to this problem. A procedure for carrying out the operations of reproduction, crossover and mutation is shown.

The effectiveness of the algorithm is compared from the viewpoints of accuracy and computation time among other methods, such as the heuristic method and the simulated annealing method. It is observed from numerical results that the genetic algorithm takes much computation time, but finds a better solution than other methods. Further, the genetic algorithm is found to be effective in spite of narrow feasible region.

Key Words: genetic algorithm, scheduling, FA, optimization

1. 緒 言

最近、最適化手法において遺伝アルゴリズム (Genetic Algorithm: GA)^{1)~3)} が注目を集めつつある。GA とは、生物の進化過程とそれを支えているとされる遺伝のメカニズムを応用した解法である。通常、最適化手法は、解空間の状態遷移を一つの系列のみで行うが、GA は解空間の状態遷移を複数の系列を用いて行うので、解空間を幅広く探索することができ、より良い解が得られることが期待できる。このように GA は最適化問題を解く場合に有効な手段となるわけであるが、GA を実用的な規模で具体的な問題に適用したという報告はまだ多くない。

本論文では生産比率の一定化を目標とした製品投入順序問題を取り扱う。多品種混合組立ラインにおいてどのような順序で製品を投入するかは重要な問題の一つである。生産の平準化を行うことによって、材料や部品を供

給している前工程の生産量の変動を小さくしたり、在庫の低減を図ることができ、コストを下げるができる⁴⁾。本論文では機械の入力側および出力側にバッファが設けられている多品種混合組立ラインに対するスケジューリング問題を考える。

GA を用いて上述の問題を解き、ほかの近似解法および最適解と比較することによって、得られた解の精度および計算時間を検討する。

2. 問題の記述

本論文では、**Fig. 1** に示される加工工程で、 N 種類の製品 A_i (または単に i) ($i=1, 2, \dots, N$) が加工される場合の製品の投入順序問題を考える。この工程には、機械

Fig. 1 Assembly process

[†] 第 34 回自動制御連合講演会で発表 (1991・11)

* 京都工芸繊維大学工学部 京都市左京区松ヶ崎御所海道町

* Faculty of Engineering and Design, Kyoto Institute of Technology, Kyoto

(Received October 25, 1991)

(Revised June 16, 1992)

の入力側および出力側に容量 B のバッファが設けられており、この工程での加工前および加工後の製品はこのバッファに置くことができる。製品は前工程→入力側バッファ→機械→出力側バッファ→次工程と流れていく。このうち、前工程→入力側バッファおよび出力側バッファ→次工程への運搬を作業者が行っている。作業者は前工程→入力側バッファ→出力側バッファ→次工程→前工程と移動し、1周する時間(サイクルタイム)は h_0 で一定である。また、製品 A_i の機械による生産時間を h_i 、生産量を Q_i とする。

この問題に対してつぎのようなバッファに関する制約が課されている。

1) 入力側あるいは出力側のバッファに容量 B より多い個数の製品を置かないこと

2) 入力側あるいは出力側のバッファが空になり、機械あるいは作業者の作業に待ち時間を生じさせないこと

このとき、各製品をできる限り一定の割合で生産するには製品の投入順序をどのように決定すれば良いかという問題を考える。組立加工ラインにおいてジャストインタイムの意味で最適化を行うには、部品の使用頻度が平滑化されるように製品を投入することが望ましい⁴⁾。本論文では、部品に関する条件を考慮していないが、それを考慮すると、目的関数が増え制約条件が新たに付加されることになる。しかしその場合でも、本論文の解法を適用することができる。

上に述べた問題の定式化を行う。ただし、問題の簡略化のために、作業者が入力側のバッファから出力側のバッファへ移動するのに時間は全くかからないものとする。また、作業開始時には、作業者は出力側のバッファの所にいるものとし、機械には一つの製品が、入力側のバッファには C_0 ($0 < C_0 \leq B$) 個の製品が置かれているものとする。

このとき、総生産量 Q および総生産時間 T は次式のようになる。

$$Q = \sum_{i=1}^N Q_i \quad (1)$$

$$T = \sum_{i=1}^N h_i Q_i \quad (2)$$

いま、 k 時点における変数をつぎのように定義する。

$x_i(k)$: 製品 A_i の生産累計の実績値。

$m_i(k) = Q_i k / T$: 製品 A_i の生産累計の目標値。

$u_i(k)$: 製品 A_i が機械で加工完了したとき 1、否のとき 0 となる変数。

$v(k)$: 作業者が機械に到達したとき 1、否のとき 0 となる変数。

$b(k)$: 出力側バッファ内の製品数。

$c(k)$: 入力側バッファ内の製品数。

生産比率の一定化を図るには、各時点で $x_i(k)$ ができる限り $m_i(k)$ の値に近接することが望ましい。そこで、本論文で考察する問題の目的関数をつぎのように与える。

$$\min Z = \sum_{i=1}^N \sum_{k=1}^T [x_i(k) - m_i(k)]^2 \quad (3)$$

製品の生産累計およびバッファ内の製品数は、次式で与えられる。

$$x_i(k) = x_i(k-1) + u_i(k) \quad (4)$$

$$b(k) = b(k-1) + \sum_{i=1}^N u_i(k) - v(k) \quad (5)$$

$$c(k) = c(k-1) - \sum_{i=1}^N u_i(k) + v(k) \quad (6)$$

$$x_i(0) = 0 \quad (7)$$

$$b(0) = 0 \quad (8)$$

$$c(0) = C_0 \quad (9)$$

$$x_i(k) \leq Q_i \quad (10)$$

$$x_i(T) = Q_i \quad (11)$$

$$0 \leq b(k) \leq B \quad (12)$$

$$0 \leq c(k) \leq B \quad (13)$$

$$i = 1, 2, \dots, N; k = 1, 2, \dots, T$$

s を製品の投入順序を表すパラメータとする。第 s 番目に投入される製品を A_{i_s} とすれば、投入順序列 Γ はつぎのように与えられる。

$$\Gamma = A_{i_1}, A_{i_2}, \dots, A_{i_s}, \dots, A_{i_Q} \quad (14)$$

ここに、添字 $i_1, i_2, \dots, i_Q$ は製品番号の集合 $\{1, 2, \dots, N\}$ から選ばれるものとする。

また、第 s 番目に生産される製品 A_{i_s} の生産完了時点は、次式で与えられる。

$$t(s) = h_{i_1} + h_{i_2} + \dots + h_{i_s} \quad s = 1, 2, \dots, Q \quad (15)$$

このとき、 $u_i(k)$ および $v(k)$ は次式で与えられる。

$$u_i(k) = \begin{cases} 1 & k = t(s), i = i_s \text{ のとき} \\ 0 & \text{その他のとき} \end{cases} \quad (16)$$

$$v(k) = \begin{cases} 1 & k = mh_0 \text{ のとき} \\ 0 & k \neq mh_0 \text{ のとき} \end{cases} \quad (17)$$

$$m = 1, 2, 3, \dots$$

まず、バッファに関する制約条件を導く。(5)式を $k=1$ から $k=t(s)$ まで加えると次式が成り立つ。

$$\sum_{k=1}^{t(s)} b(k) = \sum_{k=1}^{t(s)} b(k-1) + \sum_{k=1}^{t(s)} \sum_{i=1}^N u_i(k) - \sum_{k=1}^{t(s)} v(k) \quad (18)$$

(8), (16), (17)式より

$$b(t(s)) = s - [t(s)/h_0] \quad (19)$$

ここで、 $[\]$ はガウス記号である。

(i) $v(t(s)) = 0$ のとき

(5)式はつぎのようになる。

$$b(t(s)) = b(t(s-1)) + 1$$

$$\therefore b(t(s-1))=b(t(s))-1 \quad (20)$$

また、(12)式より

$$0 \leq b(t(s)) \leq B \quad (21)$$

$$0 \leq b(t(s-1)) \leq B \quad (22)$$

(22)式に(20)式を代入して次式が得られる。

$$1 \leq b(t(s)) \leq B+1 \quad (23)$$

(21)式と(23)式の共通部分をとって次式が成り立つ。

$$1 \leq b(t(s)) \leq B \quad (24)$$

(19)式を代入すると

$$1 \leq s - [t(s)/h_0] \leq B$$

$$\therefore (s-B)h_0 \leq t(s) < sh_0 \quad (25)$$

(ii) $v(t(s))=1$ のとき

(17)式より次式が成り立つ。

$$[t(s)/h_0]=t(s)/h_0 \quad (26)$$

したがって、(19)式はつぎのようになる。

$$b(t(s))=s-t(s)/h_0 \quad (27)$$

これを(12)式に代入して次式を得る。

$$\therefore (s-B)h_0 \leq t(s) \leq sh_0 \quad (28)$$

これより、(25)式と(28)式のいずれかが成り立てばよいので次式が得られる。

$$(s-B)h_0 \leq t(s) \leq sh_0 \quad (29)$$

$c(k)$ に関しても同様に、(6)式を $k=1$ から $k=t(s)$ まで加えて、(9)、(16)、(17)式を用いると次式が得られる。

$$c(t(s))=C_0-s+[t(s)/h_0] \quad (30)$$

(13)式に代入して

$$0 \leq C_0-s+[t(s)/h_0] \leq B$$

$$\therefore (s-C_0)h_0 \leq t(s) < (B+s-C_0+1)h_0 \quad (31)$$

(29)式と(31)式は同時に成立しなければならないから、次式のような制約条件式が得られる。

$$(s-C_0)h_0 \leq t(s) \leq sh_0 \quad s=1, 2, \dots, Q \quad (32)$$

(32)式において、右側の不等式は作業者が s 回目に入力側のパッファの前に来るときまでには第 s 番目の製品を生産しなければならないことを表わし、左側の不等式は第 s 番目の製品を $(s-C_0)h_0$ 時点以後に生産しないと入力側のパッファが空になって機械に待ち時間が生じてしまうことを表わす。

以上により、問題は、つぎのように定式化される。

$$\min Z = \sum_{i=1}^N \sum_{k=1}^T [x_i(k) - m_i(k)]^2 \quad (3)$$

subject to (4), (7), (10), (11), (16) and (32).

本問題は(32)式の制約のため、任意に選んだ投入順序が実行可能となる可能性はきわめて少なく(一例として、後述の Fig.4 参照)、実行可能解を得ることさえ難しくなっている。

3. 遺伝アルゴリズム(GA)の適用

生物は進化によって生存競争にうまく適合した形態や行動を獲得してきたと考えられているが、GA とは、このような進化とそれを支えているとされる遺伝のメカニズムを応用することにより、最適化問題を解こうとするものである。GA は一種のランダム探索法であり、局所解におちいることなく大域的な最適解を目指している。以下に本問題を解くための GA の構成方法を述べる。

3.1 アルゴリズムの概要

いま、生物の進化過程では、世代 t における各個体 $p_j(t)$ ($j=1, 2, \dots, M$) は、 n 個の遺伝子の配列で表現されるものとする。GA を本問題に適用するには、まず個体の表現を定義しなければならない。そこで、遺伝子を製品番号、すなわち 1 から N までの数字で表わし、配列の s 番目の遺伝子を第 s 番目に生産される製品に対応させるものとする。したがって、一つの個体は投入順序列 Γ を表わし $n=Q$ である。たとえば、 $N=3$, $Q_1=3$, $Q_2=3$, $Q_3=4$, $Q=10$ のときのある個体はつぎのように表わされる。

1 1 2 3 3 3 1 2 2 3

このような M 個(ただし、 M は偶数とする)の個体からなる個体群 $P(t)=\{p_1(t), p_2(t), \dots, p_M(t)\}$ を考える。

このとき、アルゴリズムの概要はつぎのようになる。

アルゴリズム 1

- STEP 1 初期個体群 $P(1)$ を(32)式を満たす範囲でランダムに生成する。世代 $t=1$ とする。
- STEP 2 各個体 $p_j(t)$ に対して適応度を計算し、その分布に従って適応度の大きい個体はその個数を増加させ、適応度の小さい個体を排除する(この操作を reproduction(複製)と呼ぶ)。
- STEP 3 ある選択確率に従って STEP 4 あるいは STEP 5 のいずれかへ進む。
- STEP 4 M 個体全てをランダムに 2 個体ずつ組み合わせ、各 2 個体からそれぞれ新たに 2 個体を生成する(この操作を crossover(交叉)と呼ぶ)。作られた M 個体の子孫で世代 $(t+1)$ の個体群 $P(t+1)$ を構成する。STEP 6 へ進む。
- STEP 5 M 個の個体の中から 1 個体をランダムに選び、その個体に対してある変化を与えて新たな個体を生成する(この操作を mutation(突然変異)と呼ぶ)。選ばれた個体のみを新たに生成した個体で置き換えて、世代 $(t+1)$ の個体群 $P(t+1)$ を構成する。
- STEP 6 世代数 t があらかじめ定めた値 t^* に達したならば、最も適応度の高い個体を解として計

算を終える。否ならば $t=t+1$ として STEP 2 へ戻る。

以上のように、GA とは多数の解候補を考え、遺伝演算子と呼ばれる reproduction, crossover および mutation の操作を実行し、世代交代を繰り返すことによって最適解を得ようとするものである。

本論文で扱う問題は制約条件が厳しいことが特徴である。そこで、GA を本問題に適用する場合には、実行可能領域内のみを探索することができるように、アルゴリズム 1 の中に出てきた初期個体群の生成方法、遺伝演算子の実行方法およびパラメータの設定を検討しなければならない。そこで、それらの詳細を以下に提案する。

3.2 初期個体群の生成方法

一般に GA ではランダムに初期個体群を生成することが多い。しかし、本アルゴリズムでは実行可能な初期解を得るために、つぎのようにして初期個体群を生成する。ここに、 Q 個の製品のうち s 個が後ろから順序づけられているときその順序列を Γ_{0-s} とする。

アルゴリズム 2

STEP 1 全ての $s(s=1, 2, \dots, Q)$ に対して $\Omega(s)=\{1, 2, \dots, N\}$ とする。

STEP 2 $s=Q$ とおく。このとき、 $t(Q)=T$ であり、 $\Gamma_0=\phi$ とする。

STEP 3 次式を満たす i の中からランダムに一つ選び、それを i_s とし STEP 5 へ進む。

$$(s-1-C_0)h_0 \leq t(s)-h_{i_s} \leq (s-1)h_0 \quad i \in \Omega(s) \quad (33)$$

(33)式を満たす i が存在しなければ STEP 4 へ進む。

STEP 4 $\Omega(s)=\{1, 2, \dots, N\}$, $s=s+1$, $\Omega(s)=\Omega(s)-\{i_s\}$ として STEP 3 へ戻る。

STEP 5 A_s を Γ_s の前に順序づけこれを Γ_{s-1} とする。

STEP 6 $s=1$ ならば一つの個体が得られ STEP 7 へ進む。 $s \neq 1$ ならば $t(s-1)=t(s)-h_{i_s}$, $s=s-1$ として STEP 3 へ戻る。

STEP 7 M 個の個体が得られるまで STEP 1~STEP 6 を繰り返す。

(33)式はパッファ制約条件式(32)と同じである。よって、このアルゴリズムでは制約条件を満たしつつ逐次的に順序づけていくので必ず実行可能解が得られる。

3.3 reproduction の手続き

reproduction は各個体の適応度を用いて適応度の低い個体(いわゆる悪い個体)を排除することにより、解空間における探索領域を削る役割をもつ。個体 p_j の目的関数を z_j とすると、個体 p_j に対する適応度 f_j の表現を次式のように提案する。

$$f_j = \min_{1 \leq k \leq M} z_k \times U + \max_{1 \leq k \leq M} z_k - z_j \quad (34)$$

これより、 f_j は j に依存しない初めの 2 項から z_j を引いたものであるから、目的関数の最も小さい個体が最大の適応度をもつことになる。(34)式の右辺第 2 項は、 f_j の値が常に正となるように導入された定数である。また、第 1 項で U は、正の変数パラメータである。

このようにして得られた適応度 f_j を用いて、個体 p_j が次世代に生き残る個数の期待値 E_j を次式より求める。

$$E_j = \frac{f_j}{\bar{f}} \quad (35)$$

ここに、 $\bar{f}$ は適応度 f_j の平均値である。すなわち、

$$\bar{f} = \frac{1}{M} \sum_{j=1}^M f_j \quad (36)$$

E_j を整数化した値を用いて各個体の増減を行う。

(34)式で U の値を変えることにより、どの個体までを生き残らせるようにするかを決定することができる。すなわち、(34)式において U の値が大きくなると、任意の 2 個体 p_i, p_j の適応度の差 $f_i - f_j$ は変わらないが、適応度そのものの値は各個体とも大きくなり、その平均値 $\bar{f}$ も大きくなる。よって、個体 p_i, p_j の複製後における個数の差は、 $E_i - E_j = (f_i - f_j) / \bar{f}$ となり小さくなる。 E_i の平均値は 1 で変わらないから、各個体の E_i は 1 に接近することになり、増殖や死亡する個体は少なくなる。

GA を構成する場合に問題となるのは、reproduction において各世代でどの個体までを生き残らせるようにするかを決定することである。一般に、次世代の個体群に悪い個体を多く残していると解の改善があまりできず、他方悪い個体を捨てると解のパターンが揃ってしまい結果として得られる解は悪くなってしまう。そこで、本アルゴリズムでは U の値を $U=0.5$ から $U=0.05$ へ世代とともに一定の割合で減少させていくようにして良い解が得られるようにしている。すなわち、初めは個体の多様性を保っておき、最終的には良い個体のみを残すという方法で解を得るわけである。具体的には、世代 t における U の値は次式で与える。

$$U = -\frac{0.45}{t^*} t + 0.5 \quad (37)$$

3.4 crossover の手続き

新しい解空間の探索は crossover および mutation によって行う。crossover は個体群 P 中の 2 個体に対して行う操作である。まず、2 個体を全くランダムに選ぶ。選んだ 2 個体における部分列を互いに交換し、新しい 2 個体を作る。

crossover を行う場合に重要な点は、部分列を交換することによって解の実行可能性が保存されねばならない

ということである。そのためには、これまで order crossover, cycle crossover および partially matched crossover が提案されている¹⁾。これらの方法は、巡回セールスマン問題のような重複を許さない順列の問題に有効である。しかし、本問題の場合には、交換する部分列に対する製品の個数が製品の種類ごとに全て一致しなければならない。また、交換して得られた投入順序列に対してバッファの制約条件を満たさなければならない。以上の条件を満足するために、本論文ではつぎのような方法で crossover を行う。

二つの個体に対して、投入順序列の最初から順番に製品の種類ごとの個数を調べていき、全ての種類の個数が一致した列の位置を記憶しておく。最後まで調べたら、全て一致した位置および最初と最後の位置を要素とする集合を A とする。このとき、集合 A の中からランダムに二つ選び、その二つの位置の間の部分列を交換する。一例として、 $p_1(t)$ および $p_2(t)$ が Fig. 2 のように与えられた場合を考えよう。この場合、 $A = \{0, 5, 8, 13\}$ (0 は最初的位置を表わす) である。いま、 A より 5 および 13 を選ぶと、その間の部分列が交換され、同図のような二つの個体 $p_1(t+1)$ および $p_2(t+1)$ が生成される。

このとき、交換した部分列に対する所要の生産時間は等しく、それぞれの部分列はバッファの制約条件を満たしているのので、交換した後の投入順序列も実行可能解となっている。

3.5 mutation の手続き

上に述べた crossover では A の中から二つの要素を選んで部分列の交換を行い、それ以上の部分列の交換は手続きが複雑になるので行わない。その意味でこの crossover は完備ではないので、これだけでは十分な探索が行えない。そこで mutation との併用で解をさらに改善する。mutation は一つの個体の中の 2 箇所の遺伝子を交換する操作を行う。個体の選び方および遺伝子の交換位置はランダムに決める。選択した交換位置にある数字が同一の場合には、交換しても新しい個体は得られないのでもう一度交換位置を決め直す。また、交換して得られた解が (32) 式を満足せず実行可能とならない場合は、もう一度個体を選ぶところからやり直す。実行可能となる場合は遺伝子を交換し mutation は終了する。

一例として Fig. 3 では、交換位置として前から 2 番目と 9 番目の遺伝子が選択された場合の mutation の例を示している。

3.6 パラメータ設定

GA のパラメータには計算終了までの世代数 t^* 、個体数 M 、crossover や mutation の選択確率がある。本節ではこれらのパラメータの値を設定する。

Fig. 2 Crossover operation

Fig. 3 Mutation operation

t^* および M は値が大きいほど良い解が得られるが計算時間は長くなるものと思われる。そこで、予備計算の結果を参考にして $t^* = 2000$, $M = 60$ とする。また、crossover および mutation の選択確率であるが、GA では crossover を主に行うのが一般的である。そこで、crossover および mutation の選択確率は、それぞれ 0.9 および 0.1 とする。

4. 数 値 例

数値例として、製品の種類 $N = 6$ 、バッファの容量 $B = C_0 = 1$ 、作業者のサイクルタイム $h_0 = 6$ とし、各製品の生産時間をつぎのように与えた。

$$h_1 = 1, h_2 = 3, h_3 = 5, h_4 = 7, h_5 = 9, h_6 = 11 \quad (38)$$

生産量 Q_i は事例によって種々の値を与えた。

まず、バッファの制約条件式 (32) によって実行可能領域がどの程度狭くなっているのかを調べることにする。

Fig. 4 は、 $Q = 10$ のとき (32) 式がない場合の投入順序列の個数を P_1 とし、(32) 式を考慮した場合の投入順序列の個数を P_2 としたときのそれぞれの問題に対する P_2/P_1 を非増加順にプロットしたものである。これより任意に選んだ投入順序列が実行可能となる可能性は高々 4% 程度であり、最も少ない場合では 0.3% 程度しかないことがわかる。

Fig. 4 Rate of feasible solutions ($Q = 10$)

なお、 $Q=10$ の場合は問題の規模が小さいので、列挙法により P_2 個の実行可能な投入順序列を見出すことができた。しかし一般には、 P_1 個のうちの P_2 個の順列を取り出すことは不可能であり、 P_1 個の順列を列挙することもその数が膨大で困難である。

つぎに、本アルゴリズムを適用する場合に、(34)式に含まれるパラメータ U の値が、アルゴリズム終了時に得られた解にどのような影響を与えるかを調べる。一例として、 $Q=50$ の場合の 50 個の事例を計算した結果を **Table 1** に示す。同表の (a), (b) に比べて (c) のように (37) 式による U の値を採用すると、比較的良好な解が得られる。

GA の有効性を検討するためにヒューリスティック解法および Simulated Annealing 法 (SA) によって得られた解と比較する。これらの解法は以下に要約するが、詳しくは文献 5) を参照されたい。

ヒューリスティック解法：

欲張り法を改善した解法で、目的関数値の良さを示す局所的な情報だけで実行可能な投入順序列を直接構成する方法である。

SA：与えられた初期解を改悪も許しながら、徐々に改善して精度の良い解を得ようとするランダム探索法である。逐次改善法としては、順列 I のとなり同士の要素を入れ換えるペア交換法と I の部分列を入れ換えるブロック交換法を併用する。改悪された解を受理する確率を定める温度パラメータ

θ の更新則は、 $\theta \rightarrow 0.98\theta$ とする。GA と比較するために、アルゴリズム 2 で 1 個体だけ生成してこれを初期解として与えるので、この点で文献 5) とは異なる。

まず、総生産量 $Q=10$ および 20 といった小規模な問題について検討する。各製品の生産時間を (38) 式のように与えると、実行可能な事例数は $Q=10$ の場合が 43、 $Q=20$ の場合が 414 であることがわかった。この場合の計算結果を **Table 2** に示す。同表において (a) は平均目的関数値、(b) は平均計算時間である。また、**Fig. 5** は $Q=10$ における GA で得られた目的関数値をそれぞれの事例に対してプロットしたものである。横軸の事例の順序は **Fig. 4** と同じであり、 P_2/P_1 の非増加順に並んでいる。この小規模な場合には、分枝限定法によって最適解を求めることができるので、各解法で求めた目的関数値はそれぞれ最適な目的関数値で除した相対値を用いている。これより GA はほかの解法に比べて、計算時間が長いものの最適解に近い解が得られていることがわかる。また、数事例を除いて、実行可能領域の大きさにかかわらずほぼ一定に良い解が得られており、GA が有効に働いていることがわかる。

つぎに、総生産量が多い場合として $Q=30, 40$ および

Fig. 5 Objective value obtained by GA ($Q=10$)

Table 1 Dependence of the objective value on U ($Q=50$)

U	Objective value average (min. - max.)
(a) 0.5 (constant)	349 (259 - 504)
(b) 0.05 (constant)	387 (255 - 935)
(c) 0.5~0.05 (by (37))	318 (246 - 440)

Table 2 Comparison of computational results for the case of $Q \leq 20$

(a) Average normalized objective value

	Heuristic method	SA	GA
$Q=10$	1.197	1.457	1.028
$Q=20$	1.311	1.478	1.083

(b) Average CPU time (sec)

	Heuristic method	SA	GA
$Q=10$	0.0058	14.47	49.70
$Q=20$	0.061	77.90	142.81

Table 3 Comparison of computational results for the case of $Q > 20$

(a) Average objective value

	Heuristic method	SA	GA
$Q=30$	242.79	256.03	176.53
$Q=40$	368.67	310.33	256.88
$Q=50$	452.39	379.43	317.64

(b) Average CPU time (sec)

	Heuristic method	SA	GA
$Q=30$	0.017	227.65	388.02
$Q=40$	0.022	403.76	492.19
$Q=50$	0.029	704.18	775.80

50 の場合を考える。 $Q=50$ は実際のラインで用いられている規模に対応している。これらの場合では、(38)式の生産時間に対して実行可能な事例数は非常に多くなる。そこで、事例数はそれぞれ 50 に限定して計算した。

Table 3 にそれぞれの結果を示す。この場合は規模が大きいために分枝限定法を用いて最適解を求めることができない。同表より GA は規模が大きくなってもほかの解法に比べて計算時間はかかるものの良い解が得られていることがわかる。

計算時間について SA と GA を比較すると、 $Q=10$ の場合には GA は SA の 3.4 倍かかっているが、 Q の増加とともにその差は少なくなり、 $Q=50$ の場合には GA は SA の 1.1 倍になる。 GA は解集合を扱うために一般に SA よりも多くの計算時間が必要である。しかし、問題の規模が増加するにつれて、 SA の逐次改善法よりも遺伝演算子が有効に作用して、両者の差が接近するものと思われる。

なお、3.5 節で述べたように、mutation より実行可

能とならなかった場合には再試行するが、計算結果より $Q=50$ の場合の平均再試行回数は 4 回であった。すなわち、実行可能解を mutation により変化させて実行可能となる可能性は 25% 程度であり、任意に選んだ投入順序列が実行可能となる可能性(たとえば Fig. 4) に比べて高いことがわかる。

最後に、 $Q=50$ の場合において GA とほかの解法を事例別に比較する。 Fig. 6 は横軸を GA、縦軸をヒューリスティック解法としたときの目的関数値の分布を示したものである。

斜線の上に点があれば、その事例では両解法によって得られた目的関数値は一致することを表わす。同図より斜線より上の部分に位置する点が多くみられる。よって、ほとんどの事例で GA はヒューリスティック解法より良い解が得られていることがわかる。また、 Fig. 7 は横軸を GA、縦軸を SA としたときの目的関数値の分布を示したものである。 GA と SA は同じアルゴリズムで初期解を得ており、 $Q=50$ の場合には計算時間も同程度であるが、解の精度は GA のほうが優れていることがわかる。

なお、計算機としてワークステーション NEWS-3460 を用いたことを付記しておく。

5. 結 言

バッファに制約のある組立ラインにおける生産比率の一定化を目標とした製品投入順序問題に対し、 GA を適用して、制約のある最適化問題に対する遺伝演算子の具体的な構成法を示した。本問題は実行可能解を得ることさえ難しいが、提案したアルゴリズムによって実行可能領域内で解を探索することにより容易に解を得ることができた。考察した問題に対し、607 の数値例の結果を検討したところ、 GA はほかの解法に比べて計算時間は長くなるが、各事例の実行可能領域の大きさによらず、ほぼ一定に精度の良い解を求めることができ、その有効性が確かめられた。今後はより複雑な組立ラインの問題に対する GA の有効性を検討する予定である。

また、本論文では適応度として(34)式を提案した。この式の有用性、特に式中含まれているパラメータ U の役割について、定量的な評価を与えることが今後の課題である。

参 考 文 献

- 1) D. E. Goldberg: Genetic Algorithm in Search, Optimization and Machine Learning, Addison-Wesley (1989)
- 2) 須貝, 平田: 組合せ最適化アルゴリズムとその応用, 計測と制御, 29-12, 1084/1091 (1990)
- 3) 西川, 玉置: ジョブショップ・スケジューリング問題に対する遺伝アルゴリズムの構成法, 計測自動制御学会論文

Fig. 6 Comparison of the objective value between GA and heuristic method ($Q=50$)

Fig. 7 Comparison of the objective value between GA and SA ($Q=50$)

集, 27-5, 592/599 (1991)

- 4) 小谷重徳: 混合ラインへの投入順序付けの近似解法, トヨタ技術, 33-1, 31/38 (1983)
- 5) 三宮, 鳥居, 飯間, 川合: ある製品投入順序問題におけるジャストインタイム最適化, 電気学会システム制御研究会資料, 31/39 (1991)

三 宮 信 夫 (正会員)

1967 年, 京都大学大学院博士課程単位修得退学(電気工学専攻)。京都大学工学部助手, 講師, 助教授を経て, 86 年京都工芸繊維大学工学部教授。システムのモデリングおよび最適化に関する研究に従事(工学博士)。システム制御情報学会, 電気学会などの会員。

[著 者 紹 介]

飯 間 等 (学生会員)

1991 年, 京都工芸繊維大学工学部電気工学科卒業。同年同大学大学院工学科学研究科電子情報工学専攻に入学, 現在に至る。最適化問題の解法として用いられる遺伝アルゴリズムの研究に従事。システム制御情報学会の会員。