フレキシブルショップ問題への遺伝的機械学習アプローチ —— リアルタイム・スケジューリングのためのルール獲得法

榊原一紀[†] 玉置久[††]
村尾元[†††] 北村新三[††]

フレキシブルショップ問題を取り上げ，リアルタイム・スケジューリングをふまえた最適化モデル，すなわち，計画立案段階において「どれだけのジョブがどのタイミングで発生」するのかが十分に予測できない状況下におけるスケジューリング・モデルを対象とする．このようなモデルに対するアプローチとして，遺伝的機械学習（GBML）に基づくスケジューリング・ルールの獲得法を提案する．いくつかの例題に対して提案方法を用いた計算機実験を行い，求められたルールの特徴等を調べる．さらに，ルールによって得られた解の良さ，ルール（集合）の汎用性の評価を通して，提案方法の有効性・可能性について検討する．

Genetics-Based Machine Learning Approach for Flexible Shop Scheduling Problems: Rule Acquisition for Real-time Scheduling

Kazutoshi Sakakibara,[†] Hisashi Tamaki,[††] Hajime Murao[†††] and Shinzo Kitamura[††]

In this paper, we deal with an extended class of flexible shop scheduling problems, and consider a solution under the condition in which information on jobs to be processed may not be given beforehand, i.e., under the framework of real-time scheduling. To realize a solution, we apply such a method where jobs are to be dispatched by applying a set of rules (a rule-set), and propose an approach in which rule-sets are generated and improved by using the genetics-based machine learning technique. Through some computational experiments, the effectiveness and the potential of the proposed approach are investigated.

1. はじめに

近年，スケジューリングは生産システムの計画・運用における重要な問題として注目されており，理論および実用的な観点から多くの報告がなされている．これらの多くは，仕事の属性の値を既知として仮定した静的な問題を対象としている[1),2)]．しかし一般に，生産システム等においては，運用時において，事前に作成された計画では予測・考慮されていないことが起ることが少なくない．さらには，問題の構成要素のすべてがあらかじめ確定的に与えられるとは限らず，事前にスケジュールを用意することが困難であることも多い．このような場合には，実行可能なスケジュールを高速に提供することが重要となり，状況に応じてリアルタイムに意思決定していくアプローチが一般的である．

リアルタイム・スケジューリングに対しては，到着した仕事を随時ディスパッチングするルールを用いたアプローチが代表的である[3)~5)]．しかし問題の規模が大きくなるとルールの抽出・構築および評価は容易なことではない．そこで，我々は生産システムにおけるリアルタイム・スケジューリングの問題を取り上げ，そのモデル化と解法構成に関する研究を進めている．これまでにも，完成した製品を逐次在庫設備に振り分ける在庫設備選択問題を取り上げ，機械学習によりルールを獲得・調整することを目的とした研究を行っている[6)]．

本論文では，現実の様々な問題に対して広範な応用の可能性を有するフレキシブルショップ問題を取り上げる．このフレキシブルショップ問題は，ジョブショップ問題において，各ショップに複数の機械が並列に存在するような問題であり，本研究では，さらに段取り

† 神戸大学大学院自然科学研究科
Graduate School of Science and Technology, Kobe University
†† 神戸大学工学部
Faculty of Engineering, Kobe University
††† 神戸大学国際文化学部
Faculty of Cross-Cultural Studies, Kobe University

替えに関する制約も考慮する．この問題は，ディスパッチング時に割付け設備の選択のみを繰り返せばよいといった在庫設備選択問題におけるリアルタイム・スケジューリングとは異なり，仕事を割り付ける機械の選択に加えて，仕事を処理する順序という時間軸上の広がりを持つような決定を有する．この問題に対しては，これまでに問題を混合整数計画問題として定式化するとともに，定式化を基に進化型計算法による解法の構成に関する研究を行っている[7]．本論文では，このアプローチをベースとしつつ，リアルタイム・スケジューリングの枠組みによるアプローチについて検討する．

フレキシブルショップ問題に対するリアルタイム・スケジューリングとして，ルールを用いる方法を考慮し，このルールを事前に・オフライン的に例題を用いたシミュレーションにより学習・獲得する枠組みを考える．一般に，問題の特徴（計画期間中の平均負荷の大きさ等）に対して，スケジューリング・ルールの性能は均一ではなく，さらには，大規模・複雑な問題において適切なルールの選択・構築は容易ではない．リアルタイム・スケジューリングに関する従来研究[3)~5)]に対して本研究は，あらかじめ特定のスケジューリング・ルール（処理時間最小作業優先則，納期最早作業優先則）が用意されることを仮定しないという特徴を持つ．ルールを自動的に生成・調整する機構としては，遺伝アルゴリズム（Genetic Algorithms: GA）[8)]を用いたルール学習方式である遺伝的機械学習（Genetics-Based Machine Learning: GBML）[9)]の適用を試みる．その際，生産システムの状態に基づいて，作業割付けのための優先度を与えるようなルールを考慮し，このルールそのものを GA を用いて学習することを目的とする．

GBML によるルール学習については，従来，生産システムの状態に基づいて割付け仕事を選択するような if-then 形式のルールを考慮し，このルールを自動的に調整・獲得するために，後件部（then 部）のみを GA を用いて決定する計算モデルが提案されている[10)]．このモデルは，(離散）状態の各々に対していずれか 1 つのルールを対応させておくものであり，規模の大きな問題あるいは本論文で対象とするような複雑な構造を持つ問題には適用が難しい．本論文では，ルール集合が状態空間の一部を保持するものとして，前件部と後件部の双方を GA を用いて決定するような計算モデルを提案する．これにより，大規模・複雑な問題に対しても適切なルールが獲得されることが期待される．また後件部について，文献 6) では，決定事項を直接後件部として記述していたが，本研究で取り扱う問題において，このような単純なルール構成では効果的なルールの獲得が困難であると予想される．そこで，決定の際に用いられる問題の属性値の重みを後件部とするようなルール構成を提案する．これによって，様々な状況に対応したルールを実現することができると考えられる．

2. フレキシブルショップ問題

本研究で対象とするフレキシブルショップ問題を以下のように定義する．N^{J} 個の仕事 J_j（$j = 1, \cdots, N^{\mathrm{J}}$）を，$N^{\mathrm{M}}$ 台の機械 M_i（$i = 1, \cdots, N^{\mathrm{M}}$）で処理するものとする．仕事 J_j は，n_j 個の作業 O_k（$k = N_{j-1} + 1, \cdots, N_j$; $N_j = \sum_{\ell=1}^{j} n_\ell$; $N_0 = 0$）で構成され，各仕事の作業は，$\mathrm{O}_k \rightarrow \mathrm{O}_{k+1}$（$k = N_{j-1} + 1, \cdots, N_j - 1$）の順に処理される．各作業 O_k（$k = 1, \cdots, N$; $N = N_{N^{\mathrm{J}}} = \sum_{j=1}^{N^{\mathrm{J}}} n_j$）に対して処理可能機械集合 $\mathcal{M}_k$ と，処理タイプ δ_k といった属性があり，処理タイプ δ_k と機械 M_i の処理能力 μ_i との組合せによって処理時間 p_{ik} が決定する．

スケジューリングに際しては，次のような制約が課せられる：

（a）処理機械：各作業は処理可能機械のいずれか 1 つにおいて処理される．

（b）段取り替え時間：ある機械上で連続して処理される作業の処理タイプが異なる場合，2 つの作業の間に段取り替え時間が必要となる．段取り替え時間は前段取りと後段取りの 2 つの部分からなる．

問題をモデル化するにあたり，まず，

N^{M} ：機械数，

N^{J} ：仕事数，

N ：作業数（$= N_{N^{\mathrm{J}}}$），

とし，機械 M_i，仕事 J_j および作業 O_k に対する属性を以下のように定義する：

（A）機械属性

$\mu_{i\delta}$ ：処理タイプ δ に対して，機械 M_i 上における単位量あたりの処理時間．

（B）仕事属性

d_j ：納期，

r_j^{J} ：処理量，

n_j ：作業数．

（C）作業属性

δ_k ：処理タイプ，

r_k^{O} ：処理量，

s_k^1，s_k^2：前段取り時間および後段取り時間，

$\mathcal{M}_k$：作業 O_k を処理可能な機械の集合．

p_{ik} ：処理時間（$p_{ik} = \mu_{i\delta_k} \times r_k^{\mathrm{O}}$）.

スケジュールの評価については様々な指標が存在するが，ここでは最大完了時間 $C_{\max}$ と納期遅れ和 T_{sum} を考え，

$$z_1 = C_{\max} = \max_j t_{N_j}^{\mathrm{F}} = \max_k t_k^{\mathrm{F}}\,, \qquad (1)$$

$$z_2 = T_{\mathrm{sum}} = \sum_j \left\{\max\left(0,\, t_{N_j}^{\mathrm{F}} - d_j\right)\right\}\,, \quad (2)$$

とする．ここで，

t_k^{S}：作業 O_k の処理開始時刻，

t_k^{F}：作業 O_k の処理終了時刻，

である．このとき，目的関数はこれらの重みつき和で与えられるものとし，

$$z = w_1\, z_1 + w_2\, z_2 \qquad (3)$$

とする．ここで w_1 および w_2 は正の定数である．

このとき問題は，上述の制約を満たしながら評価値が最小となるようなスケジュール，すなわち

（1）作業の機械割付け，

（2）機械上の作業の処理順序，

（3）作業の開始時刻，

（4）作業の終了時刻，

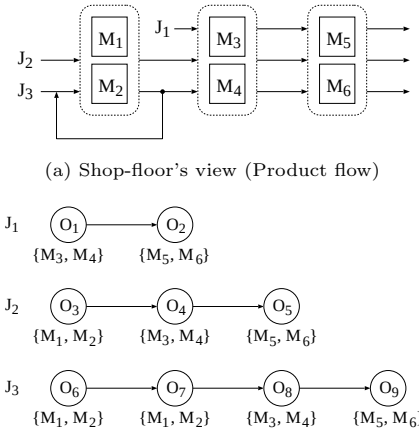
を決定することであると定義される．このとき，(1) の作業の機械割付けを決めることにより作業の処理時間属性が決定する．フレキシブルショップ問題の一例を**図 1** に示す．

ここで取り上げた目的関数の下での最適スケジュールは，一般にアクティブ・スケジュールであることが知られている[1)]．スケジュールのクラスをアクティブに限定した場合には，決定項目（1）および（2）によってスケジュールが一意に定められる．

スケジューリングの過程において，意思決定のタイミングに注目すると，本研究では事前に仕事に関する情報が与えられない状況下におけるスケジューリングを考慮している．よって，事前にスケジュールを準備することができず，仕事が到着した時点でその仕事のスケジュールを決定することになる．

3. 優先度に基づくスケジューリング

フレキシブルショップ問題に対し，ディスパッチング・ルールを用いたリアルタイム・スケジューリングの方法について考える．2 章で示したように，この問題に対してはアクティブ・スケジュールのクラスを考えればよく，以下では，上工程より順に作業を割り付けるフォワード スケジューリングを前提とする．具体的には，いずれかの機械で仕事が完了する時刻を τ とし，処理開始可能作業を割り付けたとして最早に完了する機械を割付け対象の機械 $\mathrm{M}_{i^*}(\tau)$ に選ぶ．割付け作業 $\mathrm{O}_{\tilde{k}}$ については，$\mathrm{M}_{i^*}(\tau)$ 上で時刻 τ において，（仕事内の）直前の作業が処理中であるかあるいは処理済みであるような作業の集合 $\mathcal{O}_{i^*}^3(\tau)$ の中から優先度に基づいて決定する．具体的なスケジューリングの手続きを 3.1 節に示す．また，**図 2** にその一例を示す．



(a) Shop-floor's view (Product flow)

(b) Operations' view (Technical order)

図 1 問題例 — 6 機械 3 仕事 9 作業問題
Fig. 1 Example with 6 machines, 3 jobs and 9 operations.

3.1 スケジューリングの手順

1° （初期化）

ディスパッチングのタイミングを $\tau = 0$，割付け済み作業集合を $\mathcal{O}^1 = \emptyset$ とする．

2° （割付け機械の決定）

（a）機械 M_i に対して τ における開始可能作業集合 $\mathcal{O}_i^2(\tau) = \emptyset$ とする．作業 O_k が仕事内の処理順において先頭（ある j について $k-1 = N_j$）であるか，あるいは直前の作業が割付け済み（$\mathrm{O}_{k-1} \in \mathcal{O}^1$ かつ，すべての j について $k-1 \neq N_j$）であれば，O_k を割付け可能なすべての機械（$\mathrm{M}_i \in \mathcal{M}_k$）に対し $\mathcal{O}_i^2(\tau) = \mathcal{O}_i^2(\tau) \cup \{\mathrm{O}_k\}$ とする．いずれの i に対しても $\mathcal{O}_i^2(\tau) = \emptyset$ ならば，$\tau = \tau + 1$ として 2° へ．

（b）$\mathrm{O}_k \in \mathcal{O}_i^2(\tau)$ に対し，最早の開始時刻：

$$t_k^{\mathrm{S}'} = \begin{cases} \max(t_{k-1}^{\mathrm{S}} + p_{i,k-1}, \\ \quad t_{\ell_i}^{\mathrm{S}'} + p_{i\ell_i} + s_{\ell_i k}\Delta_{\ell_i k}\,), \\ \quad \text{if} \quad k \neq N_{j-1},\ j = 1, \cdots, N^{\mathrm{J}}, \\ \max(0,\ t_{\ell_i}^{\mathrm{S}'} + p_{i\ell_i} + s_{\ell_i k}\Delta_{\ell_i k}), \\ \quad \text{otherwise}, \end{cases}$$

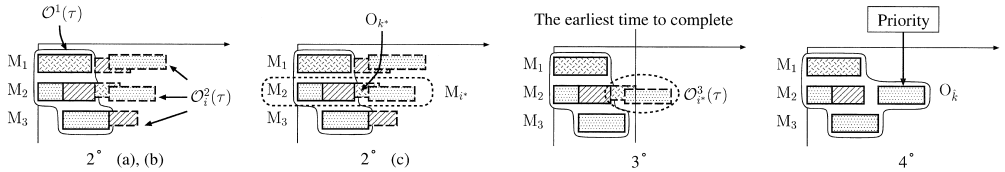


図 2 スケジューリング手続き
Fig. 2 Procedure for generating a schedule.

を求める（O_{ℓ_i} は機械 M_i の最後尾に割り付けられた作業；$\Delta_{kk'}$ は $\delta_k \neq \delta_{k'}$ のとき 1，そうでないとき 0 であるような定数）．

（c）$\bigcup_i \mathcal{O}_i^2(\tau)$ の中で完了時間（$t_k^{S'} + p_{ik}$）が最小となる作業ならびにその処理機械を求め，それぞれ O_{k^*} および M_{i^*} とする．

3°（割付け作業候補の決定）
$\mathcal{O}_{i^*}^2(\tau)$ の中で $t_k^{S'} < t_{k^*}^{S'} + p_{i^*k^*}$ を満たす作業の集合を $\mathcal{O}_{i^*}^3(\tau)$（$\subseteq \mathcal{O}_{i^*}^2(\tau)$）とする．

4°（作業の機械割付け）
すべての $O_k \in \mathcal{O}_{i^*}^3(\tau)$ に対して優先度を求め，優先度の最も高い作業を $O_{\hat{k}}$ とし，$O_{\hat{k}}$ の開始時刻を $t_{\hat{k}}^S = t_{\hat{k}}^{S'}$ とする．$\mathcal{O}^1 = \mathcal{O}^1 \cap \{O_{\hat{k}}\}$ とする．

5°（終了判定）
$|\mathcal{O}^1| = N$ なら終了．そうでなければ 2° へ．

3.2 優先度の計算方法

3.1 節の 4° における割付け作業の決定のための優先度 ϕ_{uk} を，次式を用いて算出する：

$$\phi_{uk} = \sum_{\ell=1}^{n_a} w_{u\ell} a_{k\ell}. \tag{4}$$

ただし $a_{k\ell}$（$k = 1, \cdots, N$; $\ell = 1, \cdots, n_a$）は，作業 O_k に付随する属性（優先度属性）値を，$w_{u\ell}$ は状態（生産システムがとりうる状態（空間）$\mathcal{S}$ をあらかじめ分割したもの）が $\mathcal{S}_u$ である場合の優先度属性値の重み係数をそれぞれ表す．

優先度属性としては，作業に付随する属性として次の 4 種類（$n_a = 4$）を考える：

（A）処理時間（段取り替え時間を含む）$\hat{p}_{ik}$，
（B）処理可能機械台数 $|\mathcal{M}_k|$，
（C）納期余裕時間 u_k ☆，
（D）残りの工程数 n_k^L．

また，生産システム全体の状態を表す指標としては，
（1）仕事の完成率 c^J，
（2）最大完了時間 $C_{\max}$，
（3）納期余裕時間平均 D_{avg}，
（4）納期余裕時間分散 D_{var}，
を考え，この 4 つの指標値の直積空間 $\mathcal{S}$ により状態空間を定義する．

部分状態 $\mathcal{S}_u$ に対する重み $w_{u\ell}$ の決定にルールを用いる方法を考える．ルールは前件部（⟨condition⟩）と後件部（⟨action⟩）からなる一般的な構成：

if ⟨condition⟩ then ⟨action⟩ (5)

とし，前件部に 4 つの離散化された指標値からなる数値列が記述される．これより，ある 1 つの部分状態（空間）$\mathcal{S}_u$ が指定される．また後件部には，各 ℓ（$= 1, \cdots, n_a$）について前件部で表される部分状態 $\mathcal{S}_u$ における優先度属性の重み $w_{u\ell}$ をそれぞれ記述する．ここで各重みは $\{-n_\ell^w, \cdots, n_\ell^w\}$ のいずれかの整数値をとるものとする．

スケジューリングを行うにあたっては，n_r 個のルールからなる集合 $\{r_1, r_2, \cdots, r_{n_r}\}$ を用意しておく．各ディスパッチングのタイミングにおいて，生産システムの状態（入力状態）と各ルールの前件部が比較され，入力状態が属する部分状態を前件部として持つようなルールが適用される．ここで，ルール集合が状態空間全体（$\bigcup_u \mathcal{S}_u$）に対応することを仮定しない（各々の $\mathcal{S}_u$ に対して，対応する前件部を持つルールがルール集合中に存在するとは限らない），ということに注意されたい．つまり，ルール集合は状態空間の一部を保持する．

なお，入力状態が属する部分状態を前件部として持つルールがない場合には，入力状態（四次元空間内の 1 点に対応）と，ルールの前件部が持つ部分状態を代表する点（重心）とのユークリッド距離を計算し，この距離が最小となるルールを適用するものとする．これにより，状態空間の規模とは無関係にルール集合を構成することが可能となる．

4. 遺伝的機械学習によるルール学習方式

3 章で示したルールを自動的に獲得・調整するために遺伝アルゴリズム（GA）[8] を用いたルール学習方式である遺伝的機械学習（GBML）[9] を適用する．

☆ $u_k = d_j - t_k^{F*}$（$k = N_{j-1} + 1, \cdots, N_j$）．ただし t_k^{F*} は（O_k に後続する）未割付けの作業を，作業間の先行制約に基づいて（処理可能機械集合の中で処理時間が最大となる機械に）割り付けたときの仕事 J_j の完了時刻を表す．

GBML は，記号列（染色体）にコーディングされたルールあるいはルール集合を，GA を用いて自動的に選択・生成するものであり，大きく 2 つのアプローチ，すなわち Michigan アプローチと Pitt アプローチに分類される．Michigan アプローチでは各ルールを 1 個体と見なし，ルール集合を用いて得られる結果に基づいて各ルールの信頼度を変更しつつ，適当なタイミングで GA を適用して新しいルールを生成する．一方，Pitt アプローチではルール集合を 1 つの個体に対応させ，各ルール集合を用いた場合の結果に対する評価値を適応度と見なして GA を適用し，新しい記号列（ルール集合）を生成する．

ここで，Michigan アプローチでは個体（ルール）の適応度の算出にクレジット分配機構を用いており，計算機上に実現することは容易ではない．これに対して，Pitt アプローチではあらかじめルール集合を生成・獲得しておくといった枠組みになるものの，Michigan アプローチのようなクレジット分配システムを必要としないために実装が容易である．そこで，本研究では Pitt アプローチを用いた解法構成を考える．

GBML の適用にあたっては，ルール集合を 1 個体と見なし，各ルール集合を用いて得られる結果に対する評価値を適応度として GA を適用する Pitt アプローチの構成を考える．

4.1 ルール学習の手順

フレキシブルショップ問題に対して Pitt アプローチを適用する場合の枠組みを **図 3** に示す．このとき，Pitt アプローチによるルール学習の手順は以下のようになる．

1° ルール集合を個体にコーディングし，初期個体集合 $\mathcal{P}(1)$ をランダムに（ルールを構成する各属性値を，定められた範囲内で一様乱数を用いて与えることにより）生成する．計算世代数 n_{g} を設定するとともに，世代を $t=1$ と初期化する．

2° $\mathcal{P}(t)$ 内の各個体（ルール集合）を問題に適用し，その結果に基づいて適応度を計算する．

3° $t<n_{\mathrm{g}}$ であれば 4° へ．そうでなければ 5° へ．

4° 個体集合に対して遺伝演算子を適用し，次の世代の個体集合 $\mathcal{P}(t{+}1)$ を生成する．世代を $t=t+1$ とし，2° へ．

5° 計算終了．これまで得られた最大適応度の個体からルール集合を求める．

4.2 遺伝アルゴリズムの構成

GBML の適用に際しては，GA における個体の遺伝子表現，適応度の計算，世代交代モデルおよび遺伝演算子を具体化する必要がある．

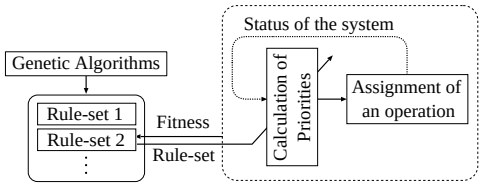


図 3 Pitt アプローチによるルール学習の枠組み
Fig. 3 Outline of the Pitt approach.

個体の遺伝子表現

Pitt アプローチでは，ルール集合を GA における個体として取り扱う．そこでルール r_ℓ（$\ell=1,\cdots,n_{\mathrm{r}}$）を複数個並べて表されるルール集合：

$$\{\mathrm{r}_1,\mathrm{r}_2,\cdots,\mathrm{r}_{n_{\mathrm{r}}}\}$$

を個体の遺伝子表現とする．

適応度計算

個体の適応度はルール集合の評価値に対応している．本研究ではスケジュールの評価値，すなわち式 (3) で表される値をルール集合の評価値とする．なお，3.2 節で述べたように，優先度計算において適用するルールが見つからない場合には，ペナルティとして十分大きな値を評価値とする．

世代交代モデルと遺伝演算子

GA の適用において，各世代における個体すなわちルール集合の多様性の維持の観点から，世代交代モデルとして Steady-state GA [8] を用いる．Steady-state GA の適用においては，個体群の中からランダムに 2 個体を抽出し，遺伝的操作を適用して，新たに 1 個体を生成する．新たに生成された子個体の適応度を計算し，その値が親個体群の最悪個体の適応度値よりも優れている場合には，最悪個体と子個体を入れ替えた個体群を次世代に残す．このとき，遺伝的操作として次の 2 つを考える：

（a）交叉：ランダムにペアリングされた 2 個体に対して多点交叉を行い，子個体を 1 つ生成する．この操作により，いずれかの親個体（ルール集合）のルールを含む子個体が生成される．

（b）突然変異：個体の遺伝子座をランダムに 1 カ所選択し，突然変異確率 p_{m} で他の値に（前件部に関しては（突然変異を加えた後に）いずれかの部分状態 $\mathcal{S}_u$ に対応するように，また後件部に関しては $\{-n_\ell^{\mathrm{w}},\cdots,n_\ell^{\mathrm{w}}\}$ の範囲に含まれるように）変更する．この操作により，新たなルールを含む個体（ルール集合）が獲得される．

5. 計算機実験

3 章および 4 章で述べた解法を，**表 1** に示す 3 つの

表 3 スケジュールの評価値
Table 3 Results of computational experiments.

	GBML	GA	CPLEX
E_1	50.0 (70, 30)	50.0 (70, 30)	50.0 (70, 30)
E_2	78.5 (151, 6)	77.5 (155, 0)	—
E_3	2971.0 (4450, 1492)	2771.0 (4395, 1027)	—

∗ In the parenthesis, two kinds of sub-objective functional values (z_1, z_2) are also shown.

表 1 例題のパラメータ
Table 1 Examples of the problem.

	N^{M}	N^{J}	N	(w_1, w_2)
E_1	6	6	18	(0.5, 0, 5)
E_2	12	9	45	(0.5, 0, 5)
E_3	11	70	219	(0.5, 0, 5)

表 2 解法のパラメータ設定値
Table 2 Setting of the parameters.

Population size n_{p}	100
Number of generations n_{g}	5000
Mutation rate p_{m}	0.1/individual
Number of rules n_{r}	50

例題（E_1，E_2 および E_3）に適用し，(1) 特定の例題に対してルール学習を行った場合に，どのようなスケジュールが得られるか（ルールの学習状況），(2) (1) で得られたルール集合の内容，および (3) ルール学習により得られたルール集合を用いて，(ルール学習用とは異なる）新たな例題に適用した場合に，どのようなスケジュールが得られるか（ルール集合の汎用性），に関して検討する．

5.1 ルールの学習状況について

例題 E_1，E_2 および E_3 に対し，乱数系列を変えて 10 回の試行を行った．この際各例題に対して，状態の分割数 $|\mathcal{S}_u| = 10^4$，ルールの後件部のとりうる値の範囲を $n_\ell^{\mathrm{w}} = 6$（$\ell = 1, \cdots, 4$）と設定した．GBML を適用する際のパラメータを**表 2** にように設定した場合に，得られた最良のスケジュールの評価値を**表 3** に示す．なお，表 3 には 3 章で示したスケジューリング方法における作業 O_k の優先度 ϕ_k を遺伝子表現：

$$\{\phi_1, \phi_2, \cdots, \phi_N\}$$

として，GA を適用する方法（方法 GA）により得られた最良スケジュールの評価値を合わせて示しておく．方法 GA の適用にあたっては，世代交代モデルとして Steady-state GA を適用し，個体数 n_{p}，世代数 n_{g} および突然変異率 p_{m} については表 2 と同じ設定とした．さらに表 3 には，例題 E_1 に対して商用の数理計画パッケージ（CPLEX [11]）を用いて得られた最適スケジュールの評価値を合わせて示す．ルール獲得に要した時間（10 試行の平均値，Pentium III 933 MHz 使用時）を，方法 GA の適用（10 試行の平均値）および CPLEX の適用に要した時間とともに**表 4** に示す．

表 4 計算時間 [秒]
Table 4 Computational time [s].

	GBML	GA	CPLEX
E_1	1.00	0.42	482.74
E_2	4.40	3.64	—
E_3	68.68	39.24	—

表 5 適用されたルール数
Table 5 Number of rules applied for scheduling.

	E_1	E_2	E_3
Number of rules	6	8	7

表 3 および表 4 より，

- 方法 GBML により求められるルール集合を用いて得られるスケジュールは，方法 GA により求められるスケジュールに対して数%劣る程度のものである，また，方法 GBML に要する計算時間は方法 GA にかかる時間の 1.2〜2.4 倍程度である，

ことが確認される．方法 GBML と方法 GA では，同じ決定（作業の優先度）に対してそれぞれ異なる探索空間が定められていることから，方法 GBML で用いたルール集合による探索が，決定そのものを探索空間とするものと同程度の性能を有すること，およびこのルール集合の獲得に要する計算時間が許容範囲内であること，が分かる．

5.2 得られたルール集合について

ルール学習により得られた最良のルール集合について，計画期間中に優先度計算で用いられたルールの数を**表 5** に示す．なお，仮にディスパッチングの各タイミングにおいて互いに異なるルールが適用された場合，計画期間中に適用されたルールの数は作業数 N に等しくなる．さらに，スケジューリングの過程におけるルールの適用状況を見るために，例題 E_3 に対する各ディスパッチングのタイミングでの作業の優先度属性の重みを**図 4** に示す．

表 5 より，

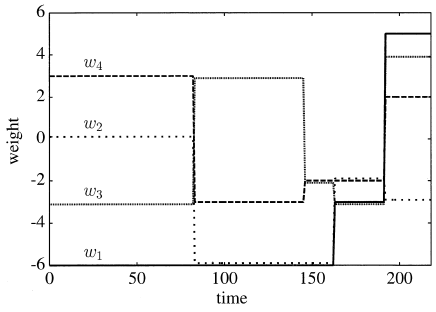


図 4 優先度属性値の重み
Fig. 4 Weight of attributes associated with operation.

- 3つの問題に対して，優先度計算に用いられるルールの数は少ない

ことが分かる．最良ルール集合により，最適もしくは準最適なスケジュールが得られていること（表 3）および，提案方法により汎用的なルール集合が得られていること（5.3 節に後述）を考え合わせると，最適・準最適なスケジュールを得るための汎用的なルールが提案方法により絞り込まれていると考えられる．ただし表 3 より，規模の大きな問題（E_3）に提案方法を用いた場合には，方法 GA よりも評価値が 1 割近く劣っていることが分かる．提案方法ではルールの前件部，すなわち状態空間の定義ならびに分割をあらかじめ定めておく必要があるが，これが適切になされていなかったことが原因として考えられる．今後，状態空間の定義ならびに分割方法について，さらにはそれらの自動化について考察・検討が必要である．

ルールの適用状況について見てみると，図 4 より，

- 作業の優先度属性の重みが時間とともに変化している

ことが分かる．すなわち，システムの状態に応じて各重みの値および（それにともなう）重みどうしの相対的な関係が変化しており，ルールを用いて優先度属性の重みを動的に切り替えることにより，多様なヒューリスティクスを内包するようなルール集合が獲得されていることが確認できる．

5.3 ルール集合の汎用性について

例題 E_1，E_2 および E_3 に対して，それぞれ納期と処理量に乱数による摂動を加えることにより,（学習のための）例題を 20 個ずつ作成した（$\mathcal{E}_1^{\mathrm{L}}$，$\mathcal{E}_2^{\mathrm{L}}$ および $\mathcal{E}_3^{\mathrm{L}}$）．これらの例題を用いたルール学習を行い（20 個の例題の平均評価値を各世代における適応度とする），得られた最良ルール集合を，$\mathcal{E}_1^{\mathrm{L}}$，$\mathcal{E}_2^{\mathrm{L}}$ および $\mathcal{E}_3^{\mathrm{L}}$ とは異なる（新たに作成した評価のための）20 個の例題 $\mathcal{E}_1^{\mathrm{E}}$，$\mathcal{E}_2^{\mathrm{E}}$ および $\mathcal{E}_3^{\mathrm{E}}$ にそれぞれ適用した．評価用の例題のそれぞれに（学習用の例題を用いて獲得された）ルール集合を適用した際の評価値を，これらの例題の最良評価値（$\mathcal{E}_1^{\mathrm{E}}$ については最適スケジュール，$\mathcal{E}_2^{\mathrm{E}}$ および $\mathcal{E}_3^{\mathrm{E}}$ については方法 GA を用いて得られた最良スケジュール）との比をとったものを**表 6** に示す．

表 6 獲得されたルールの汎用性
Table 6 Robustness of the obtained rule-set.

	Ratio of the obtained value to the best objective			
	Maximum	Minimum	Average	Std. Dev.
$\mathcal{E}_1^{\mathrm{E}}$	1.29	1.04	1.09	0.06
$\mathcal{E}_2^{\mathrm{E}}$	1.18	1.01	1.07	0.05
$\mathcal{E}_3^{\mathrm{E}}$	1.31	1.07	1.16	0.07

表 6 より，獲得されたルール集合は，

- $\mathcal{E}_1^{\mathrm{E}}$ に対して最適スケジュールから約 1 割程度劣ったスケジュールを平均的に得ている，

ことが分かる．さらに

- $\mathcal{E}_2^{\mathrm{E}}$ および $\mathcal{E}_3^{\mathrm{E}}$ に対し，方法 GA で獲得されたスケジュールと比較して，$\mathcal{E}_1^{\mathrm{E}}$ の場合と同様な結果が得られている．

これより，提案方法によって様々な状況（問題）に対しても良好なスケジュールを生成しうるような汎用的なルール集合が獲得されていると評価できる．

以上の結果より，本研究で構成したルール集合による探索方法（方法 GBML）は方法 GA に比べて探索効率が若干劣るが，これは許容範囲内であり，一方で，方法 GBML によって汎用性に優れたルール集合が得られることが分かる．これらより，本論文で提案した方法はその利用価値が高いものであると考えられる．

ただし，$\mathcal{E}_3^{\mathrm{E}}$ に対しては，方法 GA による最良スケジュールから最悪の場合に 30% 以上評価値が悪くなっている．この点を含め，獲得されたルールの内容および，得られたスケジュールそのものに関する詳細な検討については，今後の課題としたい．

6. おわりに

本論文では，フレキシブルショップ問題を対象として，リアルタイム・スケジューリングの条件下で解法構成を行った．その際，システム全体の状態に基づいて，作業の優先度を決定するルールを用い，ルールを自動的に生成・調整するために GBML を適用した．さらに，提案方法の有効性・可能性について計算機実験により検討した．

今後の課題として，以下の点があげられる：

（a）生産システムの状態空間の定義ならびに分割の方法について，さらにはそれらの自動獲得についての検討．

（b）獲得されたルールの内容および，得られたスケ

ジュールそのものに関する詳細な検討，および（c）提案方法のオンライン学習への拡張についての検討．

参 考 文 献

1) Pinedo, M.: *Scheduling: Theory, Algorithms, and Systems*, 2nd edition, Prentice-Hall (2002).
2) Blazewicz, J., Ecker, K., Pesch, E., Schmidt, G. and Weglarz, J.: *Scheduling Computer and Manufacturing Processes*, Springer-Verlag (1996).
3) Shaw, M.J., Park, S. and Raman, N.: Intelligent Scheduling with Machine Learning Capabilities — The Induction of Scheduling Knowledge, *IEE Trans.*, Vol.24, No.2, pp.156–168 (1992).
4) Shafaei, R. and Brunn, P.: Workshop scheduling using practical (inaccurate) data Part 1: The performance of heuristic scheduling rules in a dynamic job shop environment using a rolling time horizon approach, *Int. J. of Production Research*, Vol.37, No.17, pp.3913–3925 (1999).
5) Shafaei, R. and Brunn, P.: Workshop scheduling using practical (inaccurate) data Part 2: An investigation of the robustness of scheduling rules in a dynamic and stochastic environment, *Int. J. of Production Research*, Vol.37, No.18, pp.4105–4117 (1999).
6) 榊原一紀，玉置 久，村尾 元，北村新三，岩谷敏治，松田浩一：リアルタイムスケジューリングに対する遺伝的機械学習アプローチ，電気学会論文誌 C，Vol.123, No.4, pp.823–831 (2003).
7) Sakakibara, K., Tamaki, H., Murao, H. and Kitamura, S.: Mathematical Modeling and Hybrid Solution for a Class of Flexible Shop Scheduling Problems, *Proc. Int. Symp. on Scheduling 2002*, pp. 93–96 (2002).
8) Bäck, T., Fogel, D.B. and Michalewicz, Z.(Eds.): *Evolutionary Computation 1*, Institute of Physics Publishing (2000).
9) Smith, S.F.: A Learning System Based on Genetic Algorithms, Ph.D. Thesis, Univ. of Pittsburgh (1980).
10) 玉置 久，越智通有，荒木光彦：スケジューリング・ルール選択における状態フィードバックの試み，計測自動制御学会論文集，Vol.35, No.3, pp.428–434 (1999).
11) ILOG Inc.: *CPLEX 8.0*, http://www.ilog.co.jp/ (2001).

(平成 15 年 4 月 12 日受付)
(平成 15 年 5 月 30 日再受付)
(平成 15 年 10 月 5 日採録)

榊原 一紀

1975 年 5 月 15 日生．1999 年神戸大学工学部電気電子工学科卒業．2001 年神戸大学大学院自然科学研究科博士課程前期課程修了．2001 年神戸大学大学院自然科学研究科博士課程後期課程進学，現在に至る．スケジューリング・ルール獲得，およびリアルタイム/リアクティブ・スケジューリングに関する研究等に従事．計測自動制御学会，システム制御情報学会等の会員．

玉置 久

1962 年 12 月 14 日生．1985 年京都大学工学部電気工学科卒業．1990 年京都大学大学院工学研究科博士後期課程研究指導認定退学．同年京都大学工学部助手，1995 年神戸大学工学部講師，1999 年同助教授，現在に至る．スケジューリング問題のモデル化と解法，創発的問題解決の方法論等の研究に従事．博士（工学）．IEEE，計測自動制御学会，システム制御情報学会等の会員．

村尾 元

1969 年 1 月 7 日生．1991 年神戸大学工学部計測工学科卒業．1995 年神戸大学大学院自然科学研究科（博士後期課程）2 年次中途退学．同年神戸大学工学部情報知能工学科助手，2003 年国際文化学部助教授，現在に至る．博士（工学）．強化学習，創発システム等の研究に従事．計測自動制御学会，システム制御情報学会等の会員．

北村 新三

1940 年 5 月 31 日生．1963 年神戸大学工学部計測工学科卒業，東レ（株）勤務．1966 年神戸大学大学院工学研究科修士課程修了．大阪大学工学部助手，神戸大学助教授を経て，1985 年教授．1995 年 2 月大学院自然科学研究科長，1997 年 2 月工学部長併任（2000 年 3 月まで）．1998 年度システム制御情報学会会長．2002 年 4 月神戸大学副学長，現在に至る．ロボット制御，創発システム等の研究に従事．工学博士．システム制御情報学会，計測自動制御学会等の会員．