総納期ずれ最小化のためのマルチエージェント強化学習を用いた自律分散型スケジューリング*

岩村 幸治[†]・横手 隆幸[‡]・菅野 翼[‡]・杉村 延広[†]

Autonomous Distributed Scheduling by Using Multi-agent Reinforcement Learning for Minimizing Sum of Earliness and Tardiness*

Koji IWAMURA[†], Takayuki YOKOTE[‡], Tsubasa SUGANO[‡] and Nobuhiro SUGIMURA[†]

Multi-agent reinforcement learning have been applied to scheduling method based on utility values for autonomous distributed manufacturing systems in order to improve the objective functions of individual job agents and resource agents, in the previous researches. New distributed scheduling method based on utility values is proposed to improve the sum of earliness and tardiness of all the job agents by applying the multi-agent reinforcement learning to the resource agents, in this research. The resource agents learn the suitable criteria to determine the utility values of the candidate job agents for next machining operation based on the status of manufacturing systems. Some case studies have been carried out to verify the effectiveness of the proposed method.

1. はじめに

生産システムの構成要素が通信機能と意思決定機能を持ち、情報交換を行いながら自律的に生産プロセスを実行する自律分散型生産システム (ADMS: Autonomous Distributed Manufacturing Systems)[1–3], ホロニック生産システム (HMS: Holonic Manufacturing Systems)[4], 生物指向型生産システム (BMS: Biological Manufacturing Systems)[5] などが提案されている。

著者らは、加工機械および加工される製品を対象とした ADMS におけるジョブショップスケジューリング問題について研究を行っている。この問題では、製品の各加工プロセスを実行する順序および加工プロセスを実行できる加工機械が制約条件となり、加工機械における各製品の加工プロセスの実行順序が設計変数となる。

この ADMS におけるジョブショップスケジューリン

グ問題に関して、これまでの研究[3]では、加工される製品を表すジョブエージェントおよび加工機械を表すリソースエージェントにマルチエージェント強化学習を適用し、効用値に基づいてつぎに実行する加工プロセスのスケジュールを決定する手法を提案した。

この手法では、個々のジョブエージェントが滞留時間または加工コスト、個々のリソースエージェントが稼働率または加工精度の目的関数をもつ場合に、ジョブエージェントおよびリソースエージェントにマルチエージェント強化学習を適用することで、生産システムの状態に応じて適切な効用値を算出することができ、目的関数の値がマルチエージェント強化学習を適用しない場合と比較してよくなることを示した。

ただし、これまでの研究[3]では、ジョブエージェントおよびリソースエージェントが異なる目的関数をもつため、つぎの加工プロセスを実行する候補エージェントを選択した場合の目的関数の評価値の和が最大となるように、リソースエージェントとジョブエージェントの組合せを決定するためのコーディネーションエージェントが必要であった。また、これまでの研究では、個々のジョブエージェントおよびリソースエージェントの目的関数をよくするためのマルチエージェント強化学習の適用方法については検討しているが、スケジューリングにおけるシステム全体の目的関数をよくするための方法については検討できていないという課題があった。

* 原稿受付 2016年9月5日

[†] 大阪府立大学 大学院 人間社会システム科学研究科
Graduate School of Humanities and Sustainable System
Science, Osaka Prefecture University; 1-1 Gakuen-cho,
Naka-ku, Sakai city, Osaka 599-8531, JAPAN

[‡] 大阪府立大学 大学院 工学研究科 Graduate School of
Engineering, Osaka Prefecture University; 1-1 Gakuen-
cho, Naka-ku, Sakai city, Osaka 599-8531, JAPAN

Key Words: multi-agent reinforcement learning, autonomous distributed scheduling, minimizing sum of earliness and tardiness.

本研究では、コーディネーションエージェントが存在しない自律分散型生産システムにおいて、個々のジョブエージェントおよびリソースエージェントの目的関数ではなく、ADMS全体の目的関数の一つであるジョブエージェントの総納期ずれの最小化を目的関数として、リソースエージェントがつぎに加工プロセスを実行するジョブエージェントを選択するための適切な効用値を、マルチエージェント強化学習を用いて決定する手法について提案する。

納期ずれ最小化を目的としたスケジューリング手法に関する研究は、ジャストインタイム生産に必要な生産管理の手法として活発に行われており、たとえば、混合整数計画法を用いた解法[6]、メタヒューリスティクスを用いた解法[7]が提案されているが、これらはいずれも生産システム全体を一つの要素が管理してスケジューリングを行うことを前提としているため、本研究で対象とする自律分散型のスケジューリング手法には適用できない。また、自律分散型のアプローチとして、マルチエージェントを用いた解法[8]が提案されているが、この研究では、まだスケジュールされていないジョブを表すエージェント、すでにスケジュールに割り当てられているジョブ群を表すエージェント、およびこれらを管理するエージェントから構成しており、本研究における個々のジョブエージェントおよびリソースエージェントが意思決定を行うことでスケジュールを作成する自律分散型のアプローチとは異なる。

なお、本論文の内容の一部は講演論文集[9]に掲載されているが、その内容は提案手法の基本的なアイデアに留まり、有効性を検証するための実験も行われていない。本論文では、提案手法を厳密に記述するとともに、ケーススタディを行い有効性について検証する。

2. マルチエージェント強化学習を用いた自律分散型スケジューリング

2.1 強化学習

強化学習は、学習主体であるエージェントと、エージェントがおかれた環境との相互作用によって学習が進行する[10]。すなわち、エージェントは、

- (1) 環境から状態 s を観測し、
- (2) 状態 s に基づいて、行動 a を起こし、
- (3) その結果として状態 s' へ遷移するとともに、報酬 r を得る。

以上の三つのサイクルを繰り返しながらタスクを実行する。ここで、(2)において、状態から行動を導き出す規則を方策とよぶ。エージェントは、最適な方策を決定する指標として、状態 s の価値 $V(s)$ 、および／または状態 s における行動 a の価値 $Q(s, a)$ を内部に保持する。

強化学習のアルゴリズムは、現在の状態 s における価値を更新するときに、遷移先の状態 s' における価値に基づいて更新するブートストラップ型、および遷移先の状

態 s' における価値を用いずに更新する非ブートストラップ型に分類される。前者のアルゴリズムとして、TD学習、Q学習、バケツリレーアルゴリズムなどがあり、後者のアルゴリズムとして、Profit Sharing、モンテカルロ法、LMS法などがある。

複数のエージェントが同時に学習を行うマルチエージェント強化学習では、単純マルコフ性を仮定することができない非マルコフ決定過程環境となる場合が多い[11]。ここで、単純マルコフ性とは、時刻 t の状態 s_t と行動 a_t のみから時刻 $t+1$ の状態 s_{t+1} を決定できる性質を示し、単純マルコフ性をもった確率過程を単純マルコフ過程とよぶ。また、単純マルコフ過程を基にした逐次決定過程のことをマルコフ決定過程とよぶ。すなわち、マルコフ決定過程であるということは、将来の状態は、現在の状態とそのときとる行動にのみ依存し、過去の状態や行動には依存しないことを意味する。

マルチエージェント強化学習では、他のエージェントの行動や内部状態を正確に知ることができないという設定が一般的であり、その環境では単純マルコフ性は仮定できなくなる。本研究においても、自律分散型のスケジューリングを対象としており、ジョブエージェントおよびリソースエージェントは、他のエージェントの行動や内部状態を知ることができないため、非マルコフ決定過程環境となる。

以上の点から、本研究では、強化学習のアルゴリズムとして、非ブートストラップ型のモンテカルロ法を用いる。非ブートストラップ型のアルゴリズムは、非マルコフ決定過程環境に対して頑健な特長がある[12]。

モンテカルロ法[11]では、エージェントは、報酬を得た時点から、つぎに新しい報酬を得る時点までの g 個のルール $(s_t, a_t) (t=0, 1, \dots, g-1)$ を保存する。ここで、ルール (s, a) は、状態 s と行動 a の組合せを示す。このとき、価値 $Q(s, a)$ は、以下の式から算出する。

$$SumReward(s_t, a_t) \leftarrow SumReward(s_t, a_t) + r \quad (1)$$

$$Q(s, a) \leftarrow SumReward(s, a) / RewardCount \quad (2)$$

ここで、 $SumReward(s, a)$ および $RewardCount$ は、ルール (s, a) における報酬の累積値、およびルール (s, a) において報酬を獲得した回数である。

2.2 強化学習を用いたスケジューリングプロセス

これまでの研究[3]で提案した滞留時間または加工コストの目的関数をもつジョブエージェントおよび稼働率または加工精度の目的関数をもつリソースエージェントにマルチエージェント強化学習を適用し、つぎの加工プロセスのスケジュールを効用値に基づいて決定する手法について述べる。以下、単にエージェントとよぶ場合は、ジョブエージェントおよびリソースエージェントを示す。

Fig. 1は、各エージェントへのマルチエージェント強

Fig. 1 Application of multi-agent reinforcement learning

強化学習の適用方法を、ジョブエージェントが三つ、リソースエージェントが二つの場合の例におけるガントチャートを用いて示している。ガントチャートの横軸は時間軸を、縦軸はエージェントを、ただし括弧内は目的関数の種類を示している。また、棒図表におけるエージェントの名前は加工の対象を示している。

各エージェントは、以下の五つのSTEPを実行することにより、強化学習に基づいて適切な効用値の算出基準を獲得することができる。

STEP1 時刻 t において待機状態の各エージェントは、効用値に基づくスケジューリングプロセス (1) から (4) を実行する。

- (1) 各エージェントの状態の収集：ジョブエージェントは各リソースエージェントが加工可能な加工プロセスを、リソースエージェントは各ジョブエージェントのまだ実行されていない加工プロセスを求める。また、強化学習に用いる生産システムの状態 s を観測するため、目的関数ごとに待機中のエージェントの数を収集する。
- (2) 候補エージェントの選択：ジョブエージェントは、つぎの加工プロセスを加工することができるリソースエージェントを求める。リソースエージェントは、まだ実行されていない加工プロセスの中で実行することができる加工プロセスを求め、その加工プロセスをもつジョブエージェントを候補とする。
- (3) 効用値の算出：各エージェントは、状態 s に対する行動 a の価値 $Q(s, a)$ に基づいて、候補エージェントの効用値を決定する。また、決定された行動 a に対応するルール (s_t, a_t) を保存する。
- (4) 協調：コーディネーションエージェントは、各エージェントから送られてきた効用値に基づいて、つぎの加工プロセスを実行するための効用値の和が最大となるリソースエージェントとジョブエージェントの組合せを決定する。コーディネーションエー

ジェントには強化学習を適用しないため、生産システムの状態にかかわらず、待機中でつぎの加工プロセスの候補をもつエージェントとその効用値の情報だけに基づいて、つぎの加工プロセスを実行するエージェントの組合せを決定する。

STEP2 時刻 t を進めながら、ジョブエージェントのすべての加工プロセスに対するリソースエージェントの組合せが決定するまでSTEP1のスケジューリングプロセスを実行する。

STEP3 各エージェントは、スケジューリングが終了した時点における目的関数の値に基づいて報酬 r を得る。また、得られた報酬 r を用いて、モンテカルロ法に基づいて価値 $Q(s, a)$ を算出する。すなわち、(1) 式に基づいて保存したルール (s_t, a_t) における報酬の累積値 $SumReward(s_t, a_t)$ に報酬 r を加える。また、ルール (s, a) が獲得した報酬の回数 $RewardCount$ を更新する。つぎに、(2) 式に基づいて価値 $Q(s, a)$ を求める。

STEP4 新しく生産システムに投入されるジョブエージェントに対して、STEP1からSTEP3を実行しスケジューリングを行う。

STEP5 STEP4を繰り返し、各エージェントの状態 s に対する行動 a の実行と報酬 r の算出を繰り返し行うことで価値 $Q(s, a)$ を収束させる。

3. 総納期ずれ最小化のための自律分散型スケジューリング

3.1 自律分散型スケジューリング手法

本研究では、スケジューリングの目的関数を以下の式で示す総納期ずれの最小化とする。

$$\text{minimize } \sum_{i=1}^{\alpha} |DD_i - FT_i| \quad (3)$$

ここで、 α はジョブの総数を表し、 DD_i および FT_i は、ジョブ i における納期および加工完了時間を表す。これまでの研究[3]では、各エージェントが異なる目的関数をもつため、各エージェントがつぎの加工プロセスを実行する候補エージェントを選択した場合の目的関数の評価値の和が最大となるように、コーディネーションエージェントがリソースエージェントとジョブエージェントの組合せを決定した。本研究では、各エージェントの目的関数の最適化ではなく、システム全体の目的関数の最適化のため、著者らの従来の研究[2]で提案したコーディネーションエージェントが存在しない自律分散型のスケジューリング手法を用いる。

本研究における自律分散型のスケジューリング手法は、Fig. 2に示すように、時刻 t において待機状態にあるリソースエージェントおよびジョブエージェントが、つぎの加工プロセスを実行するための組合せを決定する。

はじめに、同図(a)に示すように、すべてのリソースエージェントが、効用値が最大となるジョブエージェン

Fig. 2 Autonomous distributed scheduling method

トを選択する。つぎに、同図(b)に示すように複数のリソースエージェントから選択されたジョブエージェントは、競合を解消するために適切なリソースエージェントを選択する。ここでは、つぎの加工プロセスおよびそのつぎの加工プロセスを連続して加工できるリソースエージェントを選択する。該当するリソースエージェントがない場合はランダムに選択する。以上の手順を繰り返すことで、各エージェントはつぎに実行する加工プロセスを順次決定する。

ここで、リソースエージェント j は、候補ジョブエージェント i に対する効用値 UV_i を、以下の式を用いて算出する。

$$UV_i = \frac{\{\max_{i \in \kappa} S/OPN_i - S/OPN_i\}}{\{\max_{i \in \kappa} S/OPN_i - \min_{i \in \kappa} S/OPN_i\}} \quad (4)$$

ただし、 κ は候補ジョブエージェントの集合を表す。また、 S/OPN_i ($SLACK/Operation Numbers$) は、ジョブ i の納期に対する余裕を表す $SLACK_i$ を、残り工程数 OPN_i で除した値を示し、納期遅れ最小化に有効なディスパッチングルールの一つである。 $SLACK_i$ は、以下の式から求める。

$$SLACK_i = DD_i - RW_i \quad (5)$$

ここで、 RW_i はジョブ i における残りの加工プロセスを実行するために必要な加工時間を表す。

(4) 式を用いることにより、リソースエージェントは、他のジョブと比べて S/OPN_i の値が小さいジョブ、すなわち納期が迫っており、かつ残り工程数が多いジョブに高い効用値を与える。しかし、本研究では、総納期ずれの最小化が目的関数であるため、たとえば、納期までに余裕がある場合は(4)式の効用値の低いジョブを選択する方が、目的関数をよくすると考えられる。そのため本研究では、強化学習を用いることにより、リソースエージェントが生産システムの状態に応じて、(4)式から得られる効用値の修正を行う。

3.2 マルチエージェント強化学習の適用

本研究では、これまでの研究[3]で提案した強化学習を用いたスケジューリングプロセスを改良し、前節に示した自律分散型スケジューリング手法において、リソースエージェントにマルチエージェント強化学習を適用することにより、適切な効用値を算出する基準を獲得する手法を提案する。そのため、第2.2節に示したSTEP1を以下のSTEP1'に変更する。

STEP1' 時刻 t において待機状態の各エージェントは、前節に示した自律分散型スケジューリングの手法を実行し、つぎの加工プロセスを実行するためのリソースエージェントとジョブエージェントの組合せを決定する。ただし、各リソースエージェントは、強化学習に基づいて効用値を算出するために、状態 s に対する行動 a の価値 $Q(s, a)$ に基づいて、候補ジョブエージェントの効用値を算出する。また、決定された行動 a に対応するルール (s_t, a_t) を保存する。

また、STEP3およびSTEP5における学習を行うエージェントを、各エージェントから各リソースエージェントに変更する。STEP2およびSTEP4については、これまでの研究の通りとする。

以上の五つのSTEPを実行することにより、各リソースエージェントは、強化学習に基づいて適切な効用値の算出基準を獲得することができる。

3.3 リソースエージェントの状態・行動・報酬

リソースエージェントが強化学習で用いる状態 s 、行動 a および報酬 r の与え方を以下に示す。

(1) 状態 s

リソースエージェント j が観測する状態を以下の式により与える。

$$s_j = (s_1, s_2, s_3, s_4, s_5, s_6, s_7) \quad (6)$$

ここで、 $s_1 \sim s_6$ は、まだ加工が完了していないジョブエージェント i の納期までの残り時間 $SLACK_i$ の値によって、以下に示す各範囲に含まれるジョブの数 $e_l (l=1, 2, 3, 4, 5, 6)$ に基づいて求める。

$$\begin{aligned} e_1 &: SLACK_i < (1/16)\mu_d \\ e_2 &: (1/16)\mu_d \leq SLACK_i < (1/8)\mu_d \\ e_3 &: (1/8)\mu_d \leq SLACK_i < (1/4)\mu_d \\ e_4 &: (1/4)\mu_d \leq SLACK_i < (1/2)\mu_d \\ e_5 &: (1/2)\mu_d \leq SLACK_i < (3/4)\mu_d \\ e_6 &: (3/4)\mu_d \leq SLACK_i \end{aligned}$$

ここで、 μ_d はすべてのジョブの納期の平均値を示す。求めた $e_l (l=1, 2, 3, 4, 5, 6)$ を、そのまま状態 $s_l (l=1, 2, 3, 4, 5, 6)$ とすることも考えられるが、ジョブ数が多い場合に状態の組合せが非常に多くなるため、計算機実験を行う際に、価値の保存に大量のメモリが必要となる問題がある。そのため、以下に示すように、ジョブの総数 α および $e_l (l=1, 2, 3, 4, 5, 6)$ の値に基づいて0~9に正規化した

値を、状態 $s_l (l=1,2,3,4,5,6)$ の値とする.

$$s_l = 0 \quad (0 \leq e_l < 0.1\alpha)$$

$$s_l = 1 \quad (0.1\alpha \leq e_l < 0.2\alpha)$$

$$s_l = 2 \quad (0.2\alpha \leq e_l < 0.3\alpha)$$

$$s_l = 3 \quad (0.3\alpha \leq e_l < 0.4\alpha)$$

$$s_l = 4 \quad (0.4\alpha \leq e_l < 0.5\alpha)$$

$$s_l = 5 \quad (0.5\alpha \leq e_l < 0.6\alpha)$$

$$s_l = 6 \quad (0.6\alpha \leq e_l < 0.7\alpha)$$

$$s_l = 7 \quad (0.7\alpha \leq e_l < 0.8\alpha)$$

$$s_l = 8 \quad (0.8\alpha \leq e_l < 0.9\alpha)$$

$$s_l = 9 \quad (0.9\alpha \leq e_l \leq \alpha)$$

また、 s_7 はリソースエージェント j の代替機械の数を示す.

したがって、各リソースエージェントは $s_1 \sim s_6$ により、各ジョブの納期に対する残り時間を、納期の平均値で範囲分けした分布として観測する. このとき、納期が迫っているジョブの分布をより細かく観測することが重要と考え、該当する $SLACK_i$ の範囲の幅を狭めている. また、代替機械の数はリソースエージェントの行動に対するスケジュールの影響が大きく関係すると考えられるため状態として s_7 を含めている.

(2) 行動 a

リソースエージェントは、行動 a として、以下の式から定義される行動 1, 2, 3 から一つを選択することにより効用値 UV' を算出する.

$$\text{Action1:} \quad UV' = UV \quad (7)$$

$$\text{Action2:} \quad UV' = 1 - UV \quad (8)$$

$$\text{Action3:} \quad UV' = 4(UV - UV^2) \quad (9)$$

ここで、 UV は (4) 式を用いて求めた効用値を示す.

例として、リソースエージェントが五つの候補ジョブエージェントに与える効用値が、各行動を選択することにより変化する様子を Fig. 3 に示す. 同図からわかるように行動 1 では、(4) 式と同様に S/OPN_i の値が小さいジョブに高い効用値を、逆に、行動 2 では、 S/OPN_i の値が大きいジョブに高い効用値を、また、行動 3 では、 S/OPN_i の値が中央付近のジョブに高い効用値を、それぞれ与えていることがわかる.

本研究では、総納期ずれを目的関数とするため、観測された状態から、納期までの余裕が短いジョブが多い場合かつ代替機械の数が少ない場合は、行動 1 により S/OPN_i の値が小さいジョブ、すなわち納期に余裕がないジョブを選択することを想定し、行動 1 を設定した. 反対に、納期までの余裕が長いジョブが多い場合かつ代替機械の数が多く場合は、納期までの余裕が長いジョブの工程を代替機械を含めて積極的に進めることで、それらのジョブの納期遅れが生じないようにできる可能性があると考え、納期までの余裕が長いジョブに高い効用値を与える行動 2 を設定した. また、納期までの余裕が短

Fig. 3 Example of utility values obtained by action a

いジョブの数および代替機械の数が多くとも少ないともいえない場合は、納期までの余裕が中間的なジョブの工程を進めることで、それらのジョブのすべての工程が終了した際の納期ずれを小さくできる可能性があると考え、納期までの余裕が中間的なジョブに高い効用値を与える行動 3 を設定した.

なお、これまでの研究[3]と同様に、リソースエージェントが価値 $Q(s, a)$ に基づいて行動 a を決定する方法は、 ϵ グリーディ手法[11]を用いる. ϵ グリーディ手法は、確率 $1-\epsilon$ で、一番大きい価値をもつ行動を選択し、確率 ϵ ($0 \leq \epsilon \leq 1$) でランダムな行動を選択する.

(3) 報酬 r

はじめに、報酬 r を算出する基準を作成するために、マルチエージェント強化学習を適用しない場合、すなわち、行動 a を (7) 式に示す行動 1 に限定した場合の基準スケジュールを作成する. リソースエージェント j は、スケジューリングが終了した時点において、以下の式を用いて報酬を算出する.

$$r = (1/\beta) \sum_{j=1}^{\beta} \{(B_j - A_j)/B_j\} \quad (10)$$

$$A_j = \left[\sum_{i=1}^{\alpha} (p_{ij}/u_i) |DD_i - FT_i| \right]_A \quad (11)$$

$$B_j = \left[\sum_{i=1}^{\alpha} (p_{ij}/u_i) |DD_i - FT_i| \right]_B \quad (12)$$

ここで、 β はリソースエージェントの総数を、 A_j および B_j は、強化学習を用いた場合の結果および用いない場合の基準スケジュールの結果から得られる値であり、 p_{ij} および u_i はリソースエージェント j が実行したジョブ i の加工プロセスの数およびジョブ i の加工プロセスの総数を示す.

(11) 式および (12) 式により、各リソースエージェント j が、総納期ずれを求めている. ただし、リソースエー

ジェント j が実行した加工プロセスの数が多いほど総納期ずれの値が大きくなるように、 p_{ij} および u_i を用いて重み付けを行っている。また、(10) 式から各リソースエージェント j が求めた重み付きの総納期ずれについて、強化学習を行わない場合を基準とした改善の割合を求め、リソースエージェント全体の平均値を求めている。これは、これまでの研究[3]によりエージェントの報酬を全体の平均とすることが有効であったためである。

4. ケーススタディ

4.1 実験の条件

提案する手法の有効性を検証するためにケーススタディを行った。ケーススタディの対象として、加工機械が6台、ジョブが30個または100個の加工システムにおいてジョブを繰り返し生産する場合を考える。ただし、加工機械6台のうち、加工機械3と4および加工機械5と6については代替機械とし、同じ加工プロセスを実行することができる。なお、ジョブ30個および100個の場合において、それぞれ加工プロセスおよび納期の異なる三つの条件を作成した。

また、 ϵ グリーディ手法における ϵ は、0.2とした。ただし、生産の繰り返し回数 c が100回ごとに、リソースエージェントは、これまでに学習した価値 $Q(s, a)$ を用いて、最も価値の高い行動 a を選択することとしている。

4.2 総納期ずれの検証

Fig. 4にケーススタディの結果を示す。同図において、横軸および縦軸は、それぞれ生産の繰り返し数 c および改善率 λ を示している。改善率 λ はジョブエージェントの総納期ずれについて、提案手法により得られた値と、学習を行わない手法により得られた値との比率を表し、以下の式から求める。

$$\lambda = \{(X_b - X_a) / X_b\} \times 100 \quad (13)$$

ここで、 X_a および X_b は強化学習を適用することにより得られた目的関数の値および強化学習を適用しない基準スケジュールにより得られた目的関数の値を示す。

Fig. 4から、各条件において収束する値にばらつきがあるものの、ジョブ数が30個の場合および100個の場合、それぞれ学習を行わない場合と比較して、総納期ずれが改善していることがわかる。つまり、強化学習の状態・行動・報酬が適切に機能していると考えられる。しかし、改善率が収束するまでに掛かる繰返し数はジョブ数が100個の場合に比べて30個の場合の方が小さくなっている。これは、生産ジョブ数の増加に伴い、リソースエージェントが観測する状態空間が大きくなることで学習の進行が遅くなるためであると考えられる。

4.3 状態における範囲の分け方の検証

本研究では、状態 s について、(6) 式で定義した通り各ジョブの納期に対する残り時間を、納期の平均値で範囲

(a) Case for 30 jobs

(b) Case for 100 jobs

Fig. 4 Results of case study

Fig. 5 Comparison of status to observe SLACK of jobs

分けした分布として観測している。提案する状態における範囲の分け方が適切であるかを検証するため、以下のように範囲の分け方を等間隔にした場合との比較を行った。

- $e_1 : SLACK_i < (1/6)\mu_d$
- $e_2 : (1/6)\mu_d \leq SLACK_i < (2/6)\mu_d$
- $e_3 : (2/6)\mu_d \leq SLACK_i < (3/6)\mu_d$
- $e_4 : (3/6)\mu_d \leq SLACK_i < (4/6)\mu_d$
- $e_5 : (4/6)\mu_d \leq SLACK_i < (5/6)\mu_d$
- $e_6 : (5/6)\mu_d \leq SLACK_i$

納期が迫っているジョブの分布を細かく観測することができる提案手法、および上記のように範囲の分け方を

等間隔にした比較手法を用いて、ジョブ数が100個の場合において三つの条件の平均値により改善率を比較した結果をFig. 5に示す。同図から、比較手法の範囲の分け方を等間隔にした場合でも、スケジューリングの繰り返しにより総納期ずれが若干改善しているが、提案手法ほどの改善率は得られていないことがわかる。そのため、提案手法のように納期が迫っているジョブをより細かく観測することが、状態として適切であることがわかる。

4.4 価値の分析

ジョブ数が100個の場合の一つの条件において、提案手法により3000回生産を行った後に得られた各リソースエージェントの価値の分析を行った。はじめに、各リソースエージェントにおいて上位30個の高い価値をもつ状態および行動の組合せに関して、ジョブの納期に対する残り時間の分布を表す状態 $s_1 \sim s_6$ を抽出した。リソースエージェントは6台あるため、これにより180個の状態 $s_1 \sim s_6$ が抽出される。つぎに、180個の状態 $s_1 \sim s_6$ のうち、複数のリソースエージェントで共通している状態について、共通しているリソースの台数と、状態の数を求めた。その結果をTable 1に示す。

Table 1の結果から、180個の状態 $s_1 \sim s_6$ のうち、共通するリソースの台数が1台、すなわち、他のリソースエージェントと共通しない状態 $s_1 \sim s_6$ は全体の2割弱の35個だけであった。また、4台以上のリソースエージェントで共通している状態 $s_1 \sim s_6$ は全体の4割強の74個存在することがわかった。このことから、総納期ずれ最小化のために重要となる価値の高い状態 $s_1 \sim s_6$ が、各リソースエージェントで共通していることがわかる。

つぎに、Table 2に示すように、6台および5台のリソースエージェントで共通して高い価値をもつ状態 $s_1 \sim s_6$ について、最も高い価値の行動は行動1, 2, 3のいずれであったかを調べた。本研究では、3.3節において行動を定義する際に、たとえば、納期までの余裕が短いジョブが多い場合は、納期に余裕がないジョブを選択する行動1を取ることを想定している。しかし、Table 2の結果で該当する状態 $s_1 \sim s_6$ として、納期までの余裕が平均納期の1/16以上1/8未満のジョブのみを観測している(0,1,0,0,0,0)の場合に着目すると、行動1に最も高い価値を与えているのは、リソースエージェント2, 3, 6のみで、リソースエージェント4および5は行動2および3に最も高い価値を与えている。

また、すべてのリソースエージェントで共通する状態(0,1,0,1,0,0)の場合に着目すると、納期までの余裕が平均納期の1/16以上から1/8未満のジョブおよび平均納期の1/4以上から1/2未満のジョブを観測しているため、納期までの余裕が短いジョブを選択する行動1または納期までの余裕が中間的なジョブを選択する行動3に高い価値が与えられることを想定しており、リソースエージェント1,2,4,5,6はこれらに相当するが、リソースエー

Table 1 Number of statuses which common to resources

x : Statuses common to x resources	Number of statuses
6	6
5	40
4	28
3	33
2	38
1	35
Total	180

Table 2 Actions corresponding to maximum value

$s_1 \sim s_6$	R.1	R.2	R.3	R.4	R.5	R.6
(0,1,0,1,0,0)	3	1	2	3	1	1
(0,1,0,2,0,0)	2	1	—	1	2	2
(0,1,0,1,1,0)	2	1	—	2	2	1
(0,0,1,1,1,0)	3	1	3	—	1	1
(0,0,1,1,0,0)	—	1	1	2	2	3
(0,0,1,0,0,0)	—	1	2	1	2	3
(0,0,0,2,1,0)	—	1	1	3	1	3
(0,0,0,1,0,0)	—	1	2	1	2	1
(0,1,0,0,0,0)	—	1	1	2	3	1

ジェント3は行動2に最も高い価値を与えている。

以上から、各リソースエージェントは、状態である各ジョブの納期までの残り時間の分布に対して、一律に同じ行動を選択しているのではなく、自身が得られる報酬が最大化するように、スケジュールを作成した結果として総納期ずれが最小化するように、状態に対する行動を選択していることがわかる。言い換えると、学習結果から、ディスパッチングルールのように生産システムの状態に対して、各リソースエージェントの行動を一律に定義するルールを抽出することは困難であることがわかる。

4.5 ロバスト性の検証

動的な変動に対するロバスト性を検証するためのケーススタディも行った。同様の生産システムにおいて、スケジューリングの繰り返し数500回ごとにジョブ数を30から100, 100から30と繰り返し変化させて生産する場合と、ジョブ数が30個のケースにおいて、繰り返し数500回で代替機械をもつ加工機械6に故障が生じ、つぎの500回で修理が完了することを繰り返す場合の2通りを考える。このとき、変化が生じる前までに学習した価値 $Q(s, a)$ を変化後も使用する場合と、変化後に価値 $Q(s, a)$ を初期値に戻して学習をやり直す場合とで比較を行った。Fig. 6にジョブ数が変化する場合、Fig. 7にリソースに故障が生じる場合のケーススタディの結果を示す。

Fig. 6より、生産するジョブ数に変化が生じた場合において、以前の学習情報を利用しない場合に比べて、利

Fig. 6 Results of case for changing number of jobs

Fig. 7 Results of case for breakdown of resources

用した場合の方がそれ以降の改善率が高くなっていることが確認でき、ジョブ数の変化に対するロバスト性があると考えられる。これは、リソースエージェントが観測する状態を、各ジョブの納期に対する残り時間を、納期の平均値で範囲分けした分布としているため、生産するジョブ数が異なる場合でも、スケジューリングの過程で同じ状態を観測する場合があり、その状態における行動の価値が総納期ずれをよくするために有効であったためと考えられる。

Fig. 7より、リソースに故障が生じた場合において、以前の学習情報を利用しない場合に比べて、利用した場合の方がそれ以降の改善率が高くなっていることが確認でき、リソースの故障に対するロバスト性があると考えられる。これは、強化学習における状態の一つとして代替機械の数を観測していることが理由であると考えられる。すなわち、この状態を用いることにより、リソースエージェントの故障が生じた場合に、代替のリソースエージェントが観測する代替機械の数が減ることで、これまでと同じ状態は観測なくなり、代替機械が減った新たな生産システムの状態に対する学習を始める。しかし、故障したリソースエージェントが復旧した場合は、故障前と代替機械の数が同じになることで、これまでに学習した生産システムの状態および行動に対する価値を継続して学習できるため、故障復帰の直後から高い改善率を得ることができていると考えられる。

5. おわりに

本研究では、総納期ずれの最小化を目的として、ジョブエージェントおよびリソースエージェントが効用値に基づいて次の加工プロセスを実行するための組合せを決定する自律分散型のスケジューリング手法を提案した。また、各リソースエージェントにマルチエージェント強化学習を適用し、生産システムの状態に応じて適切な効用値を算出する方法について提案した。

ケーススタディを行った結果、スケジューリングの繰り返し回数の増加に伴い、リソースエージェントが適切な効用値の算出基準を学習することで、総納期ずれをよくできていることを示した。また、ジョブ数の変化やリソースの故障といった生産システムの変化に対して、ロバストな性質をもつことが確認できた。

参考文献

- [1] 森脇, 杉村, Martawirya, 蜷川: 自律分散形生産システムの設計と運用に関する研究: 基本構成と生産スケジューリングの適用; 日本機械学会論文集C編, Vol. 58, No. 549, pp. 1674–1679 (1992)
- [2] 岩村, 桑原, 谷水, 杉村: 人に優しい自律分散型生産システムに関する研究: 作業者の希望を考慮したリアルタイムスケジューリング; システム制御情報学会論文誌, Vol. 22, No. 3, pp. 89–96 (2009)
- [3] 岩村, 眞弓, 谷水, 杉村: 自律分散型リアルタイムスケジューリングへのマルチエージェント強化学習の適用; システム制御情報学会論文誌, Vol. 26, No. 4, pp. 129–137 (2013)
- [4] H. V. Brussel, J. Wyns, P. Valckenaers, L. Bongaerts and P. Peeters: Reference architecture for holonic manufacturing systems: PROSA; *Computers in Industry*, Vol. 37, No. 3, pp. 255–276 (1998)
- [5] K. Ueda, I. Hatono, N. Fujii and J. Vaario: Reinforcement learning approach to biological manufacturing systems; *Annals of the CIRP*, Vol. 49, pp. 343–346 (2000)
- [6] Z. Zhu and R. B. Heady: Minimizing the sum of earliness/tardiness in multi-machine scheduling: a mixed integer programming approach; *Computers & Industrial Engineering*, Vol. 38, No. 2, pp. 297–305 (2000)
- [7] R. Alvarez-Valdes, J. M. Tamarit and F. Villa: Minimizing weighted earliness-tardiness on parallel machines using hybrid metaheuristics; *Computers & Operations Research*, Vol. 54, pp. 1–11 (2015)
- [8] S. Polyakovskiy and R. M'Hallah: A multi-agent system for the weighted earliness tardiness parallel machine problem; *Computers & Operations Research*, Vol. 44, pp. 115–136 (2014)
- [9] 岩村, 横手, 谷水, 杉村: 納期ずれ最小化のためのマルチエージェント強化学習を用いた自律分散型スケジューリング; 日本機械学会生産システム部門研究発表講演会2013講演論文集, pp. 33–34 (2013)
- [10] 電気学会 GA・ニューロを用いた学習法とその応用調査

専門委員会：学習とそのアルゴリズム，森北出版（2002）

- [11] 高玉：マルチエージェント学習-相互作用の謎に迫る-，コロナ社（2003）
- [12] 荒井：マルチエージェント強化学習：実用化に向けての課題・理論・諸技術との融合；人工知能学会誌，Vol. 16, No. 4, pp. 476-481(2001)

著者略歴

いわむら 幸治（正会員）

2000 年大阪府立大学大学院工学研究科博士前期課程機械系専攻機械システム工学分野修了。同年 4 月東京都立航空工業高等専門学校助手，2003 年 4 月大阪府立大学大学院工学研究科助手，2012 年 4 月同准教授，2017 年同大学院人間社会システム科学研究科教授となり，現在に至る。生産工学に関する研究に従事。博士（工学）。2006 年日本機械学会生産システム部門新技術開発表彰受賞。日本機械学会，精密工学会などの会員。

よこて たかゆき
横手 隆幸

2010 年大阪府立大学工学部機械工学科卒業，2012 年同大学院工学研究科博士前期課程機械系専攻機械工学分野修了。同年 4 月株式会社石川島播磨重工業入社，生産技術関連業務に従事，現在に至る。

すがの つばさ
菅野 翼

2015 年大阪府立大学工学部機械工学科卒業，2017 年同大学院工学研究科博士前期課程機械系専攻機械工学分野修了。同年 4 月三菱電機株式会社入社，生産技術関連業務に従事，現在に至る。

すぎむら のぶひろ
杉村 延広（正会員）

1978 年神戸大学大学院工学研究科修士課程修了，1980 年大阪大学大学院工学研究科博士課程中退，博士（工学）。1980 年神戸大学工学部助手，1987 年同助教授，1994 年大阪府立大学工学部教授，現在に至る。日本機械学会奨励賞，工業標準化事業経済産業大臣表彰などを受賞。日本機械学会フェロー，精密工学会フェロー，計測自動制御学会などの会員。