階層型ニューラルネットワークを用いた 動的ジョブショップスケジューリング* (第1報, 納期遅れコスト最小化のための二段階学習法)

江口 透^{*1}, 大場 史憲^{*1}, 豊岡 理^{*2}

Dynamic Job Shop Scheduling Using a Neural Network (1st Report, A Two-Stage Training Method to Minimize Tardiness Cost)

Toru EGUCHI^{*3}, Fuminori OBA and Satoru TOYOOKA^{*3} Graduate School of Engineering, Hiroshima University,
1-4-1 Kagamiyama, Higashi-Hiroshima-shi, Hiroshima, 739-8527 Japan

We consider dynamic job shop scheduling problems with a tardiness cost criterion. The most practical scheduling method in dynamic scheduling environments is using dispatching rules. However, there is no dispatching rule that performs best under all scheduling conditions. In this paper, we apply a multi-layer feed-forward neural network to the scheduling, and propose a two-stage training method to minimize the average tardiness cost of jobs under various scheduling conditions. In the first stage, each neural network for its specified scheduling condition is trained separately by a simulated annealing method. In the second stage, a final neural network for all scheduling conditions is trained by a back-propagation algorithm. Simulation results show that the trained neural network outperforms the best dispatching rules in any levels of due-date tightness and job arrival rate.

Key Words: Scheduling, Neural Network, Production System, Job Shop, Dynamic, Due Date, Tardiness Cost, Simulated Annealing

1. 緒 言

本論文では, 多品種少量生産を行うジョブショップ型の生産システムにおける動的スケジューリング問題について検討する. 動的スケジューリング問題は, ジョブがショップに断続的に到着するもので, ジョブの到着時刻が確率的にしか予測できない. したがって, 現時点での最適なスケジュールを作成しても全体最適性は保証されず, また大規模なジョブショップ問題をリアルタイムで最適化すること自体も困難である. そのため, ディスパッチングルールなどの優先規則によってジョブの処理順序を決定する方法が最も実用的な手法として利用されている. しかし, ディスパッチングルールは試行錯誤的な数値実験によって発見的に求められたもので, その性能は生産条件に依存して変化し, あらゆる条件に対して常に最良の性能を示すルールは存在しない.

本論文では, 納期遅れコスト(重み付き納期遅れ時間)を評価関数とした場合に, 生産条件(ジョブの

到着率と納期のきつさ)の変動に対して優れた性能をもつスケジューリング法を提案し, 数値実験によるディスパッチングルールとの比較によってその有効性を示す. 提案手法では, 階層型ニューラルネットワーク(NN)を用いて, ジョブの処理優先度を決定する.

NNは, 事例からの学習が可能であり, これまでも様々なスケジューリング問題に適用されてきた. その多くは, Sabuncuoglu⁽¹⁾やJainとMeeran⁽²⁾の論文に紹介されている. その他にも, 北岡ら⁽³⁾は, 段ボール生産において, 専門家の経験をNNに学習させることで, 専門家と同等のスケジュールを行うシステムを開発している. また渡部ら⁽⁴⁾⁽⁵⁾は, ジョブショップスケジューリングにおいて, NNを用いることでSLACKルールを用いる方法よりも納期遅れ時間を短縮できることを示している.

NNは, 通常その結合重みを事前に学習によって固定した後にスケジューリング問題に適用されている. よって, その性能は学習法によって左右されることになる. 特に動的スケジューリング問題において納期遅れコスト最小化を目的とする場合には, 全体最適性を指向した学習をすることが非常に重要なポイントである. ジョブ選択の時点で納期遅れ時間の大きいジョブ

* 原稿受付 2001年7月23日.

^{*1} 正員, 広島大学大学院工学研究科(〒739-8527 広島市東山1-4-1).^{*2} 広島大学大学院.

E-mail: eguchi@hiroshima-u.ac.jp

により高い優先度を与えるような局所最適化を指向した学習事例を用いる方法では、全体最適なスケジューリングを行うための NN の重みを探索することはできず、ディスパッチングルールにも劣る NN しか得られないからである。そこで本研究では、局所解からの脱出が可能なシミュレーテッドアニーリング (SA) 法を用いて、動的スケジューリング問題の例題を解きながらスケジュールを全体として最適化する重みを探索する方法を採用する。また、一度に全ての生産条件に対応できる NN を学習することは非常に困難であるため、探索空間を分割して限定された生産条件に対する学習を個別に行った後、最後に全ての生産条件を考慮した統合学習を行うという二段階の学習法を提案する。

2. 動的ジョブショップスケジューリング問題

本研究で対象とする動的スケジューリング問題は次のように記述される。 M 台の機械により構成されるジョブショップを考える。各ジョブ J_i (i はジョブにつけられた番号) は、 n_i 個の工程 $\{o_{ij} \ (j=1, \dots, n_i)\}$ を持ち、工程番号 j の増加順に処理される。ただし、工程順序や処理時間はジョブ毎に異なっているものとする。各工程 o_{ij} の処理時間を p_{ij} とし、これには段取り時間も含まれているものとする。なお、機械は一度に一つのジョブしか処理できず、処理を開始したら途中でその処理が中断されることはない。また、ジョブの搬送時間は無視できるとし、機械の故障はないものとする。

本研究でのスケジューリングの目的は、次式で表されるジョブ 1 個あたりの平均納期遅れコスト TC を最小化することである。

$$TC = \sum_{i=1}^N v_i \times \max(0, C_i - d_i) / N \quad (1)$$

ただし、 C_i, d_i, v_i は、それぞれジョブ J_i の処理完了時刻、納期、納期遅れ単位時間あたりのコストであり、 N は評価対象のジョブ総数である。 $\max(\)$ は括弧内の数値の大きい方の値を示す。ジョブ J_i の納期 d_i は次式で設定されるものとする。

$$d_i = r_i + k_i \times \sum_{j=1}^{n_i} p_{ij} \quad (2)$$

ここで、 k_i は納期のきつさを表す係数であり、 r_i はジョブ J_i のショップへの投入時刻である。なお、各ジョブは平均到着率 λ のポアソン過程でショップに到着した後、直ちに投入されるものとし、全ての処理が完了したら直ちにショップ外に搬出されるものとする。

Fig.1 Structure of the neural network

本論文では、特に、機械台数 M 、工程数 n_i と納期係数 k_i の分布範囲、ジョブの平均到着率 λ が与えられたときの問題を次のように記述する。

$P(M, n [n_l, n_h], k [k_l, k_h], \lambda(\lambda_l, \lambda_h))$ (3)
ただし、 $n [n_l, n_h]$ は、工程数 n_i が n_l から n_h の一様分布 (整数) で与えられることを示し、同じく $k [k_l, k_h]$ は納期係数 k_i が k_l から k_h の一様分布 (実数) で与えられることを示す。また、 $\lambda(\lambda_l, \lambda_h)$ はジョブの平均到着率 λ の変動範囲が λ_l から λ_h までであることを示す。ただし、 λ を固定する場合は、その値を用いて次のように記述するものとする。

$$P(M, n [n_l, n_h], k [k_l, k_h], \lambda) \quad (4)$$

3. 階層型ニューラルネットワークによるスケジューリング

3.1 ニューラルネットワークの構成 本研究で用いる NN の構成を Fig.1 に示す。入力層、中間層、出力層の 3 層から構成されていて、各ユニットの出力関数は出力値が区間(0,1)をとるシグモイド関数である。この NN は、1 つのジョブに対する優先度を出力するためのユニット 1 つを出力層に持ち、ショップの状況や待ちジョブの情報を入力層に入力することで、そのジョブに対する優先度を出力する。待ちジョブが複数ある場合に、この NN をそれぞれのジョブに適用し、最も高い優先度が出力されたジョブが次に処理される。

3.2 ニューラルネットワークの入力情報 高性能な NN を作成するためには有用な情報を NN に入力する必要があるが、どのような情報が有用であるかを事前に特定することは困難である。そこで本研究では、経験的に重要であると思われる情報を入力ユニットとしてすべて列挙するとし、以下に示す 36 個の入力情報を考える。ただし、次の記号を用いる。

t : 現在時刻

j' : ジョブ J_i の現在の (処理待ちの) 工程番号

rpt_i : ジョブ J_i の残り総処理時間。

$$rpt_i = \sum_{j=j'}^{n_i} p_{ij} \quad (5)$$

a_i : ジョブ J_i の納期までの余裕時間.

$$a_i = d_i - t \quad (6)$$

s_i : ジョブ J_i のスラック値.

$$s_i = a_i - rpt_i \quad (7)$$

crt_i : ジョブ J_i のクリティカル値, $const$ は定数.

$$crt_i = s_i - const \quad (8)$$

J_{shop} : 現在ショップ内に存在するジョブ番号の集合

J_{buffer} : 現在ジョブ選択を行う機械の入力バッファ内に存在するジョブ番号の集合

$J_{s<0}$: 現時刻で $s_i < 0$ であるジョブ番号の集合

$J_{s \geq 0}$: 現時刻で $s_i \geq 0$ であるジョブ番号の集合

J_{crt} : 現時刻で $crt_i < 0$ であるジョブ番号の集合

$|X|$: ジョブ集合 X に所属するジョブ数

$abs(x)$: 変数 x の絶対値

①ショップ内の状態に関する情報 (2個)

I_1 : ショップ内の遅れジョブ存在率

$$I_1 = |J_{shop} \cap J_{s<0}| / |J_{shop}| \quad (9)$$

I_2 : ショップ内のクリティカルジョブ存在率

$$I_2 = |J_{shop} \cap J_{crt} \cap J_{s \geq 0}| / |J_{shop}| \quad (10)$$

②機械に関する情報 (6個)

I_3 : 入力バッファ内の待ちジョブ数

$$I_3 = |J_{buffer}| \quad (11)$$

I_4 : 入力バッファ内で待ち状態になっているジョブを全て処理するための時間

$$I_4 = \sum_{i \in J_{buffer}} p_{ij} \quad (12)$$

I_5 : 入力バッファ内の遅れジョブ数

$$I_5 = |J_{buffer} \cap J_{s<0}| \quad (13)$$

I_6 : 入力バッファ内のクリティカルジョブ数

$$I_6 = |J_{buffer} \cap J_{crt} \cap J_{s \geq 0}| \quad (14)$$

I_7 : 入力バッファ内のスラック値 (負の値) の総和

$$I_7 = \sum_{i \in J_{buffer} \cap J_{s<0}} abs(s_i) \quad (15)$$

I_8 : 入力バッファ内のクリティカル値の総和

$$I_8 = \sum_{i \in J_{buffer} \cap J_{crt} \cap J_{s \geq 0}} abs(crt_i) \quad (16)$$

③各ジョブに関する情報 (8個)

I_9 : その工程での処理時間

$$I_9 = p_{ij} \quad (17)$$

I_{10} : 残り総処理時間

$$I_{10} = rpt_i \quad (18)$$

I_{11} : 残り工程数

$$I_{11} = n_i - j' + 1 \quad (19)$$

I_{12} : スラック値 (正の値)

$$I_{12} = \max(s_i, 0) \quad (20)$$

I_{13} : スラック値 (負の値)

$$I_{13} = \min(s_i, 0) \quad (21)$$

I_{14} : 納期余裕 (正の値)

$$I_{14} = \max(a_i, 0) \quad (22)$$

I_{15} : 納期余裕 (負の値)

$$I_{15} = \min(a_i, 0) \quad (23)$$

I_{16} : 納期遅れコスト係数

$$I_{16} = v_i \quad (24)$$

④ジョブの次工程の処理機械情報 (6個)

$I_{17} \sim I_{22}$: ジョブの次工程の機械についての②の情報. 次工程が存在しない場合は, この情報には全て 0 を入力する.

⑤待ちジョブ間の相対情報 (14個)

$I_{23} \sim I_{36}$: ③および④で与えられた待ちジョブの個別情報を待ちジョブ間の最大値で除した選択対象ジョブの相対値. すなわち, 待ちジョブの中での情報 I_k の最大値を $\max_{J_{buffer}}(I_k)$ とすると,

$$I_{k+14} = I_k / \max_{J_{buffer}}(I_k) \quad (k=9, \dots, 22) \quad (25)$$

なお, ②から④の情報についても, 0 から 1 の間に正規化してから入力するものとする. 例えば, バッファ内の待ちジョブ数に関する情報 I_3 はバッファの最大容量に対する比率として入力される.

3.3 二段階学習法 本研究で扱う NN は, 事前にその重みを学習によって決定した後, スケジューリングに適用される. 一般に階層型ニューラルネットワークは, その入出力事例を教師データとして誤差逆伝播法などを用いて学習することができる. しかし, 本研究で扱う動的スケジューリング問題に対しては, スケジューリングの全体最適性を保証する入出力事例 (ショップの状況やジョブの情報と選択優先度の対からなる事例) は事前には与えられない. そこで, 本研究では, 動的スケジューリングの数値シミュレーションを行いながら, 全体として良いスケジューリング性能を示す NN の重みを学習する方法を検討する.

全体最適を追求する最適化法の一つとしてシミュレーティッドアニーリング (SA) 法があり, 様々な最適化問題でその有効性が示されている⁽⁶⁾. 本論文でも, NN の学習に SA 法を用いる. しかし, 種々の生産条件にわたって有効な NN の学習を SA 法によって一度に行おうとすると, 以下の理由により膨大な時間を要するばかりでなく, 優れた性能を得ることも困難である.

(1) 全体最適性を満足するためには, スケジューリング全体でその性能を評価する必要がある. 動的スケジュールでは, 多数のジョブを用いた数値シミュレーションを行い, その定常状態における性能評価をする必要がある. これを繰り返し計算する負荷は大きく, SA 法が解空間の勾配情報を用いない探索法であることも合わせると, 膨大な学習時間が必要となる.

(2) SA 法では, 冷却スケジュールにおける適切な冷却速度を設定することが重要であるが, それは解空間の構造に依存する. したがって, 種々の生産条件を対象とする複雑な解空間においては, その設定が困難であり, 一度に最適解を得ることは難しい.

そこで本研究では, まず, 探索空間を生産条件によって複数の狭い領域に分割し, 探索範囲をそれぞれの領域に限定することで, とりあえず特定の生産条件に対する NN の高性能化と探索時間の短縮を図る. その後, 各生産条件に対する NN を基に, 全ての生産条件に対応できる NN の統合学習を行うという二段階の学習法を提案する. 以下にその方法を示す.

3.4 第一段階の学習 過去のディスパッチングルールの研究から, 動的スケジューリングの性能に与える影響が大きい生産条件は, 納期のきつさと機械稼働率であることが分かっている⁽⁷⁾⁽⁸⁾. 機械稼働率は, ジョブの到着率によって変化する. そこで, 本研究では, 納期のきつさ係数 k_t とジョブの到着率 λ によって, Table1 のように探索空間を分割する. この例では, スケジューリング問題を 9 個の部分問題に分割している.

Table1 中の各部分問題 $P_{\alpha\beta}$ に対して, スケジューリング問題の例題を作成し, SA 法によって NN を学習させる. 学習用の例題を式(4)の表記法を用いて,

$P_{\alpha\beta_training}(M, n [n_l, n_h], k [k_l, k_h], \lambda)$ (26) とする. あるいは, 単に $P_{\alpha\beta_training}$ と記す. 添え字の $training$ は学習用の問題であることを示す. この問題を重みベクトル $w^{(l)}$ を持つ NN でスケジュールした時の平均納期遅れコストを $TC(w^{(l)})$ とすると, SA 法による学習アルゴリズムは以下ようになる.

Table 1 The division of scheduling problem

		Due date tightness k_t		
		Tight	Middle	Loose
Average arrival rate λ	High	P_{11}	P_{12}	P_{13}
	Middle	P_{21}	P_{22}	P_{23}
	Low	P_{31}	P_{32}	P_{33}

Step1. 学習終了回数 L を定め, 学習回数 $l=1$ とする.

初期重み $w^{(1)}$ をランダムに設定する.

Step2. ノイズベクトル ξ をランダムに発生させ,

$\Delta TC = TC(w^{(l)} + \xi) - TC(w^{(l)})$ を求める.

$\Delta TC < 0$ ならば $w^{(l+1)} = w^{(l)} + \xi$ とし Step4 へ.

そうでなければ Step3 へ.

Step3. 受理確率 $Prob = \exp(-\Delta TC / T(l))$ を求める.

ただし, $T(l) = c / \ln(1+l)$ とする. c は定数である.

$U[0,1] < Prob$ ならば $w^{(l+1)} = w^{(l)} + \xi$ とし Step4 へ.

そうでなければ $w^{(l+1)} = w^{(l)}$ とし Step4 へ.

ここで U は括弧内の範囲の一様乱数である.

Step4. $l=L$ なら終了.

そうでなければ, $l=l+1$ とし Step2 へ.

解空間を限定したとしても, 任意の初期重みから出発して有限時間内に良い重みに到達できる保証はない. そこで, 良い重みを見つけるためには, 冷却パラメータ c や初期重みを数回変えて学習を試みる必要がある. それらのなかで最も良い性能を出した重みを最終的な重みとして採用する.

3.5 第二段階の学習 ジョブ選択場面に対して全体最適性を保証する教師事例は事前には与えられないが, いったん第一段階の学習で求めた NN を用いて動的スケジューリング問題 $P_{\alpha\beta_training}$ を解けば, 納期遅れコストが小さい動的スケジュールにおけるジョブ選択事例を多数収集することが可能である. そこで, 生産条件の異なる各部分問題におけるジョブ選択事例をそれぞれ収集し, それらをまとめて統合学習することで最終的な NN を作成する. この統合学習は, スケジュール全体の評価ではなく入出力事例を用いる学習であるため, 誤差逆伝播法を用いることで比較的高速な学習が可能である.

以下に, 第二段階での学習法を示す.

Step1 (教師事例の収集)

部分問題 $P_{\alpha\beta}$ について, 第一段階の学習で求めた NN を用いて動的スケジューリング問題 $P_{\alpha\beta_training}$ を解き, 待ちジョブが 2 個以上ある時のジョブ選択事例 $S_q(I, \bar{u}, \bar{k}, p)$ ($q=1, \dots, Q$) を収集する. ここで, I はジョブ選択を行う時点での入力情報の値からなるベクトルで, $I = (I_1, I_2, \dots, I_g)$ である. g は第

Fig.2 Overview of the two-stage training method
(n_s is the number of sub-problems for training)

一段階で学習する NN の入力ユニットの個数である。また、 p はジョブの優先度を表し、次に処理されるジョブとして選択された場合は 1、選択されなかった場合は 0 を与える。 Q は収集したジョブ選択事例の個数である。 $\bar{u}$ は現時点における平均機械稼働率、 $\bar{k}$ はショップ内に存在するジョブの納期係数 k_i の平均値であり、部分問題の特性を識別するための情報として統合学習時に新たに追加する入力ユニットの情報である。

ここで、 $\bar{u}$ は次式で与えられる。

$$\bar{u} = \sum_{t'=t-\Delta t}^t \sum_{m=1}^M u_m(t') / M / \Delta t \quad (27)$$

ただし、 $u_m(t) = \begin{cases} 1: \text{時刻 } t \text{ で機械 } m \text{ が稼働中} \\ 0: \text{時刻 } t \text{ で機械 } m \text{ が遊休中} \end{cases}$

Δt : 稼働率を計算する時間幅

また、 $\bar{k}$ は次式で与えられる。

$$\bar{k} = \sum_{i \in J_{shop}} k_i / |J_{shop}| \quad (28)$$

Step2 (教師事例の圧縮)

膨大なジョブ選択事例の中から典型的な事例を抽出し学習の効率化を図る。まず、各入力ユニットの入力範囲を N_f 区間に分割して入出力事例进行分类する。入力ユニットの総数は g 個であるので、ジョブ選択事例は最大 $(N_f)^g$ 個のカテゴリーに分類される。その後、一つのカテゴリーに N_m 個以上の事例がある事例だけを取り出す。取り出された事例について、各入力値および出力値をそのカテゴリーに入った事例の平均値で代表する。すなわち、一つのカテゴリーに入った複数個の事例は、一個の事例に集約され

る。この方法で圧縮された教師事例を $S'_q(I, \bar{u}, \bar{k}, p)$ ($q=1, \dots, Q'$) とする。さらに、この中からランダムに Q'' 個の事例 $S''_q(I, \bar{u}, \bar{k}, p)$ ($q=1, \dots, Q''$) を選択する。

Step3 全ての部分問題について教師事例を収集したら Step4 へ行く。そうでなければ Step1 へ行く。

Step4 (統合学習)

全ての部分問題に対する事例 $S''_q(I, \bar{u}, \bar{k}, p)$ をまとめて最終的な教師事例とし、新たな NN を誤差逆伝播法 (慣性項を付加) を用いて学習する。この時の NN の入力ユニット数は、第一段階の入力ユニットに $\bar{u}$ と $\bar{k}$ を加えた $(g+2)$ 個である。

Fig.2 に二段階学習法の概略図を示す。

4. 数値実験

提案手法の有効性を示すために数値実験を行った。スケジュール対象を式(3)の記述に従って次のような条件を持つジョブショップとする。

$$P(16, n [8, 16], k [2, 5], \lambda(1/55, 1/45)) \quad (29)$$

各ジョブの処理時間 p_i は $[5, 100]$ の一様分布、単位時間あたりの納期遅れコスト v_i は $[1, 5]$ の一様分布で与える。また、機械は処理待ちのジョブを待機させておく入力バッファを有しており、その容量はジョブ 3 個に制限されている。処理終了後に次工程の入力バッファが一杯である場合は、中央バッファに搬送され待機する。中央バッファからの搬送ジョブの選択は、 $W(CR+SPT)$ ルール⁽⁷⁾に従うものとした。なお、本研究では遅れなしスケジュールのみを考慮する。即ち、機械が遊休状態になった時、待ちジョブが一つであればそのジョブの処理を直ちに開始する。二つ以上ある場

合は、それらの優先度に従ってジョブを選択し直ちに処理を開始する。

NNの入力層のユニット数は、第一段階の学習時は36個で、最終的には38個となる。中間層のユニット数は、予備実験の結果、10個に固定した。また、クリティカルジョブの定数は $const=100$ 、稼働率計算の時間幅は $\Delta t=100$ とした。

4.1 第一段階の学習 Table1の生産条件に従って問題 P を次のような9つの部分問題に分割する。

$$\begin{aligned} P_{11}(16, n [8, 16], k [2, 3], 1/45) \\ P_{12}(16, n [8, 16], k [3, 4], 1/45) \\ P_{13}(16, n [8, 16], k [4, 5], 1/45) \\ P_{21}(16, n [8, 16], k [2, 3], 1/50) \\ P_{22}(16, n [8, 16], k [3, 4], 1/50) \\ P_{23}(16, n [8, 16], k [4, 5], 1/50) \\ P_{31}(16, n [8, 16], k [2, 3], 1/55) \\ P_{32}(16, n [8, 16], k [3, 4], 1/55) \\ P_{33}(16, n [8, 16], k [4, 5], 1/55) \end{aligned} \quad \dots (30)$$

次に、各部分問題に対して学習に用いるスケジューリング問題を作成する。ただし学習時間短縮のため、第一段階の学習に対しては、小規模の問題を学習用問題とする。また、全ての生産条件について学習するのではなく、 P_{11} , P_{13} , P_{22} , P_{31} , P_{33} の5つに対応する条件についてのみ学習するものとする。すなわち、第一段階の学習に用いる問題は次のとおりである。

$$\begin{aligned} P_{11_training}(4, n [2, 4], k [2, 3], 1/45) \\ P_{13_training}(4, n [2, 4], k [4, 5], 1/45) \\ P_{22_training}(4, n [2, 4], k [3, 4], 1/50) \\ P_{31_training}(4, n [2, 4], k [2, 3], 1/55) \\ P_{33_training}(4, n [2, 4], k [4, 5], 1/55) \end{aligned} \quad \dots (31)$$

ただし、工程数はそれぞれの部分問題における機械稼働率が変化しないように機械台数に応じて縮小している。

各部分問題に対してジョブ数 1000 からなる問題 $P_{\alpha\beta_training}$ を 10 題ずつ作成し、その 10 題に対する平均納期遅れコスト TC の平均値を $TC(w^{(i)})$ として、 $TC(w^{(i)})$ を最小化するように SA 法による学習を行った。なお、NN の初期重み $w^{(0)}$ とノイズベクトル ξ はともに一様乱数 $[-0.1, 0.1]$ で決めた。冷却パラメータを $c=25$ とした時の部分問題 P_{22} に対する学習曲線の一例を Fig3 に示す。この学習一回に要した時間は約 6 時間であった (本実験で使用した CPU は 1 GHz AMD Athlon)。このような学習を各部分

Fig.3 A training curve using the SA

Fig.4 The training curve using a BP

問題に対して冷却パラメータ c や重みの初期値を数通り変えて試行し、最も良い性能を示す重みを採用した。学習時間は試行回数に比例して増加するが、いずれの部分問題に対しても 10 回以内の試行で有効な NN を学習することができた。

4.2 第二段階の学習 第二段階では、まず、第一段階で得られた重みを用いてスケジューリングシミュレーションを行うことによって、それぞれの部分問題における NN の入出力事例 (教師事例) を収集する。第一段階では計算時間を短縮するため規模を縮小した問題 (式(31)) を用いて学習を行ったが、第二段階の学習は比較的短時間で実行できるため、最終目的の問題 (式(29)) と同規模の問題を用いる。すなわち、次の問題を新たに作成した。

$$\begin{aligned} P_{11_training}(16, n [8, 16], k [2, 3], 1/45) \\ P_{13_training}(16, n [8, 16], k [4, 5], 1/45) \\ P_{22_training}(16, n [8, 16], k [3, 4], 1/50) \\ P_{31_training}(16, n [8, 16], k [2, 3], 1/55) \\ P_{33_training}(16, n [8, 16], k [4, 5], 1/55) \end{aligned} \quad \dots (32)$$

これらの問題に対して、第一段階で学習した NN によるスケジューリングを行い、ジョブ選択事例を各部分問題について $Q=20$ 万個ずつ収集した。教師事例の圧縮ステップにおいて、 $N_f=5$, $N_m=3$ として圧縮された教師事例を作成し、さらに各部分問題に対して $Q''=2000$ 個ずつランダムに選び出した。それらを教師事例として誤差逆伝播法 (慣性項を付加) によって統合学習を行った時の学習曲線を Fig4 に示す。この学

Table2 Scheduling results (average tardiness cost and machine utilization)

		Due date tightness k_i					
		2.0~3.0 (Tight)		3.0~4.0 (Middle)		4.0~5.0 (Loose)	
Average arrival rate λ	1/45 (High)	ATC	1739.35	ATC	974.48	ATC	566.73
		W (CR+SPT)	1777.00	W (CR+SPT)	920.79	W (CR+SPT)	437.45
		W (S/RPT+SPT)	1728.66	W (S/RPT+SPT)	892.09	W (S/RPT+SPT)	433.41
		NN	1647.09	NN	779.33	NN	361.45
		Machine utilization	88.6%	Machine utilization	88.6%	Machine utilization	88.5%
	1/50 (Middle)	ATC	524.12	ATC	130.42	ATC	19.77
		W (CR+SPT)	510.40	W (CR+SPT)	131.72	W (CR+SPT)	22.33
		W (S/RPT+SPT)	519.47	W (S/RPT+SPT)	138.47	W (S/RPT+SPT)	24.11
		NN	475.83	NN	95.21	NN	12.65
		Machine utilization	79.5%	Machine utilization	79.5%	Machine utilization	79.7%
	1/55 (Low)	ATC	195.40	ATC	14.41	ATC	0.87
		W (CR+SPT)	178.25	W (CR+SPT)	21.09	W (CR+SPT)	1.93
		W (S/RPT+SPT)	193.11	W (S/RPT+SPT)	23.75	W (S/RPT+SPT)	1.73
		NN	163.13	NN	13.39	NN	0.99
		Machine utilization	72.3%	Machine utilization	72.1%	Machine utilization	72.4%

習に要する時間は約 10 分であった。

4.3 性能評価 性能評価のため、学習時に用いたスケジューリング問題とは別の問題を各部分問題 (P_{11} , P_{12} , P_{13} , P_{21} , P_{22} , P_{23} , P_{31} , P_{32} , P_{33}) について 100 題ずつ作成した。ジョブ数は 1 題につき 4000 個とした。納期遅れコスト最小化のための代表的な 3 つのディスパッチングルール ATC⁽⁸⁾, W (CR+SPT)⁽⁷⁾, W (S/RPT+SPT)⁽⁷⁾ を用いてこの問題をスケジューリングしたときの平均納期遅れコストと最終的に統合学習された NN を適用した場合の平均納期遅れコストを Table2 に示す (いずれも 100 題の平均値)。これらは、投入時刻の早い 1000 個のジョブと完了時刻の遅い 1000 個のジョブを除外して、定常状態と判断される数値実験中盤の 2000 個のジョブに対する結果を示している。また、その 2000 個のジョブがジョブショップを通過しているときの平均機械稼働率も示している。最良のディスパッチングルールは生産条件によって変動するが、NN は生産条件が最もゆるい場合 P_{33} を除いた全ての生産条件で最良のディスパッチングルールよりも優れていることが分かる。部分問題 P_{33} では、どの方法を用いても納期遅れコストの値が非常に小さく、性能差が生じない特殊な条件である。

Table3 は部分問題ごとに最良のディスパッチングルール (上段) とそれに対する NN の改善率 (下段, 太字) を示している。ただし,

$$\text{改善率}(P_{\alpha\beta})\% = (\text{best_disp}_{\alpha\beta} - \text{NN}) / \text{best_disp}_{\alpha\beta} \times 100$$

$\text{best_disp}_{\alpha\beta}$: 問題 $P_{\alpha\beta}$ において最良ディスパッチングルールを用いたときの平均納期遅れコスト (100 題の平均値)

NN: NN を用いたときの平均納期遅れコスト (100 題の平均値)

である。また、○は paired-t テスト⁽⁹⁾による統計的検定の結果、NN と最良ディスパッチングルールとが有意水準 1% で差があると認められたことを示す。△は有意差が無いと判定されたことを示す。部分問題 P_{33} では最良のディスパッチングルールとの有意差はなかったが、それ以外の生産条件については最良のディスパッチングルールよりも良いスケジュールを作成できることが確認された。これらより、比較に用いた 3 つのディスパッチングルールは生産条件によってその優劣が変化するのに対して、NN を用いる方法では、納期遅れがほとんど生じない特別な条件を除いて、常に最良のディスパッチングルールよりも優れた性能をもつロバスタなスケジューリングを行えることが確認された。

Table3 の括弧 () 内には、式(31)の学習用問題を同時に用いて、SA 法のみによって一度に NN を学習させた場合の結果の一例を示す。ただし、SA 法における学習では、重み $w^{(i)}$ に対するスケジュールの評価値を次式で与えた。

$$TC(w^{(i)}) = (TC(w^{(i)})_{P11_training} + TC(w^{(i)})_{P13_training} + TC(w^{(i)})_{P22_training} + TC(w^{(i)})_{P31_training} + TC(w^{(i)})_{P33_training}) / 5.0 \quad \dots (33)$$

ただし、 $TC(w^{(i)})_{P_{\alpha\beta_training}}$ は、部分問題 $P_{\alpha\beta}$ の学習用問題に対する平均納期遅れコスト (10 題の平均値) を意味する。なお、このように一括して学習する場合には、SA 法による学習の段階から機械稼働

率 $\bar{u}$ と納期のきつさ k の情報を NN の入力ユニットに加えている。

式(33)を用いる方法で、Table3 に示したように生産条件がゆるい問題に対して性能が劣る結果が得られたのは、式(33)中の $TC(w^{(i)})_{P\alpha\beta_training}$ の値が大きくなるような生産条件、すなわちジョブの到着率が高く納期が厳しい条件がより重視されたためと考えられる。そこで、同じく SA 法のみによる学習で、学習時の評価値を

$$TC(w^{(i)}) = (TC(w^{(i)})_{P11_training} / best_disp_{11_training} \\ + TC(w^{(i)})_{P13_training} / best_disp_{13_training} \\ + TC(w^{(i)})_{P22_training} / best_disp_{22_training} \\ + TC(w^{(i)})_{P31_training} / best_disp_{31_training} \\ + TC(w^{(i)})_{P33_training} / best_disp_{33_training}) / 5.0 \quad \cdots (34)$$

とした場合の典型的な結果を Table3 の括弧[] 内に示す。ただし、 $best_disp_{\alpha\beta_training}$ は式(31)の各学習用問題 $P_{\alpha\beta_training}$ に対して最も良い性能を示したディスパッチングルールを用いたときの平均納期遅れコスト(10 題の平均値)であり、式(34)はそれらに対する相対値で学習時の性能を評価することを意味する。この方法では、生産条件が厳しい場合にディスパッチングルールに比べて劣ることが多かった。

Table3 の×は、NN が最良のディスパッチングルールよりも劣り、有意水準 1%で差があると認められたことを示す。式(33)および式(34)の評価式を用いて一括して NN を学習する方法では、SA 法の冷却速度や重みの初期値を変えた数十回の試行の結果、全ての生産条件でディスパッチングルールと同等かそれ以上の性能を持つ NN を得ることは一度もできなかった。その際の計算時間は、5 つの部分問題を同時に学習するため、一回の試行について約 30 時間であった。これらから、提案した二段階学習法によって、より短い学習時間で性能の良い NN が学習できるといえる。

5. 結 言

本論文では、動的ジョブショップスケジューリングを行うための NN の学習法を提案し、生産条件(ジョブの到着率と納期のきつさ)の変動に対して優れたスケジューリングを行えることを数値実験によって示した。提案手法は、各生産条件に対する NN の重みを SA 法で個別に探索した後、得られた

Table3 Improvement rate for each scheduling condition

		Due date tightness k_i		
		2.0~3.0	3.0~4.0	4.0~5.0
Average arrival rate λ	1/45	W(SRPT+SPT) ○4.72% (○3.70%) [×-15.61%]	W(SRPT+SPT) ○12.64% (○9.52%) [△0.35%]	W(SRPT+SPT) ○16.60% (○13.50%) [○19.19%]
	1/50	W(CR+SPT) ○6.77% (○2.43%) [×-7.99%]	ATC ○26.99% (○9.13%) [○31.04%]	ATC ○36.05% (×-30.02%) [○38.64%]
	1/55	W(CR+SPT) ○8.49% (△0.90%) [○10.44%]	ATC ○7.12% (×-49.78%) [○34.48%]	ATC △-12.96% (×-267.11%) [△15.61%]

NN を用いた動的スケジュールから教師事例を収集し、それらを基に統合学習を行うという二段階の学習法である。

ディスパッチングルールに比べて改善できる性能の程度は生産条件によって異なり、それらは事前には分からない。このような状況で、各生産条件間のバランスを取りながら、なおかつ局所解に陥ることなく、一度に NN の重みを探索することは極めて困難である。提案手法の利点は、この困難な問題の特性を分解して取り扱うことで、ロバストな NN を段階的に作成していくことができることである。なお、新たに他の生産条件も考慮する場合には、既に学習済みの部分問題に対する第一段階の学習は省略することができる。よって、よりロバストな NN を構築していく場合にも、過去の学習結果を蓄積し有効利用することができる方法であるといえる。

参考文献

- (1) Sabunchoghlu, I., *Production Planning & Control*, 9(1998), 2
- (2) Jain, A.S. and Meeran, S., *INT.J.PROD.RES.*, 36(1998), 1249.
- (3) 北岡・他 2 名, 経営システム, 1-1(1991), 72
- (4) 橋本・渡部・他 2 名, 電気学会論文誌 C, 115-5(1995), 744
- (5) Fujii, T. and Watanabe, T., *Proc.2000 Japan-USA Flexible Automation Conference*, (2000), CD-ROM
- (6) Koukamas, C.・他 2 名, *Omega*, 22-1(1996), 41
- (7) Anderson, E.J. and Nyirenda, J.C., *INT.J.PROD.RES.*, 28(1990), 2277
- (8) Vepsäläinen, A.P.J. and Morton, T.E., *Management Science*, 33(1987), 1035
- (9) Law, A.M. and Kelton, W.D., *Simulation Modeling & analysis*, 2nd Edition, McGraw-Hill, Inc., (1991)