修 士 論 文

フローショップ・スケジューリング問題への強化学習の適用

指導教官　平石邦彦 助教授

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

田中雄介

1999 年 8 月 13 日

目 次

第 1章

序論

1.1 背景

生産スケジューリング問題は, 計算の複雑さの理論でしばしば扱われることからわかるとおり, 一部の簡単なものを除けば, ほとんどの問題は「本質的に」難しい [3]. しかし, ビジネス上, よりよいスケジュールの入手は, 競合他社との差別化を図れる大きなアドバンテージとなる. 例えば, 近年盛んに提唱されるサプライチェーン・マネジメントでは, 企業内から企業をまたぐ範囲まで, より広い範囲でオペレーション・コスト等の最適化を実現しようとしている. その中において, 生産計画の重要性は, ますます増している. このように, 経営の要請がより高度になっていることを受けて, 言い換えれば, より大規模で複雑な生産スケジューリングを行うために, 従来とは異なるアプローチで生産スケジューリング問題に取り組む機運も高まりつつある.

近年生産スケジューリング問題に対する有効性の報告のある, 強化学習アプローチもそのひとつである. Zhang and Dietterrich[13] は, NASA のスペースシャトル貨物処理スケジューリングにおいて, 強化学習を適用した. このスケジューリング問題は, ジョブショップ・スケジューリングに関して, ジョブの総処理時間最小化問題になる. 従来は, シミュレーテッド・アニーリング (焼きなまし法) ベースのスケジューリング法で, Iterative Repair 法があった. 強化学習アプローチの成績は, IR 法のそれを上回ったことが, 報告されている. このように, 同問題に対する新たなアプローチとして, 強化学習は有望である.

しかし, 機械学習においては, 学習時の訓練量が大きすぎて, 大規模で複雑な問題に対処できないことが, しばしば問題になる. 前掲の研究 [13] では, 効率的な学習を行うために

いくつかの工夫が講じられている. その中でも特に, スケジューリング問題を強化学習問題として定式化する際, 種々のヒューリスティクスを参考にして, 探索空間の大幅な削減を図った工夫が注目される [6]. つまり, 当該研究では, 定式化において問題領域に特化したことが成功要因の 1 つである. しかも, 定式化にあたっての指針と呼べるような定石は現時点では確立されていないといえる. 前述研究 [13] でも, 定式化に工夫の余地が残ることを認めている.

1.2 目的

本研究では, 以上の問題意識を背景に, 強化学習の生産スケジューリング問題に対する, 適用可能性の詳細な考察を行う. そのために, ジョブショップ・スケジューリングへの拡張可能性を考慮しつつ, フローショップ・スケジューリングに関して, ジョブの総処理時間最小化問題 ($F||C_{max}$ 問題と表記する) を考える.

ここで, 上記問題を対象にする理由を説明する. フローショップとは, 全てのジョブが, 全ての機械で処理され, かつ同じ工程順で処理される環境をいう. つまり, 機械間でのジョブの流れが同一方向になる. これに対して, ジョブショップでは, 機械間でのジョブの流れがばらばらで, 全ての機械を通らずに処理が完了するジョブがある. 従って, フローショップは, ジョブショップの特殊な場合である.

$F||C_{max}$ 問題の内,2 機械問題 ($F_2||C_{max}$ 問題と表記する) の場合は, ジョンソン則による最適解獲得が証明されている [2]. しかし, 3 機械以上の場合, 総処理時間を最小化するスケジューリング方策は, 小規模な問題を除いては, 計算量の観点から求めるのが困難である. つまり, 全数探索を行い, 各ジョブの仕掛順を総処理時間で評価する以外, 最適解は求められない. (このように, 全数探索を除いて最適解が求められないことを, 本論文では「理論解が存在しない」と呼ぶことにする.)

以上のように,$F||C_{max}$ 問題を対象にして, 強化学習問題としての定式化が考察されることは, 強化学習の適用可能性を詳細にみるには有利である. その理由のひとつには, 最適化則を参考にした定式化が可能になる点が挙げられる. さらに, その定式化を土台にして, 理論解が存在しない問題領域への有効性も, 評価することができる.

以上をふまえて, 本研究における主な目的は,

- いまだ理論解のない本問題に対し, 強化学習問題としての定式化を考案する.
- 実装評価により, その有効性を吟味し, 強化学習の適用可能性を評価考察する.

再度指摘するが, 学習が効果を挙げる鍵は, 強化学習問題としての定式化にある. まず, 強化学習問題としての定式化を考察する. そこでは, $F_2||C_{max}$ 問題について考察してから, $F_3||C_{max}$ 問題での定式化を提案する. さらに, 遺伝アルゴリズムによる優良解を学習させる強化学習も提案する. 以上の定式化に基づく実装を行い, 強化学習の適用可能性が確認できることを報告する.

第 2 章

強化学習

本章では, 強化学習の枠組みについて, Sutton and Barto[11] に依拠した説明をする.

2.1 強化学習の 5 要素

強化学習の構成要素は, 環境 (environment), エージェント (agent), 政策 (policy), 報酬 (reward), 価値関数 (value function) の 5 つに分けられる. 環境は, 問題領域を定義する. エージェントは, 問題に対して意思決定や学習を行う. 政策は, エージェントが知覚する, ある特定の環境の状態に対して, エージェントがとれる行為の中から特定の行為を決定するものである. つまり, 状態を入力とし, 行為を出力とする関数である. 報酬は, エージェントが政策に基づき行為を意思決定し実行した結果, 環境がその行為に対して与える評価である. 最後に, エージェントが獲得した報酬は, 次の機会の意思決定に活かされるために, 価値という概念に変換される. 価値関数は, それを記憶しておく領域である. なお, 価値とは, ある状態から最終状態までの累積報酬である.

エージェントの最終目標は, 初期状態から終了状態までの累積報酬を最大化する政策を獲得することにある.

図 2.1 において, 強化学習のプロセスを説明する. エージェントは, 現在の状態のみを入力として, 政策に基づき行為を決定する. (政策には, 価値関数を利用する場合と, 利用しない場合がある. 次節詳述) その行為を受けて, 環境は変化し (次の状態へ遷移し), エージェントに報酬を知らせる. 状態遷移や報酬は確率的であってもよい. エージェントは受け

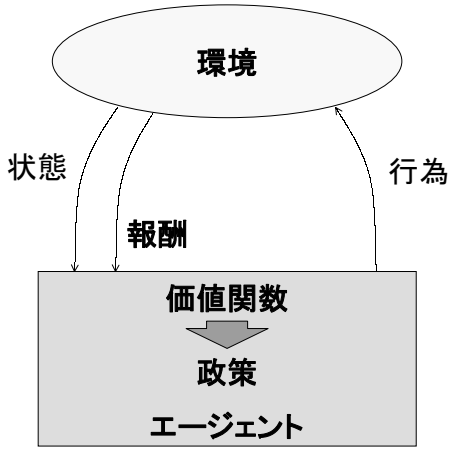

図 2.1: 強化学習の 5 要素

取った報酬を, 経験した結果として価値関数に記憶する. 以上のプロセスを繰り返しながら, エージェントは, 自身の目標である, 初期状態から終了状態までの累積報酬を最大化する政策の獲得を目指す.

2.2 政策

エージェントの行為選択戦略には 2 タイプある. ひとつは, 報酬を最大化しようとしてある状態で選択可能な可能な行為それぞれについて次状態探索を行い, 報酬と次状態価値の和が最大になるものを選択するもの (exploitation) である. もうひとつは, いろいろな状態と行為を試すために, 選択可能な行為をランダムに選ぶもの (exploration) である. 前者では価値関数を利用するが, 後者では利用しない. ここでは, 文献 [1] に従い, 前者の選択による行為を報酬獲得行為, 後者を情報獲得行為と呼ぶ.

報酬獲得行為を多くとることは, 報酬を多く獲得しようとする意味ではプラスだが, 経験の幅が狭くなってしまい真の報酬最大化政策が学習できない可能性が大きくなる意味ではマイナスである. 情報獲得行為を多くとることは, 経験を多く積む意味でプラスだが, 報酬が得られない良くない経験を積むこともある意味ではマイナスである. このように, 情報獲得と報酬獲得の間には, トレード・オフ関係がある.

本研究では,ϵ-greedy policy と呼ばれる政策の枠組みを採った. そこでは, エージェントは学習の初期において情報獲得行為をとり, 終期には報酬獲得行為をとる. 即ち, 情報獲

得行為をとる確率 ϵ (probability of random action in ϵ-greedy policy) を学習期間を通じて減少させる. そのようにすることで, 初期に大域的な探索を十分に実施し, その経験を踏まえたうえで, 最終的に貪欲な報酬獲得に専念することになる. つまり, 静的な環境ではトレード・オフ関係が解消され, より良い学習結果が得られる.

2.3 学習アルゴリズム (価値関数の更新)

本研究では, 価値関数の更新に, $TD(\lambda)$ 法を用いた. 時刻 $t \in \{0, 1, ..., T\}$ における状態を s_t とし, 状態 s_t の関数としての価値は, $V(s_t)$ と表記する. また, 時刻 t における全状態の価値は,$V_t(s)$ と表記する. 個別の状態に対する価値は, 例えば, 時刻 $t+1$ における状態 s_{t+1}, の時刻 t における価値は,$V_t(s_{t+1})$ などと表記する. 時刻 t にある行為 a_t を選択して, 報酬 r_{t+1} を得て次状態 s_{t+1} に遷移したときに, 全状態の推定価値は, 次式により更新される. α は学習率 (step-size parameter), γ は割引率 (discount-rate parameter), λ は適確波及係数 (decay-rate parameter for eligibility traces) である.

$$V_{t+1}(s) = V_t(s) + \alpha[r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)]e_t(s)$$

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) + 1 & (s = s_t) \\ \gamma\lambda e_{t-1}(s) & (s \neq s_t) \end{cases}$$

$$(0 \leq \alpha, \gamma, \lambda \leq 1)$$

定義上, 遷移元状態の価値 $V(s_t)$ は, 報酬と遷移先状態の価値の合計 $r_{t+1} + V(s_{t+1})$ である. 学習するとき, 実績値たる報酬を含んでいる, 後者の合計値の方を, より正しい推定値とし, その値に向かって修正する. $[r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)]$ を Temporal Difference といい, この学習法の名前の由来になっている.

学習率 α, 割引率 γ, 適確波及係数 λ は, 問題領域毎で設定値は異なり, 理論的に正しい設定値を決めることは難しい. 学習率 α は, 理論的には収束条件として, 各状態の訪問回数の逆数のような減少関数とする条件がある. しかし, 実装においては学習率を可変とせずに, かなり小さい値で一定とすることが多い. 割引率 γ は, 最終状態がない問題に対して, 累積報酬が無限大にならないようにするためのパラメータである. 適確波及係数 λ は, ある時

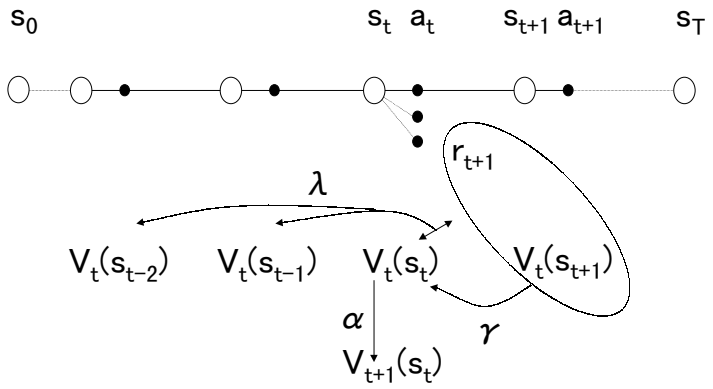

図 2.2: $TD(\lambda)$ 法による価値関数の更新

点で観測された TD を, それ以前に経験した状態の価値の更新にも使うことで, 学習を速めようとするパラメータである. かなり大きい値 (0.8 以上ただし 1 以外) でよい成績を上げることが多い.

第 3章

強化学習問題としての定式化

3.1　2機械問題

2 機械フローショップ・スケジューリングに関して, ジョブの総処理時間を最小化する解法であるジョンソン則を説明する. 2 機械 M_1, M_2 の順に処理される n 個のジョブ $j_i \in J(i = 1, 2, ..., n)$ は, 解く前に予め全て与えられているとする (これを静的スケジューリング問題と呼ぶ). また, 機械 M_1, M_2 各々でのジョブ i の処理時間を p_{1i}, p_{2i} とする. 2 ジョブ $j_i, j_k (i, k = 1, 2, ..., n; i \neq k)$ について, ジョブ j_i が次の不等式を満たすならば, ジョブ j_k に先行して処理するルールが, ジョンソン則である [2].

$$min[p_{1i}, p_{2k}] \leq min[p_{1k}, p_{2i}] \quad \text{(等号成立時の仕掛順は任意)}$$

本節では, この問題に対して, 次節の 3 機械問題への拡張を意識し, 強化学習問題としての定式化について考察する.

3.1.1　環境とエージェント

ここでは, 前章での強化学習の 5 構成要素の順に定式化していく. まず, 環境は, スケジューリングの対象になる機械やジョブ集合である. 強化学習をするエージェントは, M_1 前のジョブの待ち行列の仕掛順のみを決定することとした. M_2 前の待ち行列に対しては, 到着順の処理でよい. フローショップの 2 機械ないし 3 機械の静的スケジューリング問題でかつ, 総処理時間最小化では, M_2 前の待ち行列に対し, M_1 前の仕掛順と異なる仕掛順

で解く必要がないからである [2].

次に, スケジューリングでのジョブ仕掛順の意思決定が, 強化学習における逐次意思決定へどのように適用されるかが焦点になる. 文献 [13] では, ジョブの順序制約のみを考慮し, 資源制約が違反した状態のスケジュール (実行不能なスケジュールと呼ぶ) から, ジョブの移動や資源割当の変更を行いつつ, 資源制約違反が解消したスケジュール (実行可能なスケジュールと呼ぶ) へ到達する過程が, 逐次意思決定に位置付けられている. また, 文献 [10] では, 日々製品別生産量を決定し, 完成品在庫をいかにしてなくならないようにかつ最小化するかを永遠に繰り返す過程が, 逐次意思決定に位置付けられている. アウトプットは, 前者ではスケジュールであり, 後者では製品別の生産量である. 言い換えれば, エージェントに与えられた役割は, 前者ではスケジューラーであり, 後者ではディスパッチャー (ジョブを機械に仕掛ける人ないし工場の製造ラインに投入する人) といえる.

本研究では, 後者の, 文献 [10] と同様の逐次意思決定問題としての定式化を提案する. なぜなら, 時刻経過とともにジョブが新規に到着する, 動的過程を想定した定式化であり, 実際のスケジュール実行時に機械の不具合発生といったトラブルを, 状態遷移の不確実性にそのまま解釈できるメリットが, 今後の研究で享受できるだろうと思われるからである.

また, この定式化を行う場合の計算量としては, ディスパッチャーがジョブの投入機会毎に投入ジョブを決定するだけが要件ならば, ジョブ数 n として投入前のジョブ集合から 1 つ選ぶだけなので, $O(n)$ で済む. なお, スケジューラーとして, 全ジョブの仕掛順をスケジュールとして一括出力するには, 投入ジョブの決定を $n-1$ 回行う必要があるので, $O(n^2)$ になる. 前者のスケジューラーとして定式化した場合の計算効率は, 交換可能なジョブの数や資源余裕により可能な行為数が可変であり, かつ, 最終状態に到達するまでの状態数が可変なので, 評価自体が困難である. しかし, 少なくとも, 最終状態で実行可能なスケジュールになるまでは, 途中の意思決定の結果としてのスケジュールは実行不能であるので, 注意が必要である.

3.1.2 政策

政策は, エージェントの知覚する状態を基に行為を決定するものであるから, 状態と行為の組で表される. 前章で述べたとおり, エージェントの行う学習とは累積報酬を最大化する政策を獲得することだが, 状態と行為の定式化次第で, 学習結果が大きく左右される.

以下では, 状態と行為の定式化を複数考案し, 実装において比較することとする.

強化学習における逐次意思決定を, 前述のとおり, スケジュール実行時間軸上での逐次意思決定とする結果, エージェントの行為は, ライン投入前の待ち行列から, 仕掛けるジョブを 1 つ選択することになる. 待ち行列内全てのジョブを選択の対象にすることは汎用性を持たせる点では利点がある. 他方, 問題を解くヒューリスティクスが予め分かっている場合, 選択対象を絞り込むことがルール獲得に役立つ. そこで, 3 種類の選択候補となるジョブ集合を作った.

a1) 最大 2 ジョブが候補になる. $p_{1i} \leq p_{2i}$ のジョブの中で p_{1i} 最小, $p_{1i} > p_{2i}$ のジョブの中で p_{2i} 最大, のもの. これは, ジョンソン則から得られる仕掛順の規則性をほとんど利用したものとなっている.

a2) 最大 8 ジョブが候補になる. $p_{1i} \leq p_{2i}$ のジョブの中での p_{1i}, p_{2i} それぞれについての最小と最大のもの, $p_{1i} > p_{2i}$ のジョブの中でも同様に選ぶ. つまり, 2^3 個のジョブが候補になる. 最初のジョブ集合の条件を緩和したものとなっている.

a3) 待ち行列内全てのジョブが候補になる.

次に, 本問題の状態記述として, 問題毎による差異を正規化するため, 0-1 区間で表される指標を, 7 つ考案した.

s1) 全ジョブ中, 待ち行列内の残りジョブ数：
= 待ち行列内のジョブ数 / 問題で与えられたジョブ数

s2) 待ち行列内のジョブのうち, 第 1 工程時間が第 2 工程時間より長いものの数：
= 待ち行列内のジョブのうち, 第 1 工程時間が第 2 工程時間より長いものの数 / 待ち行列内のジョブ数

s3) 第 1 工程で, 全ジョブに対する, 待ち行列内ジョブの残り負荷：
= (待ち行列内ジョブの第 1 工程処理時間合計 / 待ち行列内のジョブ数) / (問題で与えられたジョブの第 1 工程処理時間合計 / 問題で与えられたジョブ数)

s4) 第 2 工程で, 全ジョブに対する, 待ち行列内ジョブの残り負荷：
= 待ち行列内ジョブの第 2 工程処理時間合計 / 待ち行列内のジョブ数) / (問題で与えられたジョブの第 2 工程処理時間合計 / 問題で与えられたジョブ数)

s5) 待ち行列内のジョブで, 第 1 工程処理時間が全ジョブ中最小のジョブが残っているかどうか

s6) 待ち行列内のジョブで, 第 2 工程処理時間が全ジョブ中最小のジョブが残っているかどうか

s7) 第 2 工程の機械稼働率

3.1.3 報酬と価値関数

総処理時間最小化問題であるから, 報酬は, 最終状態において総処理時間を与えればよい. ただ, 累積報酬の最大化問題とするため, 総処理時間の負値を報酬値とした. また, 総処理時間は問題により異なるので, 正規化のため, 機械毎の総負荷時間の内, 大きい方で除することとした.

価値関数は, 状態表現が連続値であるためテーブル表現が困難であることや, 将来の大規模問題への対応を考慮し, 関数近似を行うこととした. ここでは, 活性化関数にシグモイド関数を用いた, 多層型ニューラル・ネットワークにより近似し, 誤差逆伝播法による重み修正を $TD(\lambda)$ 法に組み込んだ学習法とした. 入力層が状態, 出力層が推定価値になる. 推定価値も連続値であるため, 出力層の各ノードに定数の価値を割り付け, 各出力値を確率密度とみなして, その学習を行わせるようにした [7]. これは, 文献 [13] で採用された人工ニューラル・ネットワークと機能上同一である. 実装の詳細については, 次章で述べる.

3.2 3 機械問題

本節では, 3 機械問題を考える. ここで, 全ジョブの中で機械 M_2 の処理時間 p_{2i} の最大を $max_i\{p_{2i}\}$, 機械 M_3 の処理時間 p_{3i} の最小を $min_i\{p_{3i}\}$ とすると,

$$\max_i\{p_{2i}\} \leq \min_i\{p_{3i}\}$$

を満たす問題は, 第 2 工程が第 3 工程の負荷より小さいため, 第 3 工程に含まれる形となり, 2 機械問題として読み替えてジョンソン則を適用し, 最適解が求められる [2]. つまり,

2 ジョブ j_i, j_k について,

$$min[p_{1i} + p_{2i}, p_{2k} + p_{3k}] \leq min[p_{1k} + p_{2k}, p_{2i} + p_{3i}]$$

を満たすようにジョブ j_i を j_k に先行して処理する (等号成立時の仕掛順は任意) .

そこで, 3.1 節の状態と行為の定式化を 3 機械問題に対応するように変更する. 行為, 状態ともに, 第 1 工程処理時間を第 1 工程と第 2 工程の合計処理時間と, 第 2 工程処理時間を第 2 工程と第 3 工程の合計処理時間と読み替える. まず, 行為は,

a1) 最大 2 ジョブが候補になる. $p_{1i} + p_{2i} \leq p_{2i} + p_{3i}$ のジョブの中で $p_{1i} + p_{2i}$ 最小, $p_{1i} + p_{2i} > p_{2i} + p_{3i}$ のジョブの中で $p_{2i} + p_{3i}$ 最大, のもの.

a2) 最大 8 ジョブが候補になる. $p_{1i} + p_{2i} \leq p_{2i} + p_{3i}$ のジョブの中での $p_{1i} + p_{2i}, p_{2i} + p_{3i}$ それぞれについての最小と最大のもの, $p_{1i} + p_{2i} > p_{2i} + p_{3i}$ のジョブの中でも同様に選ぶ.

a3) 待ち行列内全てのジョブが候補になる.

次に, 状態記述を再定義する. 前工程とは第 1, 第 2 工程を, 後工程とは第 2, 第 3 工程をさす.

s1) 全ジョブ中, 待ち行列内の残りジョブ数

s2) 待ち行列内のジョブのうち, 前工程時間が後工程時間より長いものの数

s3) 前工程で, 全ジョブに対する, 待ち行列内ジョブの残り負荷

s4) 後工程で, 全ジョブに対する, 待ち行列内ジョブの残り負荷

s5) 待ち行列内のジョブで, 前工程処理時間が全ジョブ中最小のジョブが残っているかどうか

s6) 待ち行列内のジョブで, 後工程処理時間が全ジョブ中最小のジョブが残っているかどうか

s7) 後工程の機械稼働率

3.3 遺伝アルゴリズムとの統合

強化学習単独では, 最適政策獲得までに多くの訓練回数を必要とする [12]. そこで, 文献 [5] を参考に, 遺伝アルゴリズムを組合せ, その優良解を強化学習エージェントに学習させることで, 訓練回数の短縮を図る.

遺伝アルゴリズムを用いた最適化では, 複数の個体集合を適合度で評価し, 適合度に従って残すべき個体を選択し, 交叉や突然変異といった遺伝操作を行い, 次世代の個体集合を生成する過程を繰り返し, 優良解を求めようとする [9].

本研究での遺伝アルゴリズムは, スケジューリング・ルールの獲得ではなく, 特定のスケジューリング問題の優良解探索に用いる. まず, 個体表現は, ジョブを各工程にまで分割したオペレーション列とした. 2 機械,3 機械フローショップ・スケジューリングの総処理時間最小化問題においては, 第 1 機械前の待ち行列の処理順を決定するだけでよいことは前述した. しかし, ジョブショップ・スケージュリングや他の目的関数を対象にするときの, 将来研究での拡張性を考慮した. オペレーション列は, 仕掛優先度を表す. 適合度は, ジョブの総処理時間をそのまま用いる. 選択は, 文献 [5] で採用されている linear ranking selection method と cross-generational competition scheme をそのまま用いる. 交叉は, 文献 [5] で採用されている partially-matched crossover と position-based crossover の内, 前者のみを用いる. 詳細なアルゴリズムは, 文献 [9] に従った. 突然変異は, 文献 [5] で採用されている swapping と insertion の内, 前者のみを用いる. 以上のアルゴリズムにおけるパラメータとして, 個体集合のサイズ, 世代数, linear ranking selection の中での閾値 η, 交叉率, 突然変異率がある. 設定値は, 5.2 節でまとめて記述する.

強化学習エージェントは, 訓練集合の複数の問題を経験学習することで, スケジューリング・ルールを獲得する. 遺伝アルゴリズムは, 複数の問題に 1 対 1 対応で, 個体集合を保持する. 強化学習エージェントが, 特定の問題を解く前段で, 遺伝アルゴリズムによる探索を進める. つまり, その問題に対応する個体集合を 1 世代更新しておく. エージェントが情報獲得行為を選択したときに, 遺伝アルゴリズムの個体集合の中で適合度上位の優良解の 1 つを, そのまま実行して, エージェントがその解の価値を学習するようにする.

第 4 章

実装

前章での定式化を受け、実験を行うための実装アルゴリズムをここで述べる. 将来の問題の拡張や学習アルゴリズムの変更を見込み, できるだけ変更容易なプログラムとなるよう, オブジェクト指向モデリングを行った. 言語は Java で記述した.

4.1 強化学習エージェント

プログラムの main 関数に相当するのは, 以下の処理である.

```
1（指定回問題を解く）
2    訓練集合中から解く問題を決める
3    （全ジョブの処理が終了するまで繰り返す）
4        状態と報酬を計算する
5        価値関数を更新する
6        （第 1 工程の機械が空いて待ち行列内にジョブが 2 つ以上あれば）
7            仕掛けるジョブを決める
8        生産を 1 単位時間実行する
```

4 行目の処理が環境の担当する部分で, 5, 7 行目の処理はエージェントが担当する. 8 行目の処理は, 将来研究での拡張を見込み, ジョブショップ問題までを対象にできるように, シミュレータを設計実装した. 本節では, 5 行目価値関数の更新部分 (学習アルゴリズム) と, 7 行目仕掛けるジョブの決定部分 (政策ないし意思決定アルゴリズム) を詳述する.

4.1.1 価値関数の更新

まず, 5 行目価値関数の更新部分の処理を説明する. 3.1.3 節で述べたとおり, 本研究の強化学習エージェントは, 価値を多層型人工ニューラル・ネットワークに近似させる. 誤差逆伝播法による重み修正を $TD(\lambda)$ 法に組み込んだ学習法とは, TD の二乗誤差を最小化するネットワークの重みを勾配降下法で求めることである. ネットワークの重み W, 状態 s の価値を $V(s, W)$ とすると,

$$\begin{aligned} \Delta W &\equiv W_{t+1} - W_t && (4.1)\\ &\equiv -\frac{1}{2}\alpha\nabla_{W_t}[r_{t+1} + \gamma V_t(s_{t+1}, W_t) - V_t(s_t, W_t)]^2 && (4.2)\\ &= \alpha[r_{t+1} + \gamma V_t(s_{t+1}, W_t) - V_t(s_t, W_t)]\nabla_{W_t} V_t(s_t, W_t) && (4.3) \end{aligned}$$

ここで, $\nabla_{W_t}[r_{t+1} + \gamma V_t(s_{t+1}, W_t) - V_t(s_t, W_t)]^2$ は, $[r_{t+1} + \gamma V_t(s_{t+1}, W_t) - V_t(s_t, W_t)]^2$ の W_t についての偏導関数である.(4.3) 式は, 合成関数の微分により導かれる. ただし,$r_{t+1} + \gamma V_t(s_{t+1}, W_t)$ は, 微分の際, 真値とみなされ定数として扱われる. 重み訂正ベクトル (4.1) は, (4.2) 式のように勾配 $\nabla_{W_t}[r_{t+1} + \gamma V_t(s_{t+1}, W_t) - V_t(s_t, W_t)]^2$ とは逆方向に学習率 (ステップサイズ)$\frac{1}{2}\alpha$ 倍と定義される. 整理された (4.3) 式は 2 章の TD 法の学習式 (価値関数更新式) となる. λ 波及を含めて, 2 章の $TD(\lambda)$ 法の形に書きなおすと,

$$\begin{aligned} W_{t+1} &= W_t + \alpha[r_{t+1} + \gamma V_t(s_{t+1}, W_t) - V_t(s_t, W_t)]e_t && (4.4)\\ e_t &= \gamma\lambda e_{t-1} + \nabla_{W_t} V_t(s_t, W_t)) && (4.5)\\ & (0 \leq \alpha, \gamma, \lambda \leq 1) && (4.6) \end{aligned}$$

ここで, 適確波及度 e は, W と 1 対 1 対応である. 重み W の更新は, 出力層と隠れ層の間と隠れ層と入力層との間で異なる. それぞれの更新式を導出する. 詳細な重み訂正アルゴリズムについては文献 [8] に従う. 出力層ノードを O, 隠れ層ノードを H, 入力層ノードを I とし, 出力層, 隠れ層の個別ノードを O_i, H_j, 入力層の入力値を I_k と表記する. 出力層と隠れ層の間の重みを $W_{j,i}$, 隠れ層と出力層の間の重みを $W_{k,j}$ とする. 併せて, 出力層と隠れ層の間の適確波及関数を $e_{j,i}$, 隠れ層と出力層の間を $e_{k,j}$ とする. 出力層, 隠れ層ノードへの入力値を in_i, in_j, 活性化関数を f, ノードからの出力値を a_i, a_j とする. また, ネットワークからの出力値に対する状態価値の割当値を v_i, 出力値 a_i の目標値を $target_i$ とする. なお, I_k, $W_{j,i}$, $W_{k,j}$, $e_{j,i}$, $e_{k,j}$, in_i, in_j, a_i, a_j, $target_i$ の時刻 t 時の値は, I_k^t などと表記

する. 以下 (4.7) から (4.12) 式は, 定義式である.

$$V(s,W) = \frac{\sum_i a_i v_i}{\sum_i a_i} \tag{4.7}$$

$$a_i = f(in_i) = \frac{1}{1+exp(-in_i)} \tag{4.8}$$

$$in_i = \sum_j a_j W_{j,i} \tag{4.9}$$

$$a_j = \frac{1}{1+exp(-in_j)} \tag{4.10}$$

$$in_j = \sum_k I_k W_{k,j} \quad (I_k \in s) \tag{4.11}$$

$$target_i^{t+1} = exp(-\frac{(r_{t+1}+\gamma V_t(s_{t+1},W_t)-v_i)^2}{10}) \tag{4.12}$$

(4.7) と (4.12) 式は, 価値が連続値であることから, 出力層の各ノードに定数を割り付け, 各出力値を確率密度とみなしている部分である. 活性化関数は, (4.8)(4.10) 式のとおり, シグモイド関数を用いている.

$W_{j,i}$ について (4.2) 式と等価なものとして,

$$\Delta W_{j,i} \equiv -\frac{1}{2}\alpha\nabla_{W_{j,i}^t}(target_i^{t+1}-a_i^t)^2 \tag{4.13}$$

$$= \alpha(target_i^{t+1}-a_i^t)\nabla_{W_{j,i}^t}a_i^t \qquad \text{(合成関数の微分)} \tag{4.14}$$

$$= \alpha(target_i^{t+1}-a_i^t)\nabla_{W_{j,i}^t}f(in_i^t) \qquad \text{((4.8) 式代入)} \tag{4.15}$$

$$= \alpha(target_i^{t+1}-a_i^t)f'(in_i^t)\frac{\partial\sum_j a_j^t W_{j,i}^t}{\partial W_{j,i}^t} \qquad \text{((4.9) 式代入)} \tag{4.16}$$

$$= \alpha(target_i^{t+1}-a_i^t)f'(in_i^t)a_j^t \tag{4.17}$$

$$= \alpha(target_i^{t+1}-a_i^t)a_j^t f(in_i^t)(1-f(in_i^t)) \tag{4.18}$$

従って (4.4)(4.5) 式と等価なものとして,

$$\Delta W_{j,i} = \alpha(target_i^{t+1}-a_i^t)e_{j,i}^t \tag{4.19}$$

$$e_{j,i}^t = \gamma\lambda e_{j,i}^{t-1} + a_j^t f(in_i^t)(1-f(in_i^t)) \tag{4.20}$$

同様に $W_{k,j}$ についても更新式を導出できる.

$$\Delta W_{k,j} = \alpha\sum_i[W_{k,j}^t(target_i^{t+1}-a_i^t)f(in_i^t)(1-f(in_i^t)]e_{k,j}^t \tag{4.21}$$

$$e_{k,j}^t = \gamma\lambda e_{k,j}^{t-1} + I_k^t f(in_j^t)(1-f(in_j^t)) \tag{4.22}$$

以上, (4.19) から (4.22) 式が, 実装された価値関数の更新式である.

4.1.2 仕掛けるジョブの決定

次に, 仕掛けるジョブの決定の処理概要は以下のとおり.

```
1 確率εで報酬獲得行為とする
2（情報獲得行為の場合）
3    待ち行列の中からランダムに1つのジョブを選択する
4（報酬獲得行為の場合）
5    （待ち行列の中の各ジョブについて1つずつ繰り返す）
6        当該ジョブを選択したときの報酬と次状態を計算する
7        報酬と次状態価値合計を計算し，現状態価値を推定する
8    推定された現状態価値が最大になるジョブを選択する
9  εを減少させる
```

情報獲得行為はランダムとする. 報酬獲得行為は, その時点での価値関数を使って現状態価値が最大になる行為を選択するものである. 次状態と報酬は, 本研究ではエージェントにとって既知であるものとして実験する. 実装では, その時点でのシミュレータのクローンを作って次状態と報酬を計算し, エージェントに伝える処理にした.

4.2 強化学習エージェントと遺伝アルゴリズムの統合

前節強化学習エージェント単独との処理の違いは2個所ある. 3, 4行目の処理が追加され, 11行目の処理基準が変更される.

```
1（指定回問題を解く）
2    訓練集合中から解く問題を決める
3    解く問題の遺伝アルゴリズムにおける世代を1つ更新する
4    遺伝アルゴリズムより，優良解を1つ選択しておく
5    （全ジョブの処理が終了するまで繰り返す）
6        状態とを計算する
7        価値関数を更新する
8        （第1工程の機械が空いて待ち行列内にジョブが2つ以上あれば）
9            確率εで報酬獲得行為とする
10               （情報獲得行為の場合）
11                   待ち行列中から遺伝アルゴリズムの優良解に従ったジョブを選択する
12               （報酬獲得行為の場合）
13                   （待ち行列の中の各ジョブについて1つずつ繰り返す）
14                       当該ジョブを選択したときの報酬と次状態を計算する
15                       報酬と次状態価値合計を計算し，現状態価値を推定する
16                   推定された現状態価値が最大になるジョブを選択する
17           εを減少させる
18       生産を1単位時間実行する
```

第 5 章

実験

5.1 問題の生成

問題は, 一様乱数により, 1 処理当たりの処理時間が 1-30 単位時間となるようにして, 10 ジョブの問題が 100 問生成された. このうち, 50 問を訓練集合とし, 50 問をテスト集合とした. 訓練集合の問題は, 順番に繰り返し実行して, 累計 20000 回（1 問につき 400 回）実行し, 1 回の学習とした. 1 問につき, 仕掛順の組合せは 10!であり, 学習中全ての行為が情報獲得行為であったとして, 全体の約 1.1×10^{-2} %の組合せが試行されることになる. 状態と行為の定義の組合せを変えながら, ひとつの組合せにつき 10 回の学習を行った.

5.2 学習アルゴリズムのパラメータ設定

情報獲得行為をとる確率 ϵ は, 初めの 5, 000 回は 1.0 で固定し, その後意思決定 1 回毎に 10^{-5} づつ減少するようにした. つまり, 16, 000 回を過ぎたあたりで, ほとんど報酬獲得行為のみを行うようになる. また, $\alpha = 0.001$, $\lambda = 0.9$ とした. 割引率は, 本研究では最終状態がある問題を解くので,$\gamma = 1.0$ とした. なお, 訓練集合を実行する回数, ϵ, α, λ の値のとり方は, 本問題を予備的に解いて, 学習効果がはっきり顕れるように, 試行錯誤を通じて調整した.

人工ニューラル・ネットワークの重みの初期値は, ネットワークの形状を予想することは大変難しいので, 全て 0.0 とした. ネットワークの隠れ層のノード数は, 入力層ノードの全てのブール関数表現をするために, 入力する状態記述の数 n に依存して, $2^n/n$ 以上の最

小の自然数となるようにした [8]. 出力層のノード数は,10 とした. 出力層の割当値は,-0.5 から-1.4 まで 0.1 刻みとした. ここでは, 出力層の割当値の平均が-0.95 となり,-1.0 よりも大きいことが重要である. ネットワークの重みの初期値が 0.0 であるから, 初期の推定価値は-0.95 となる. 報酬の値域は,-1.0 より小さいので, 学習過程において報酬獲得行為をとる場合でも, 積極的に未経験の仕掛順を試行するようになり, 学習結果の質を高める. その効果は, 予備的な実験で, 割当値の平均が-2.0 になる場合との比較で確認できた.

遺伝アルゴリズムのパラメータは, 文献 [5] を参考に決めた. 個体集合のサイズは, オペレーション数 30 の 7 倍で 210 個体, 更新する世代数は 200 世代, linear ranking selection の閾値は $\eta = 1.2$, 交叉率は 1.0, 突然変異率は 0.5 とした.

5.3 実験結果

5.3.1 2 機械問題

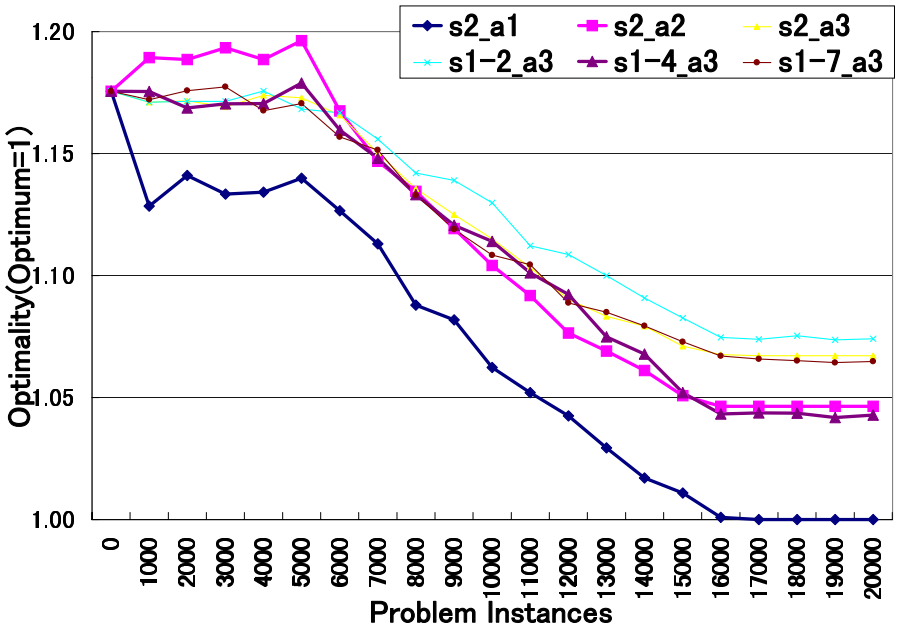

図 5.1: $F2||C_{max}$ 問題 10 ジョブ 50 問題を累計 20000 回解いた結果

図 5.1 では, 状態と行為の定義の組み合わせごとに, ジョンソン則の結果を基準とした解性能の比較をグラフ化した. 各点での値は, 10 回の学習の平均である. 0 から 20000 ま

では学習中の過程を示すため, 1000 回毎に, 直前の 50 問題平均の総処理時間を, 最適解との比較で表示してある. なお, 横軸原点の値は, 問題所与の順序で実行した場合の結果である. ここで, 記法「s 番号_a 番号」については, s が状態, a が行為の定義を表し, 後に続く番号は, 3.1 節で定義した状態や行為の番号に対応する. 例えば, s[1-4]_a3 は, 状態**s1**, **s2**, **s3**, **s4** と行為**a3** の組合せによる学習であることを示す.

また, 学習後に（価値関数更新を停止させて）テスト集合を解き, その平均を算出したところ, s2_a1=1, s2_a2=1.05, s2_a3=1.06, s[1-2]_a3=1.07, s[1-4]_a3=1.04, s[1-7]_a3=1.06 となった. ちなみに, 問題所与の順で実行すると 1.18 であった. 表 5.1 の s2_a1 では, ジョ

テスト集合中最適解と同一になった問題数				
組合せ	最悪	平均	最良	偏差
s2_a1	50	50	50	0.0
s2_a2	10	10	10	0.0
s2_a3	4	4	4	0.0
s[1-2]_a3	4	4	4	0.0
s[1-4]_a3	6	9.5	12	1.8
s[1-7]_a3	0	2.8	6	1.6

表 5.1: $F2||C_{max}$ 問題で学習後にテスト集合を解いた結果

ンソン則と全く同一のスケジュールを常に実行した. 1 回の学習についてみると, s[1-4]_a3 では, 最悪でも 50 問中 6 問でジョンソン則と同一の総処理時間を実現するスケジュールを実行した. ちなみに, 問題所与の順で実行すると, 1 問だけジョンソン則と同一になる. なお, 全ての組合せについて, 問題所与の順で, つまりランダムに実行する場合と比較すると, 全問題中 43 問から 50 問で同等かより良い結果を得ている. s[1-4]_a3 と s[1-7]_a3 には, 学習回ごとのばらつきが見られる.

状態, 行為各々についてみると, **s5**, **s6**, **s7** は, 本問題を解くジョンソン則からみて妥当な状態記述かと思われたが, 解性能の改善に貢献しなかった. また行為については, ジョンソン則に沿った絞り込みを行った**a1** の方が, **a2** や**a3** よりも良好な結果が出ている.

5.3.2　3 機械問題

本節では, 3 機械問題を考える. まず, 2 機械問題と同様の学習ができるかどうかを, 2 機械問題への読み替えによりジョンソン則を適用することで最適解が求まる問題集合 ($max_i\{p_{2i}\} \leq min_i\{p_{3i}\}$ を満たす場合) で実験する. 以後, 単にジョンソン則により最適解が求まる 3 機械問題と呼ぶ. 次に, 一般的な 3 機械問題 ($max_i\{p_{2i}\} \leq min_i\{p_{3i}\}$ を満たさない場合) で実験し, 比較考察する. 以下の図 5.2, 図 5.3 の評価法は, 図 5.1 と同様, 表 5.2, 表 5.3 も, 表 5.1 と同様である.

i) $max_i\{p_{2i}\} \leq min_i\{p_{3i}\}$ を満たす場合

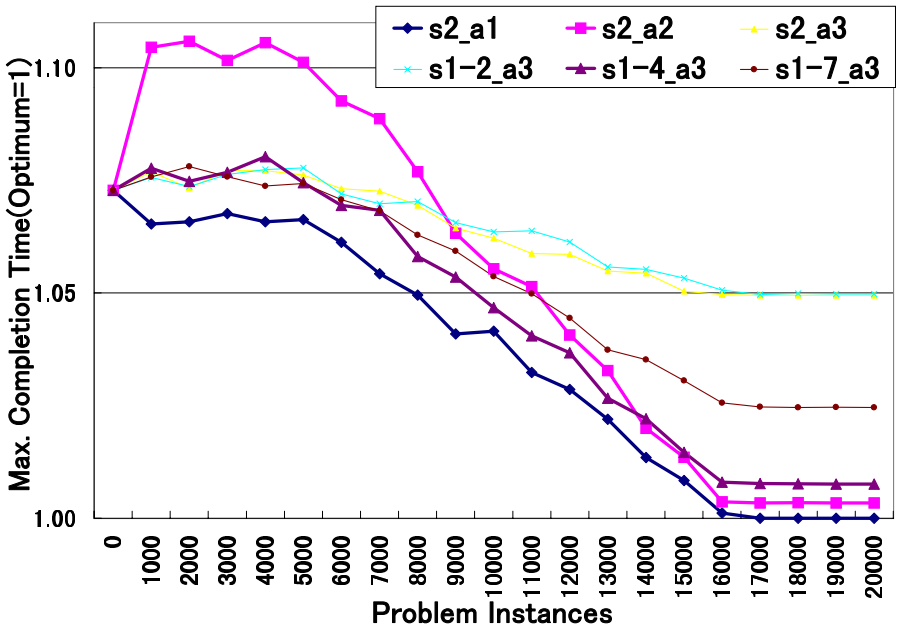

図 5.2: $F3||C_{max}$ 問題 (ジョンソン則で最小)10 ジョブ 50 問題を累計 20000 回解いた結果

結果は, 2 機械の場合とほぼ同一である. ジョンソン則により最適解が求まる 3 機械問題において, 3 機械問題の強化学習の定式化が, 2 機械問題のときと同等に行われたことが確認できた. ただ, s2_a2 の組合せで, 学習初期において過渡的に解性能が悪くなる現象がみられた. また, $max_i\{p_{2i}\} \leq min_i\{p_{3i}\}$ を満たす問題集合の特性のためか, s2_a2 や s[1-4]_a3

テスト集合中最適解と同一になった問題数				
組合せ	最悪	平均	最良	偏差
s2_a1	50	50	50	0.0
s2_a2	46	46	46	0.0
s2_a3	2	2	2	0.0
s[1-2]_a3	2	2	2	0.0
s[1-4]_a3	32	32.8	32	0.4
s[1-7]_a3	3	3.5	4	0.5

表 5.2: $F3||C_{max}$ 問題 (ジョンソン則で最小) で学習後にテスト集合を解いた結果

と, s2_a3 や s[1-2]_a3, s[1-7]_a3 との結果が対照的になっている点が, 興味深い (表 5.2).

ii) $max_i\{p_{2i}\} \leq min_i\{p_{3i}\}$ を満たさない場合

結果は, 前節のジョンソン則により最適解が求まる 3 機械問題を解く場合と比べ, 概ね同様の結果を得ることができた. 再度指摘すると, この問題集合は,3.2 節のルールによる処理順で解いても, 必ずしも総処理時間が最小になるわけではない. しかし, 本研究の定式化による強化学習エージェントは, いずれもジョンソン則で求まった解を上回ることができなかった. 学習後テスト集合でのジョンソン則による解を基準にした比較では, 平均で, s2_a1=1, s2_a2=1.05, s2_a3=1.06, s[1-2]_a3=1.07, s[1-4]_a3=1.04, s[1-7]_a3=1.06 となった. ちなみに, 問題所与の順で実行すると 1.14 であった. また, 表 5.3 では, s[1-4]_a3 や s[1-7]_a3 に加えて, s[1-2]_a3 でも学習回ごとでのばらつきがでている.

ちなみに, ここにはグラフを掲載していないが, 状態定義で第 1 工程に関する指標はそのままに, 第 2 工程に関する指標を第 3 工程と読み替えて行った実験では, 学習はするものの, 総じて悪い結果となった. 特に, s2_a1 の組合せにおいても, ジョンソン則で求まった解と同一にならない問題があった. このことは, 解空間を記憶（学習）する状態定義の重要性を示している.

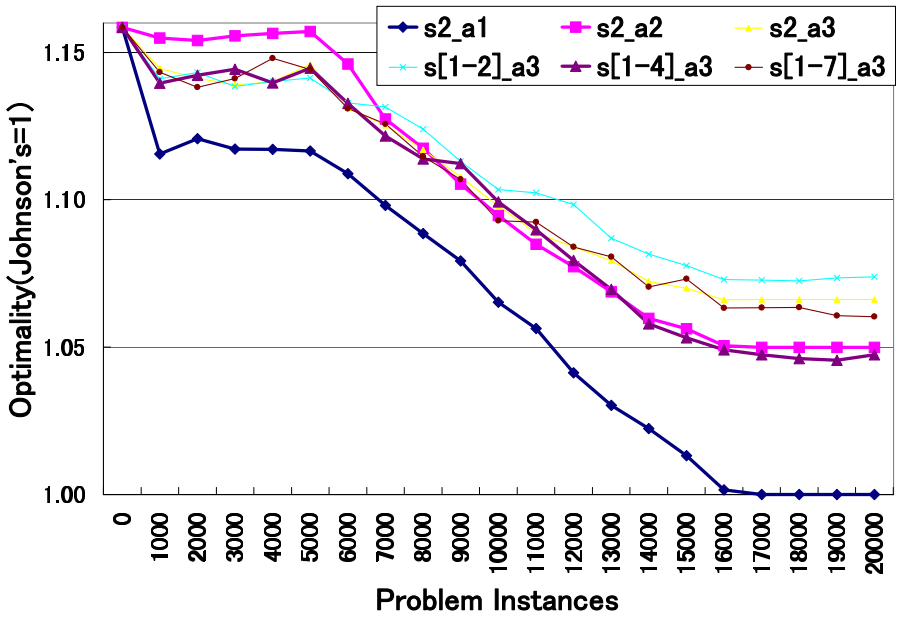

図 5.3: $F3||C_{max}$ 問題 10 ジョブ 50 問題を累計 20000 回解いた結果, ジョンソン則で求まった解を基準にして比較

テスト集合中ジョンソン則と同一ないし上回った問題数				
組合せ	最悪	平均	最良	偏差
s2_a1	50	50	50	0.0
s2_a2	15	15	15	0.0
s2_a3	9	9	9	0.0
s[1-2]_a3	8	8.2	9	0.4
s[1-4]_a3	12	14.4	17	2.0
s[1-7]_a3	8	9.3	11	0.8

表 5.3: $F3||C_{max}$ 問題で学習後にテスト集合を解いた結果

5.4 学習の過程

本研究で行った定式化がどのように機能しているかを, s2_a1 の組合せにおいて, 一般的な 3 機械問題を対象にして, ジョンソン則による解と同一解を実現する過程で, 簡単に説明する. 第 1 工程と第 2 工程を合わせて前工程, 第 2 工程と第 3 工程を合わせて後工程とする. 行為は, 前工程時間の方が短いジョブの中で前工程時間最小のジョブと, 後工程時間の方が短いジョブの中で後工程時間最大のジョブの 2 者択一であり, このどちらが良い選択であったかは, 待ち行列内のジョブで, 前工程時間の方が後工程時間より長いジョブの割合に応じて記憶される. 前工程時間の方が長いジョブばかり (s2=1.0) になるには, 前者の行為を選びつづけなければならず, 後工程時間の方が長いジョブばかり (s2=0.0) になるには, 後者の行為を選びつづけなければならない. つまり, 後者の過程はジョンソン則とは全く逆のスケジュールとなるから, 累積報酬も低くなってしまうので, 学習後には前者の行為しか選択しなくなるはずである. そして, 前工程時間の方が長いジョブばかりになった後は, この定式化では, 状態価値判断による行為選択はなくなる. 従って, 問題集合の大勢においてジョンソン則で最適解が求まるかぎり, 学習結果もジョンソン則と同一になる.

図 5.4 と図 5.5 は, 学習中の試行錯誤の過程を示している. s2_a1 の組合せで一般的な 3 機械問題を解いている. 図に示されているのは, 訓練集合中特定の 1 問についてで, この問題は, たまたまジョンソン則により最適解が求まるものである. 両方の図ともに比較のため, 10!通り全組合せを探索した結果の分布を表示してある. 5.2 節で述べたとおり, 最初の 5, 000 回は情報獲得行為のみで, 最後の 3, 000 回は報酬獲得行為のみを行う. 図 5.4 のとおり, 初めの 5, 000 回はジョブをランダムに仕掛けるので, 総処理時間の分布は, 広い範囲に散らばっている. 一方図 5.5 のとおり, エージェントが, 前節で述べたように, より報酬の大きい行為を選択するような学習をする結果, 最後の 3, 000 回では必ず総処理時間を最小化する仕掛りを行うようになっている.

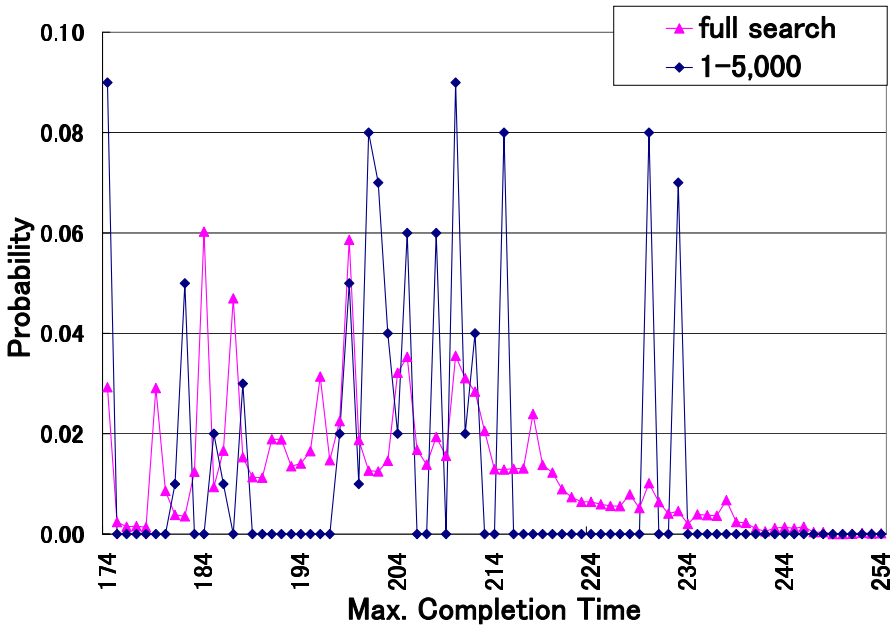

図 5.4: 1 回目から 5,000 回目まで問題を解いている間の試行錯誤の履歴

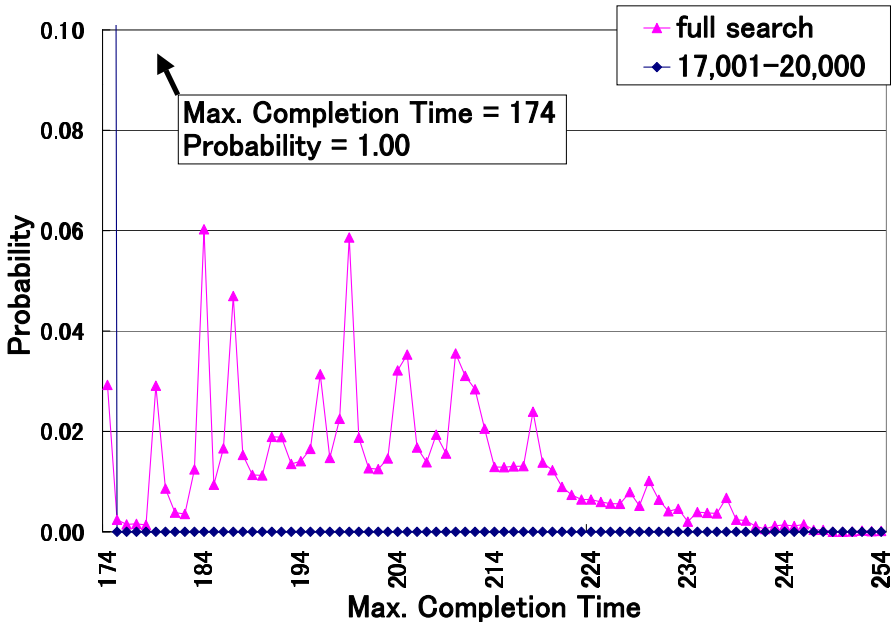

図 5.5: 17,001 回目から 20,000 回目まで問題を解いている間の試行錯誤の履歴

5.5 遺伝アルゴリズムとの統合

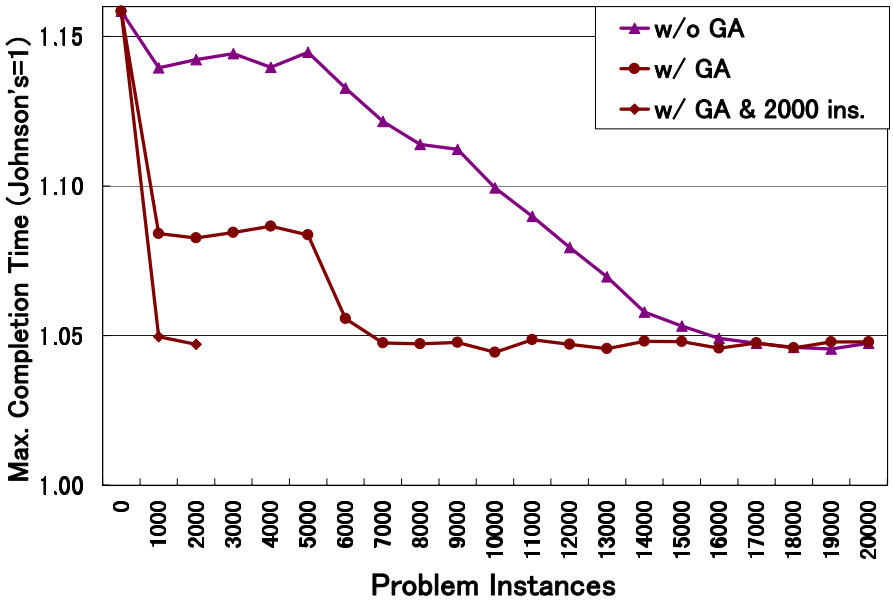

図 5.6: $F3||C_{max}$ 問題 10 ジョブ 50 問題を s[1-4]_a3 の組合せで解いた結果

図 5.6 は, 強化学習単独と遺伝アルゴリズムとの組合せの学習速度の比較結果を示している. 「w/o GA」と表記されているのが, 強化学習単独の結果で, 図 5.3 の s[1-4]_a3 と同じである. 「w/ GA」と表記されているのが, 遺伝アルゴリズムとの組合せの結果である. 遺伝アルゴリズムとの組合せの方が, 明らかに学習速度が速い. 強化学習エージェントが情報獲得行為のみをとる 5, 000 回問題を解くまでの間を見ても, 初めの 1, 000 回で強化学習単独と比べてよりよい結果を示している. さらに報酬獲得行為をとり始める 5, 000 回以降, わずか 2, 000 回程度で最終的な成績と大差ないレベルにまで学習が進んでいる.

そこで, 遺伝アルゴリズムとの組合せでの ϵ パラメータ (情報獲得行為をとる確率) を, 初め 500 回を 1.0 で固定, そこから意思決定 1 回につき 10^{-4} ずつ減少させ, 1, 600 回過ぎには報酬獲得行為のみをとるようにして, 2, 000 回で学習を打ち切る実験も行った. 図 5.6 において,「w/ GA & 2000 ins.」と表記されているのが, 遺伝アルゴリズムとの組合せでかつ訓練期間を短縮したものの結果である. 学習過程は,「w/ GA」と比べておおむね同等の弧を描いているようである. さらに, テスト集合中で比較した場合, 表 5.4 のとおり, 20, 000 回学習させた場合と比べて, 総処理時間は, 最悪, 平均, 最良ともに若干劣っている

ものの, 偏差は同程度に安定して学習することができた.

テスト集合中ジョンソン則と同一ないし上回った問題数				
組合せ	最悪	平均	最良	偏差
w/o GA	12	14.4	17	2.0
w/ GA	12	13.8	17	1.7
w/ GA & 2000ins.	10	11.9	16	1.7

表 5.4: $F3||C_{max}$ 問題で学習後にテスト集合を解いた結果

第 6 章

評価および考察

以上の実験結果は, 本研究で提案したスケジューリングを行うエージェントに, 学習能力があることを示している.

6.1 2 機械問題

2 機械問題のように最適解獲得のアルゴリズムが存在する場合, 問題の適切な定式化を行えば, 最適アルゴリズムと同一解を得られることが確認できた. また, 定式化が不完全である場合でも, 本研究で扱った例では, 最適解の 1.07 倍程度の解が得られ, 学習能力があることも観測された. これは, 最適解が獲得できない問題領域でも学習が有効となる可能性を示唆している.

6.2 3 機械問題

さらに, 3 機械問題で理論解が存在しない (全数探索を除くと最適解が求まらないこと) 場合でも, 2 機械問題と同程度の学習効果をあげることができた. ただ, ジョンソン則で求まる解の近傍に最適解が存在することが多いことから [2], ジョンソン則を参考にした本研究の定式化による学習において, 2 機械問題と同程度の成績をあげることは, 相当程度予想されていた.

なお, いくつかの特定の問題で, ジョンソン則を上回る解を出す状態と行為の組合せが

あったものの, 平均においては劣るものだった. 原因としては, 本問題では理論解が存在しないとはいえ, ジョンソン則で求まる解は, 前述のとおり, 優良解を出す可能性が高い. 従って, それをさらに上回る学習結果を出すのは, 相対的に困難であったと思われる.

また, 上記のことから, 他問題での本研究のアプローチの有効性が直ちに否定されるわけではない. 確かに, 理論解が存在しない問題領域で, 従来法を上回る成績を残そうとするときに, 従来法を参考にした定式化のみでは限界があることを念頭におく必要はあると思われる. そこで例えば, 強化学習の特徴のひとつである, 逐次意思決定による状態遷移の軌跡上の価値伝播を活用した, 独創的な定式化をするなどして, 従来法のヒューリスティクスに付加部分を生み出す努力が必要と思われる. しかし, 定式化が不完全であっても, それなりの学習結果が残せているところから, より良い解を目指す定式化とその定式化に期待する学習内容との間に明瞭な論理性が設計者によって説明できなくとも, 学習が効果を挙げる可能性は, 十分にあるものと考えられる.

6.3 遺伝アルゴリズムとの統合

強化学習と遺伝アルゴリズムとの統合は, 強化学習の訓練回数の短縮に効果があることが確認された. ただし, 強化学習と遺伝アルゴリズムとを合計した探索回数は, 強化学習単独で 20, 000 回解くのに比べて, 強化学習で 2, 000 回, 遺伝アルゴリズムで 210 個体集合を 200 世代と, 合わせて 43, 000 回解いていることになり, 全体の計算量が少なくなるわけではない. しかし, 強化学習の並列化が困難なのに対して, 遺伝アルゴリズムの並列化は一般的であるから, 計算速度を上げることは容易であると考えられる.

6.4 状態行為定式化について

s[1-4]_a3 などの状態と行為の組合せでは, 学習結果にばらつきが見られた. 考えられる原因として, (1) 確率的な探索であるために, 探索空間内で実際に探索した部分空間にずれがある, (2) 状態定義の特性から, 同一軌道に対して異なる報酬が与えられるなど, 設計時に期待しているような価値関数の更新ができていない, ことが挙げられるが, これらの原因追求は今後の課題である.

なお, 遺伝アルゴリズムの 200 世代後の個体集合中最優良解 50 問平均は, 強化学習エー

ジェントが学習後に解いた訓練集合平均よりも優れていた. このことは, s[1-4]_a3 の組合せにおける状態定義の学習能力の限界を示しているといえる.

6.5 学習結果の性質について

ここで, 学習能力として, 訓練集合に依存せずに, よりメタな知識を獲得することと, 訓練集合に依存した, より局所的な知識を獲得することを対比してみる. 訓練集合に依存しない知識獲得には, 問題の定式化に工夫が必要である. (例えば, ディスパッチングルールなどの解法の知識が予めあることを前提とし, それらの良否を学習するような場合が考えられる) このため, 現状では学習能力として期待できるのは, 訓練集合に依存した知識獲得とならざるをえない.

第 7章

関連研究

本章では, 既に引用した文献のまとめとその他の関連研究について概説する.

7.1 強化学習とスケジューリング

Zhang and Dietterrich[13] は,1 章で紹介したとおり, 本研究のきっかけとなった研究である. 目的関数はジョブの最大処理時間最小化で本研究と同一だが, ジョブショップ・スケジューリングが対象になっている. また,NASA のスペースシャトル貨物処理スケジューリング問題に特化して, 強化学習問題が定式化されている. 価値関数の更新に $TD(\lambda)$ 法を用いた点や人工ニューラル・ネットワークによる関数近似を行った点を, 本研究ではそのまま採用した. しかしながら, 強化学習の逐次意思決定の捉え方が, 本研究とは異なる.

Schneider, Boyan, and Moore[10] は, 食品工場の日別製品別生産量の意思決定問題を扱っている. その点で, ジョブの仕掛順を決定する通常の生産スケジューリング問題とは異なる. 本研究では, 逐次意思決定の捉え方で参考にした. 日別製品別生産量の生産計画が決まっていても, 操業ではそのとおりには生産できない工場の「くせ」を経験学習することで, 従来の発見的な意思決定法に比べて, 利潤をより大きくできることをシミュレーションで示した. つまり, 強化学習エージェントが不確実性を見込んだ計画を立てるようになることを示した. また, シミュレーテッド・アニーリング法との比較実験もあり, 探索回ごとでの結果にばらつきが大きいとの報告がある.

7.2 強化学習と遺伝アルゴリズム

Kim and Lee[5] は, genetic reinforcement learning approach と称して, フローショップ, ジョブショップ, オープンショップ・スケジューリング問題を, ジョブの総処理時間最小化を目的として, 同一の枠組みの中で全て解く研究を行った. 汎用的な枠組みにあっても, 多くの問題で最適解や従来法中の最良解を探索することに成功している. 本研究では, 強化学習に遺伝アルゴリズムを組み込む際に参考にした. なお, ここでいう遺伝 (的) 強化学習アプローチとは, 遺伝アルゴリズムそのものが強化学習と同質性をもつと指摘するもので, 遺伝アルゴリズムと強化学習が組み合わされているわけではない. 価値関数や学習といった概念は, 実装されていない.

神谷 [4] は, 発電プラント起動スケジューリング問題を扱っている. 種々の探索アルゴリズムを組合せて利用することによって, 発電タービン起動中の条件変更をする際に見合った十分な応答性能を確保することに成功している. 本研究との共通点が一見多いようにみえるが, 定式化が全く異なる. まず, 価値関数に人工ニューラル・ネットワークを用いている点では本研究と同一である. しかし, ネットワークの入力が状態であるのに対して, 出力が行為であり, 本研究のように価値ではない. この研究での報酬は, 行為そのものになっていて, ある条件を満たす解を出すと, その行為を正しい行為としてそのまま繰り返して学習する. 制御値が離散的な実在の発電タービンを対象にしている問題で, 行為数が極めて限定的でかつ不変であるために可能となった定式化である. また, 遺伝アルゴリズムを強化学習の加速に使っている点で, 本研究とは同一であるが, その組み込み方は異なる. 強化学習の解が何回か続けて目的関数の基準値を下回るようであれば, 遺伝アルゴリズムに探索が移り, 基準値を上回るまで続け, その後強化学習にその結果を学習させて基準値を上回る行為をとらせようとする. このように, 行為が限定されていることを利用して, 全体的に教師付きに近い学習を行わせている. 本研究で対象にする生産スケジューリング問題とは扱っている問題が異なり, 同列で論ずることができないので, 他章では引用しなかった.

第 8章

結論

8.1 まとめ

以上をまとめると,

- フローショップ・スケジューリング問題に対し強化学習問題としての定式化手法を述べた.
- 理論解の存在する 2 機械問題と, 理論解の存在しない 3 機械問題を実装評価した結果, 双方において上記定式化の有効性が確認できた.

8.2 課題

今後の研究課題としては,

- 状態の表現能力（より汎用的で, 特殊な問題のルール誘導型ではない表現はないか）
- 行為の絞り込み（最適行為を取りこぼさないようにしながら行為を限定すること）
- 状態価値ではなく行為価値による学習の検討（$Q(\lambda)$ 法での学習）
- 報酬の適切さ (よりよいスケジューリングに加重して報酬を与える, 簡便に正規化する方法など）

- パラメータ最適化による学習時間の短縮化（αが可変で初期において大きい値をとることなど）
- 他の確率的探索手法との比較（シミュレーッテッド・アニーリング (焼きなまし法) など）

などが挙げられる.

また, 対象をフローショップからジョブショップへ, 目的関数についてもジョブの総処理時間最小化以外へと拡張し, 問題がより複雑になる中でも, 学習能力のあるエージェントに関する研究が必要であるものと考えられる. 生産スケジューリング問題に対し, 本研究のアプローチをとることでそもそも意図したところは, 第 1 章で述べたとおり, 理論解が存在する問題よりも, 存在しない問題での優良解の獲得にあった. しかしながら, 問題を解くヒューリスティクスがないような問題に対して, 適切な状態記述や行為の絞り込みを行うことは困難を伴うことも予想される.

謝辞

理解が遅くアイディアが枯渇しがちな拙者に対し, 粘り強く指導を続けてくださいました吉田 武稔先生に, 深く感謝いたします.

また, 平石 邦彦先生には, 研究内容についての意見に加え, 本研究で他研究科教官の指導を受けられるよう早い段階で便宜を図っていただきました. Milan Vlach 先生からは, 本研究の進め方および意義について有益かつ貴重な意見をいただきました. 合わせて深く感謝いたします.

怠けがちな私に, 勉強会でペースメーカになっていただいた, 知識科学研究科複合システム論講座の方々, ありがとうございました.

参考文献

[1] 浅田 稔: 強化学習の実ロボットへの応用とその課題. 人工知能学会誌 12(6), pp.23-28(1997).

[2] Conway, R.W., Maxwell, W.L., and Miller, L.W.: *Theory of scheduling, Chap.5,* Addison-Wesley(1967).

[3] 茨木 俊秀: スケジューリング問題と計算の複雑さ. オペレーションズ・リサーチ 1994 年 10 月号, pp.29-34(1994).

[4] 神谷 昭基: 強化学習を用いた発電プラント起動スケジューリング. 人工知能学会誌 12(6), pp.29-36(1997).

[5] Kim, G.H., Lee, C.S.G.: Genetic reinforcement learning approach to the heterogeneous machine scheduling problem. *IEEE Transactions on robotics and automation* 14(6), pp.879-893(1998).

[6] 宮下 和雄: 知的スケジューリングにおける知識獲得. 人工知能学会知識ベースシステム研究会資料 SIG-KBS-9901, pp.41-46(1999).

[7] Pomerleau, D.A.: Efficient training of artificial neuralnetworks for autonomous navigation. *Neural computation* 3(1), pp.88-97(1991).

[8] Russell, S., and Norvig, P.: *Artificial intelligence, Chap.19,* Prentice Hall(1995).

[9] 三宮 信夫, 喜多 一, 玉置 久, 岩本 貴司: 遺伝アルゴリズムと最適化, 2 章 3 章, 朝倉書店 (1998).

[10] Schneider, J.G., Boyan, J.A., and Moore, A.W.: Value function based production scheduling. *Machine learning: proc. of the fifteenth international conference (ICML '98),* pp.522-530(1998).

[11] Sutton, R.S., and Barto, A.G.: *Reinforcement learning,* The MIT Press(1998).

[12] 田中 雄介, 吉田 武稔: フローショップ・スケジューリング問題への強化学習の適用. 人工知能学会知識ベースシステム研究会資料 SIG-KBS-9901, pp.13-18(1999).

[13] Zhang, W., and Dietterrich, T.G.: A reinforcement learning approach to job-shop scheduling. *Proc. of the fourteenth international joint conference on artificial intelligence,* pp.1114-1120(1995).

発表論文

[1] 田中 雄介, 吉田 武稔: フローショップ・スケジューリング問題への強化学習の適用. 人工知能学会知識ベースシステム研究会資料 SIG-KBS-9901,pp.13-18, 東京,6 月 21 日,(1999).

[2] Yusuke Tanaka, Taketoshi Yoshida: An application of reinforcement learning to manufacturing scheduling problems. 1999 IEEE international conference on systems, man and cybernetics, Tokyo, Japan, Oct.12-15, 1999(Accepted).

[3] 田中 雄介, 吉田 武稔: 3 機械フローショップ・スケジューリング問題への強化学習の適用. 情報処理学会第 59 回全国大会, 岩手県立大学, 9 月 28-30 日, 1999(提出済).

[4] 田中 雄介, 吉田 武稔: 生産スケジューリングへの強化学習の適用. 計測自動制御学会システム情報部門シンポジウム, 高知工科大学, 11 月 11,12 日, 1999(提出済).

[5] 吉田 武稔, 田野 勇二, 田中 雄介: サプライチェーンマネジメントのモデル化に関する考察. 計測自動制御学会システム情報部門シンポジウム, 高知工科大学, 11 月 11,12 日, 1999(提出済).