Soar アーキテクチャのための強化学習に基づくプロダクションルール獲得機構とその応用

保知 良暢† 伊藤 孝行† 大囿 忠親† 新谷 虎松†
Yoshinobu BOCHI Takayuki ITO Tadachika OZONO Toramatsu SHINTANI
† 名古屋工業大学大学院 工学研究科

1 はじめに

予め知識を与えることが困難または知識がない状況下で，帰納的に知識を獲得して問題を解決するアーキテクチャに関する研究は多く行われている．帰納的な教師なし学習手法として強化学習 [Sutton 98] が挙げられる．強化学習は知覚情報と行動との組み合わせからボトムアップ的に政策を獲得するため，人が考えていなかったような行動を獲得できる可能性がある．しかし強化学習に知識を与えることは困難であるため，自明な結果しか獲得できないことが多い．実用的なシステムでは学習した結果から行動決定に有用な情報を取り出す必要がある．学習の結果からボトムアップ的にルールを獲得できれば，強化学習を実行する部分を制限することができ計算量は減少すると考えられる．強化学習を必要な時に実行し，学習結果で得たルールや人の持つ知識を利用した問題解決方法として Soar を利用する [堀 94]．Soar の利点として，第一に記号レベルで様々な種類の知識を表現できる点が挙げられる．ユーザは問題空間，探索に必要なヒューリステック，観測や行動に対して自明な知識等をプロダクションルールとして容易に記述することができる．

本論文では，強化学習機構を備えた Soar エージェントを提案する．Soar で問題解決に行き詰まった場合，強化学習機構を呼び出し，学習の結果とユーザが与えた背景知識からプロダクションルールを生成する． Soar は得られたプロダクションルールを用い問題解決が可能となる．

2 相関性を考慮した強化学習結果からのプロダクションルール生成

強化学習の結果，価値関数 $Q : S \times A \rightarrow \Re$ が計算される．ここで S は状態の集合，A は行動の集合を表す．価値関数 Q からプロダクションルールを生成する方法について述べる．各状態において，行動に対する価値の大小関係に基づき以下の式 (4)，式 (5) および式 (6) の 3 種類の選好性を決定する．

$$\langle s, a, > \rangle \quad \textbf{where } a = \underset{a' \in A}{\operatorname{argmax}} Q(s, a') \tag{1}$$

$$\langle s, a, < \rangle \quad \textbf{where } a = \underset{a' \in A}{\operatorname{argmin}} Q(s, a') \tag{2}$$

$$\langle s, a, > a' \rangle \quad \textbf{where } \{Q(s, a) > Q(s, a')\} \tag{3}$$

ここで $s(\in S)$ は状態，$a(\in A)$ は行動を表す．式 (1) は行動 a の価値が最も高い状態と行動の組，式 (2) は行動 a の価値が最も低い状態と行動の組，式 (3) は 2 つの行動 a, a' を比較した時，行動 a の価値が a' よりも大きい状態と行動の組を表す．ここで 1 つの状態と行動は属性と属性値のリストからなる．

提案する Soar エージェントでは，上で定義した状態，行動および選好性の組の集合から相関ルールを抽出する．抽出する相関ルールは以下の 3 つの条件全て満たすものとする．式 (4)，式 (5) および式 (6) は相関ルールを抽出するための尺度として一般的に用いられている [Zhang 02]．本論文では，前件部 X に状態に関する属性と属性値のリストが現れ，後件部 Y に行動と選好性が現れるルール $(X \rightarrow Y)$ に限定し抽出する．

$$p(Y \cup X) \geq minsupp \tag{4}$$

$$p(Y|X) \geq minconf \tag{5}$$

$$p(Y \cup X) - p(Y)p(X) \geq mininterest \tag{6}$$

ここで $0 \leq minsupp$，$minconf$，$mininterest \leq 1$ はユーザが定めるパラメータである．$p(X)$ は集合内で X を満たす確率，$p(Y|X)$ は X の元で Y を満たす条件付き確率を表す．確率は状態，行動および選好性の組の集合から計算する．

式 (4) の条件で $minsupp$ 以上の確率で成立するルールを抽出する．例えば，状態，行動および選好性の組の集合の大きさが $|D|$ とした場合，10 組以上で成立するルールを抽出したい場合 $minsupp = \frac{10}{|D|}$ となる．式 (5) の条件では信頼度が $minconf$ 以上であるルールに絞り込む．ユーザはルールの信頼度を 0〜1 の範囲で指定する．式 (6) の条件ではルール $X \rightarrow Y$ が X と Y に従属している度合いが $mininterest$ 以上であるルールに絞り込む．$mininterest$ の上限は $mininterest \geq minsupp - minsupp^2$ である．また $mininterest = 0$ とは X と Y は独立しており，相関性はないことを示す．相関ルール抽出において correlation に関するルールの獲得が困難であるという欠点がある．correlation に関する困難性とは，式 (4) と式 (5) だけでは属性間の独立性を考慮しないため無意味なルールが現れることに起因する．本論文では式 (6) を用いることで無意味なルールを削除する．

未知の環境に対応できるルールを得るためには，観測状態や行動に関する意味を与える必要がある．提案する Soar エージェントは，ユーザが入力した知識を順次追加し，強化学習の結果を表現するルールを編集する．ユーザの持つ知識を用いてエージェントの持つプリミティブな情報に対して解釈を与える．提案する Soar エージェントでは，1 つの状態に対する解釈とエージェントの行動に対する解釈を背景知識として扱う．背景知識はプロダクションルールで与える．ユーザが与えた知識はデータベースに保存される．将来強化学習を実行し相関ルールを抽出した時に，提案する Soar エージェントは背景知識のデータベース内のルールを利用して，プロダクションルールを編集する．

3 倉庫番問題における評価実験

提案する Soar エージェントの有効性を確認するために図 1 に示す倉庫番問題において実験を行った．エージェントは自身の位置，荷物 b の位置，ゴール g の位置，壁の位置，経過時間を知覚し，隣接する上下左右のマスに移動できる．エージェントは荷物と隣接している時に荷物を押すことができる．但し荷物を引くことはできない．荷物とエージェントは壁を

Production Rule Generation Unit based on Reinforcement Learning on Soar Architecture
†Graduate School of Engineering, Nagoya Institute of Technology

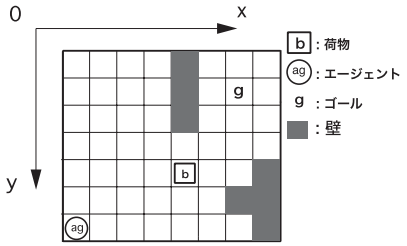

図 1: 倉庫番問題の例

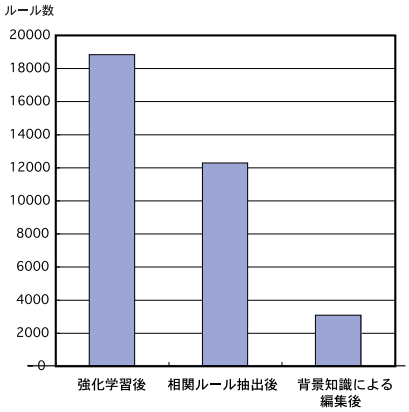

図 2: 図 1 の盤面で得られるプロダクションルール数

通り抜けることはできず，また同一マスに複数の物体が存在することはできない．エージェントは全ての荷物をゴール位置まで押して運ぶことを目標とする．

倉庫番問題の盤面は無数に考えることができる．本論文の目的は，少ない盤面を経験することで未知の盤面でも利用可能なルールを獲得することである．

本論文では (a) 強化学習後，(b) 相関ルール抽出後および (c) 相関ルールと背景知識から生成した後についてルールの比較を行った．ルール抽出に利用するパラメータの設定として，$minsupp = 5.3 \cdot 10^{-5}$，$minconf = 0.8$ および $mininterest = 3.0 \cdot 10^{-5}$ とした．背景知識として 32 個のプロダクションルールを入力した．結果を図 2 に示す．(a) でのルールの数は 18816 となった．(a) では経験した全てのルール数，つまり (経験した状態数) × (行動数) × (選好性の種類) という単純なルールが全て現れるため，非常に多くなっている．(b) でのルール数は 12265 となった．更に，強化学習の結果から相関ルールを抽出し背景知識を与えることで，(c) でのルール数は 3066 となった．ルール数は減小しても問題解決における性能に差はないため，本方法で適切なルールが獲得できたと考えられる．しかし，汎用的に利用できるルール以外に不要なルールや冗長なルールも存在した．ユーザは全てのルールについて吟味することは多大な労力を伴う．そこでルールを洗練する為に Soar のチャンキング機能を用いた．チャンキング機能によりどのルールを問題解決に利用するかを決定できる．

問題解決において困難な点は，荷物がゴール位置以外で押すことができなくなってしまった場合，それ以降はデッドロックとなり問題を解くことができなくなることである．提案する Soar システムではデッドロックに至る行動は抑制可能である．図 3 はデッドロックとなる行動の選好性が最も低いことを表すプロダクションルールを示す．盤面の一番下の列にゴールがないというルールと，エージェントが荷物を下に押すというルールが背景知識として与えられている．与えた背景知識と強化学習の結果から,「盤面の一番下の列にゴールがない場合，下に荷物を押す行動には最も低い選好性が付加される」というプロダクションルール (図 3) が得られた．図 3 に示すプロダクションルールは盤面が変化した場合にも適用できるものとなっている．

```
sp {generating*reward*rule
(state <s> ^superstate nil
^input_link <il> ^interpret <i>)
(<il> ^size_y Y)
(<il> ^agent <Ag> ^box <Box>)
(<B> ^y Y-2)
(<i> ^not_exist_in_the_row <i1>
(<i1> ^row Y-1)
(<i1> ^boolean true)
(<i> ^push_down <i2>
(<i2> ^agent <Ag>)
(<i2> ^box <B>)
(<i2> ^boolean true)
–>
(<s> ^output_link.action down <)
}
```

図 3: デッドロックに至るプロダクションルールの例

4 おわりに

本論文では，強化学習の結果からルールを抽出する Soar エージェントを提案した．実験結果から，政策の質を落とすことなくルール数を減小させることができた．提案する Soar エージェントは完全な自律的エージェントではなく，ユーザとのインタラクション行う．本論文で提案する Soar エージェントは，完全な自律的エージェントの構築を目標とした研究とは対照的である．

本論文では，相関ルール抽出方法で一般的に用いられている尺度を利用した．相関ルールの抽出には，可能な相関ルールを全て列挙する時に多くの計算量が必要となる．探索空間の枝狩りに関する研究は多く行われており，提案する方法においても同様の枝狩りを適用できる．また，本方法では現在属性と属性値の連言のみを扱っている．より柔軟なルールの抽出方法が必要であると考えられる．相関ルール抽出で得た結果をユーザの持つ背景知識で編集する利点は，(1) ユーザにとって理解の容易な形にする，(2) ルールの汎化という 2 点が上げられる．また，チャンキング機能により問題解決に必要なプロダクションルールを判断する．この点においても Soar を利用することは有用であると考える．

今後の課題としては，柔軟なルール抽出を可能とする方法の開発と大規模なシステムへの適用が挙げられる．

参考文献

[Sutton 98] Sutton, R. S. and Barto, A. G.: *Reinforcement Learning-An Introduction-*, The MIT Press (1998).

[Zhang 02] Zhang, C. and Zhang, S.: *Association Rule Mining : Models and Algorithms*, Springer (2002).

[堀 94] 堀, 池田：小特集「Soar プロジェクト」にあたって, 人工知能学会誌, Vol. 9, No. 4, pp. 478–504 (1994).