プログラム合成対象言語 Pro5Lang のための 行動価値関数圧縮アルゴリズム

Action-Value Function Compression Algorithm for the Program Synthesis Target Language Pro5Lang

一杉裕志[1*] 中田秀基[1] 高橋直人[1] 竹内泉[1] 佐野崇[2]
Yuuji Ichisugi[1] Hidemoto Nakada[1] Naoto Takahashi[1] Izumi Takeuti[1] Takashi Sano[2]

[1] 産業技術総合研究所 人工知能研究センター
[1] National Institute of Advanced Industrial Science and Technology (AIST), AIRC
[2] 東洋大学 情報連携学部情報連携学科
[2] Faculty of Information Networking for Innovation And Design, Toyo University

Abstract: We propose an algorithm for compressing the action-value function of Pro5Lang, a target language for program synthesis. In the future, we plan to compress the experience history of AGI agents so that they can acquire Pro5Lang programs autonomously. The algorithm is similar to the k-means method, but the distance and the cluster center calculation method are specific to the features of the input data and our application, leading to strong generalization capabilities. We tested the prototype implementation on artificial data that mimics experience history of agents.

1 はじめに

我々は再帰的強化学習・生成モデル・プログラム合成の３つを中核技術とした脳型ＡＧＩアーキテクチャの早期実現を目指している [1]。その一環として、プログラム合成対象言語 Pro5Lang を設計した [2]。Pro5Lang は、再帰的強化学習 RGoal [3] を用いてエージェントの報酬を最大化するプログラムを合成するための、合成対象言語である。本稿ではプログラミング言語 Pro5Lang のプログラムを自律的に獲得することを目的とした、行動価値関数のテーブル圧縮アルゴリズムについて述べる。

我々の以前の研究 [4] では、特殊な２層ベイジアンネットを用いてデータを圧縮するための予備実験を行ったが、現状では大規模化可能な推論・学習アルゴリズムが未完成であるという問題がある。そこで本稿ではベイジアンネットを用いない圧縮アルゴリズムを提案する。アルゴリズムは k-means 法に似ているが、距離やクラスタ中心の計算方法が、入力データの特徴と我々の用途に特化しており、強力な汎化能力をもたらす。

本稿は以下のような構成になっている。2 節で Pro5Lang の概要、3 節で行動ルール獲得機構の全体構想を述べた後、4 節で提案アルゴリズム、5 節で評価、6 節で関連研究について述べ、最後に 7 節でまとめを述べる。

*連絡先：産業技術総合研究所
茨城県つくば市梅園１－１－１ 中央第１
E-mail: y-ichisugi@aist.go.jp

2 Pro5Lang の概要

Pro5Lang は AGI エージェントの思考・行動を制御するプログラムを表現するためのプログラミング言語である。Pro5Lang のプログラムは固定長数値ベクトルの集合で符号化される。このプログラムを、AGI エージェントが自律的に獲得するアーキテクチャの実現を目指してる。

Pro5Lang のプログラムは、再帰的強化学習 RGoal の行動価値関数 $Q(s, g, a)$ を圧縮表現した**行動ルール**の集合である。エージェントは、現在のレジスタの状態 s およびサブゴールの値 g を、各行動ルールが持つパターン (s, g) とパターンマッチングし、マッチする行動ルールの中から価値の高いものを非決定論的に 1 つ選択し、その行動 a を実行する。

Pro5Lang の詳細については [2] を参照されたい。

3 構想中の行動ルール獲得機構

本節では、我々が早期実現を目指す脳型 AGI アーキテクチャにおいて、実装を構想している行動ルール獲得機構について概要を述べる。

エージェントは基本的には強化学習アルゴリズムを用いて自分自身の経験から知識を得る。しかし、実世界で動作すべき生物や AGI にとっては、**やみくもな試行錯誤だけでは有用な知識を得ることは不可能**である点が問題となる。実世界においては状態行動空間が極めて広い一方で、高い報酬を得る機会が極めて少ないためである。

ヒトの脳による行動ルール獲得には様々な機構が関わっていると思われる。それを再現すべく構想中の機構を以下の 4 つに分類して概要を説明する。

1. **自動応答機構 :** 刺激に対して自動的に応答するために、身体に作り込まれる機構。目立つ物体に視線を向ける、危険な物体から身をそらす、苦痛や喜びなどを示す発声をする、目の前の状況に似た過去の状況を想起する、など。エージェントが行動ルール獲得していない段階においては、自動応答だけが試行錯誤を生み出す原因となる。

2. **経験評価・保持機構 :** エージェントに作り込まれた報酬関数を用いて、自分自身の行動結果の価値を計算し、結果を経験履歴として一定期間保持する機構。

3. **帰納推論機構 :** 経験履歴に基づいて、汎用性の高い行動ルールを獲得する機構。

4. **演繹推論機構 :** すでに得られた汎用的な行動ルールや宣言的知識を組み合わせて、新たな行動を生み出す機構。模倣や対話を通じた知識獲得も、演繹推論機構を利用する。例えば「他者がこう実行したのだから自分もそれを実行した方がよいはずだ」「親がそう言っていたから正しいはずだ」といった推論も一種の演繹推論であると解釈する。模倣・対話は、やみくもな試行錯誤を必要としない効率的な行動ルール獲得手段である。

本稿で述べる行動価値関数圧縮アルゴリズムは、この中の帰納推論機構に属し、圧縮によって経験履歴にない状況に対しても行動ルールを汎化させることを目的としている。

4 提案アルゴリズム

4.1 圧縮の対象となる入力データ

提案アルゴリズムはエージェントの経験履歴を表現する入力データを、行動ルール集合に圧縮する。

今回のプロトタイプ実装では、扱う入力データは以下の構造を持つ。入力データは、状態行動対を表現する固定長の数値ベクトルと、その価値の組である。数値ベクトルの要素の値は 0 〜 99 の整数値である。

1. k 個の参照ベクトルを初期化する。
2. 定数回（または収束するまで）下記を繰り返す。
2.1. すべてのデータを、最も近い参照ベクトルに割り当てる。
2.2. 各参照ベクトルに割り当てられたデータの集合を圧縮したものを新たな参照ベクトルとする。

図 1: 提案アルゴリズムの骨格。初期化方法、距離の定義、データの圧縮方法にバリエーションがある。詳細は本文参照。

価値は任意の実数値であるが、 Pro5Lang の価値学習が進むと、価値は整数に近い値に収束するという特徴がある。（この特徴を次節で述べるアルゴリズムで利用している。）

4.2 アルゴリズム

提案アルゴリズムは図 1 のような構造をしている。これは k-means 法の構造と同じである。k-means 法は数値ベクトル集合を k 個のクラスタに分類するアルゴリズムであるが、数値ベクトル集合を k 個の数値ベクトルで圧縮表現するためのアルゴリズムとも言える。k-means 法は一種の EM アルゴリズムである。我々は大脳皮質も EM アルゴリズムを用いて学習すると予想しており [4]、k-means 法は大脳皮質の学習アルゴリズムと類似していると考えている。

提案アルゴリズムでは、各クラスタの中心を表現するデータ構造を**参照ベクトル**と呼ぶ。参照ベクトルは入力データのベクトル部と同じ長さの**パターン**および価値を表す実数値から構成される。パターンの要素は整数値、ワイルドカード、変数のいずれかである。（ただし今回のプロトタイプ実装では変数は扱わない。）

図 1 のアルゴリズムには初期化方法、距離の定義、データの圧縮方法という 3 つの主な構成要素があり、以下に述べるように、それらの各々にバリエーションがある。

初期化は 2 つの方法 (I1 と I2) を実装した。I1 は入力データの中から重複しないようにランダムに選択したものを初期値とする。I2 は、できるだけ参照ベクトルどうしの距離が遠くなるように初期化する。具体的には、まず 1 つ目の参照ベクトルを入力の中からランダムに選んだあと、すでに選んだ他の参照ベクトルから最も遠いデータを入力から順に選択し、参照ベクトルに追加する。これを参照ベクトルが k 個選択されるまで繰り返す。（この初期化方法は、 k-means 法の初期化方法の改良の 1 つである k-means++ 法 [5] にヒントを得ている。）

距離は 2 つの方法 (D1 と D2) を実装した。D1 は参

照ベクトルにマッチしない要素数、一般度、価値の二乗誤差の順に比較して判定する方法、D2 は価値の二乗誤差、参照ベクトルにマッチしない要素数、一般度の順に比較して判定する方法である。以下に具体的に説明する。

まず、参照ベクトル v とデータ x との間の距離に相当する情報 $d(x,v)$ を下記の三つ組で表現する。

$$d(v,x) = (\text{マッチしない要素数}, \text{一般度}, \text{価値の二乗誤差})$$

ただし**一般度**とはワイルドカードを使ってマッチした要素数である。D1 では、まずマッチしない要素数を比較することで 2 つの距離の大小関係を決める。その値が同一ならば一般度の大小で、一般度も同一ならば価値の二乗誤差の大小で決める。以下に例を示す。ここで $(1,2):2$ はデータのベクトル部が (1,2) 、価値が 2 であることを表す。 “$*$”はワイルドカードである。

$$d(\ (*,*):3\ ,\ (1,2):2\ ) = (0,2,1)$$
$$d(\ (1,2):3\ ,\ (1,2):3\ ) = (0,0,0)$$
$$d(\ (0,2):3\ ,\ (1,2):3\ ) = (1,0,0)$$

D1 では距離の大小関係は、下記のようになる。

$$(0,0,0) < (0,2,1) < (1,0,0)$$

一方 D2 では下記のようになる。

$$(0,0,0) < (1,0,0) < (0,2,1)$$

圧縮方法は、現在のところ 1 通りの実装のみである。クラスタに割り当てたすべてのデータに共通な要素はそのまま、1 つでも違うものがある要素はワイルドカードにする。例えば割り当てられた 3 つのデータのベクトル部が (1,2),(1,2),(9,2) であれば、参照ベクトルのパターンは (*,2) とする。参照ベクトルの価値は、割り当てられたデータの価値の平均とする。

なお、アルゴリズムのステップ 2.2 で参照ベクトルに割り当てられたデータが 1 つもないことがある。その場合はその参照ベクトルを入力データからランダムに選択した値に入れ替えるものとする。

5 評価

図 2 に示したように、人工データを用いてアルゴリズムの動作を確認した。

アルゴリズムの各構成要素が性能にどれだけ貢献するかを確認するために、構成要素の異なる組み合わせで性能を評価した（表 1）。汎化誤差は、テストデータのベクトル部と圧縮後のテーブルを用いて予測される価値と、真の価値との間の二乗誤差の平均で評価する。汎化誤差が 0 になれば、損失なしに行動価値関数が圧縮されたことを意味する。訓練誤差は訓練データを用いて計算した誤差である。（ただしニューラルネットワークと違い、訓練誤差を直接最適化しているわけではない。）表 1 の数値は、異なる訓練データ・テストデータに基づいた１００回の計測結果の平均である。

訓練データ（圧縮アルゴリズムへの入力データ）・テストデータ（汎化誤差の計算に用いるデータ）の生成に用いる生成ルールは図 2(a) で、ルール数は 8 である。この生成ルールは、我々のこれまでの Pro5Lang を用いた研究の経験をもとにして、圧縮対象の行動価値関数の性質をできるだけ再現するように定義されている。ベクトル部の第一要素はサブゴールを模しており、それ以降の要素はレジスタの状態を表すパターンを模している。価値は、特殊なパターンほど大きな値（0 に近い負の値）になるという特徴を持つ。

実験条件は以下の通りである。クラスタ数 k は生成ルール数の 2 倍、訓練データ数は $k \times 5$ 個、テストデータ数は $k \times 50$ 個を生成した。圧縮アルゴリズムの繰り返し回数は１０回で固定とした。

実験結果を表 1 に示す。今回の実験においては、 I2 + D2 の組み合わせでほぼ理想的な圧縮結果となった。図 2(c) は圧縮結果の一例である。なお、I2 + D2 ではほとんどの場合で初期化後 1 ステップで訓練誤差が収束することが観察された。他の組み合わせでも 2 ステップ以内にほぼ収束した。

6 関連研究

DQN [6] は、ニューラルネットワークを用いて行動価値関数の関数近似を行う。一般にニューラルネットワークのような識別モデルは事前知識の作り込みがしにくく、学習には膨大な試行錯誤が必要となる。それに対しベイジアンネットのような生成モデルはモデルの形やパラメタの事前分布の形に事前知識を作り込みやすく、それにより、 AGI に期待されるような、少ない経験からの知識獲得を実現できる可能性がある。将来は関数近似に特殊なベイジアンネットを用いるのが我々の計画 [4] である。

RRL(Relational Reinforcement Learning) [7] は、強化学習と帰納論理プログラミングを組み合わせたシステムであり、タスクを試行錯誤によって繰り返し解きながら、行動価値関数を圧縮表現する論理決定木と呼ぶ構造を学習する。行動価値関数を記号表現を使って圧縮することで極めて強力な汎化能力を得ようとする点で、我々の研究と目標は同じである。我々が用いようとしているパターンマッチとサブルーチンを組み合わせた表現方法は、論理決定木よりは表現力が強いものになる。

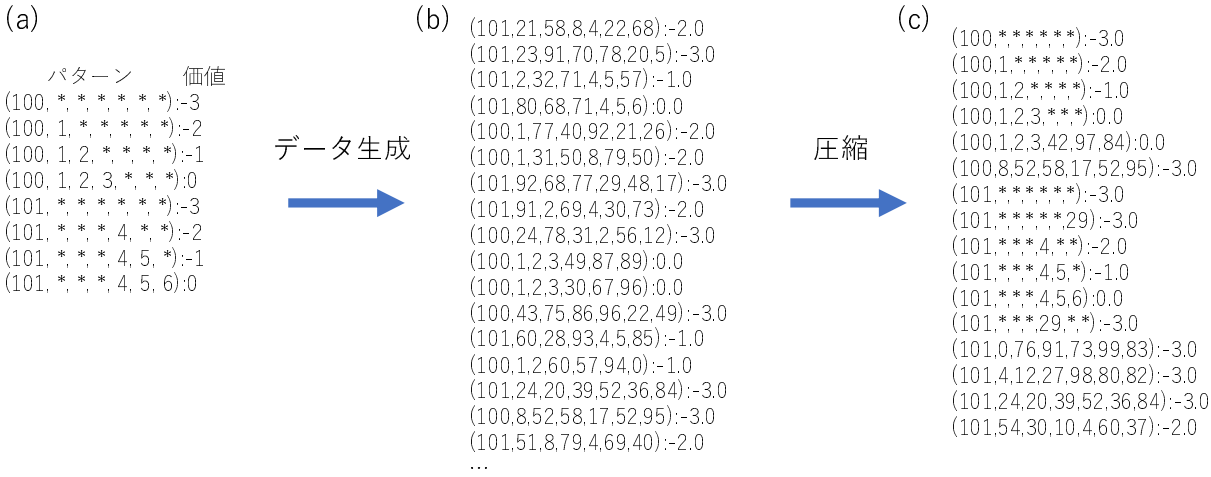

図 2: 人工データによる動作確認手順。(a) Pro5Lang のプログラムを模した生成ルール集合を手で与える。“*” は 0 から 9 9 の間の任意の整数を表す。(b) 与えた生成ルールから人工データを生成する。(c) 人工データを圧縮アルゴリズムで圧縮した結果の参照ベクトル集合が得られる。

表 1: 実験結果

初期化・距離	訓練誤差	汎化誤差
I1 + D1	0.12	0.15
I2 + D1	0.39	0.51
I1 + D2	0.20	0.24
I2 + D2	0.01	**0.01**

7 まとめ

プログラム合成対象言語 Pro5Lang の行動ルールをエージェントの経験から自律的に獲得させるために必要な要素技術の 1 つとして、行動価値関数圧縮アルゴリズムを提案した。

本研究で得られた初期化や距離に関する知見は、将来のベイジアンネットを使った行動価値関数圧縮アルゴリズムの設計にも役立つと思われる。

今後はアルゴリズムに必要な改良を加えつつ、エージェントの自律的な行動ルール獲得のデモの実現を目指す。

謝辞

本研究は JSPS 科研費 JP22K12188 の助成を受けたものです。

参考文献

[1] 一杉裕志. 報酬最大化原理にもとづく脳型ＡＧＩアーキテクチャの構想. 第 18 回 人工知能学会 汎用人工知能研究会 (SIG-AGI), 2021.

[2] 一杉裕志, 中田秀基, 高橋直人, 竹内泉, 佐野崇. 汎用人工知能のためのプログラム合成対象言語 Pro5Lang のエピソード記憶機構. 第 20 回 人工知能学会 汎用人工知能研究会 (SIG-AGI), 2022.

[3] Yuuji Ichisugi, Naoto Takahashi, Hidemoto Nakada, and Takashi Sano. Hierarchical reinforcement learning with unlimited recursive subroutine calls. In *Artificial Neural Networks and Machine Learning – ICANN 2019: Deep Learning*, pp. 103–114, Cham, 2019.

[4] 一杉裕志, 中田秀基, 高橋直人, 佐野崇. 脳の自律的プログラム合成機構のモデルに向けて：2 層ベイジアンネットによる記号処理命令の獲得・実行機構. 第 15 回 人工知能学会 汎用人工知能研究会 (SIG-AGI), 2020.

[5] David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, p. 10271035, USA, 2007. Society for Industrial and Applied Mathematics.

[6] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. Human-level control through deep reinforcement learning. *Nature*, Vol. 518, No. 7540, pp. 529–533, 2015.

[7] Sašo Džeroski, Luc De Raedt, and Kurt Driessens. Relational reinforcement learning. *Machine learning*, Vol. 43, No. 1, pp. 7–52, 2001.