関数最適化問題における並列分散型学習オートマトン計算法の適用

学生員 池 坊 繁 屋 (広島国際学院大)
正 員 銭 飛 (広島国際学院大)
正 員 平 田 廣 則 (千葉大)

A Parallel Distributed Learning Automaton Computing Model for Function Optimization Problems

Shigeya Ikebou, Student Member, Fei Qian, Member (Hiroshima Kokusai Gakuin University), Hironori Hirata, Member (Chiba University)

We propose a learning automaton computing method(LACM) with a genetic algorithm(GA) for the multi-variable function optimization problems and complete a distributed parallel implementation. The simulation results show that LACM has better computing accuracy and higher convergence property than other traditional methods.

キーワード : 学習オートマトン, 遺伝的アルゴリズム, 関数最適化問題

1. はじめに

本稿では多変数関数最適化問題 (以下 *FOP* と略す) $f(\mathbf{X}) \rightarrow \text{opt}, \mathbf{X} \in R^n$ について考え, 新しい学習オートマトン (以下 *LA* と略す)⁽¹⁾ 計算法を構築する。この問題に従来の *LA* を適応する際は, 関数の解空間が大きくなるにつれて *LA* の出力数が増えるので, 強化法に微小確率の修正が要求され, 収束するまでに多大な計算時間がかかってしまう。

そこで, 本稿では, *LA* の出力決定機構に遺伝的操作を導入することにより解空間を圧縮し, サイズの小さい探索空間を形成しながら, 解の探索を行う。更に, 学習オートマトンチームモデル⁽²⁾ を用いて, 計算法の並列分散化を図る。提案する手法の有効性を検証するために, 実際の疎結合並列分散処理環境に提案手法を実装し, 関数最適化問題の計算機実験を行ない, 解の精度や収束速度について従来の手法と比較検討をする。

2. 分散型学習オートマトン計算モデル

本稿では, オートマトンの出力数を圧縮するために, 遺伝的オペレータを学習オートマトン (LA^{GA} と略す) の出力決定機構に導入する。

LA^{GA} の定義は従来とほとんど同じであり, 確率オートマトン SA^{GA} とその同定の対象である未知環境 RE から構成される。

$$SA^{GA} \triangleq \{R, Y, \Phi, \mathbf{p}, o, T\} \dots\dots\dots (1)$$

ここで, R は入力集合, $r \in R$ は強化信号と呼ぶ。 Y は出力集合, Φ は状態集合, $\mathbf{p}$ は状態遷移確率分布または出力選択確率分布であり, o は出力関数である。 T は

$\mathbf{p}$ を修正する強化法であり, 強化信号 r に応じて対応する出力を選択する確率を増減する。遺伝的操作を取り入れた SA^{GA} は, $o: G(Y) \rightarrow Y^{GA} \subset Y$, G は遺伝的オペレータであり, $|Y^{GA}| \ll |Y|$ である。この場合, $\mathbf{p}$ は Y^{GA} 上に定義された確率分布となる。遺伝的オペレータにより, 元の出力集合から要素数の少ない対象出力集合を形成し, 強化法での微小確率修正問題を緩和する。

一方, 未知環境の定義は従来と同様, 以下のように表現される。

$$RE \triangleq \{Y, R, C\} \dots\dots\dots (2)$$

ここで, Y は SA^{GA} からの入力集合, 明らかに, $Y = Y^{GA}$ である。 R は出力集合であり, C は R 上で定義される確率分布である。ここでは, $R \triangleq \{r_w, r_p\}$ とし, r_w, r_p をそれぞれリワード応答とペナルティ応答と呼び, C を入力に対するペナルティ応答確率と呼ぶ。

LA を *FOP* に適用するために, *FOP* の「目的関数」を *LA* の「未知環境」とし, 目的関数の変数の定義域上で離散化し *LA* の出力に対応させる。並列分散的に計算を行うために, 各変数に 1 個または複数個の SA^{GA} を用いて関数の値の変化を監視しながら探索点の推定を行う。

n 変数の *FOP* に対して, 本稿では, 図 1 に示す学習オートマトンチームモデルを用いる。

LA のチームは n 個の $SA(SA^1, \dots, SA^n)$ からなり, チームの出力集合は各々の SA の出力集合の直積からなる。各オートマトンにとってチーム行動中の目標とは, あるチーム出力 (混合出力) を見出し, 自身が未知環境から受ける平均ペナルティを最小にしようとするのである。

図1 学習オートマトンチームモデル
Fig. 1. A learning automaton team model.

図2 目的関数と探索空間の分散表現
Fig. 2. An objective function and the representation of search space.

前述した遺伝的オペレータを有する LA を利用した並列分散型 LA 計算法のアルゴリズムを以下に示す。
初期設定 $SA^1, \dots, SA^n$ を n 個の変数に割当て、各変数を離散化し $SA^1, \dots, SA^n$ の出力集合に対応させる。
Step1: $SA^1, \dots, SA^n$ において、ランダムに各探索空間を形成する。
Step2: 形成された各探索空間に対して LA の動作を行ない、各変数の推定を行う。
Step3: $SA^1, \dots, SA^n$ において、出力集合を形成するための遺伝的操作 (選択, 交叉, 突然変異) を行なう。
Step4: 終了条件が満たされるまで 2-3 を繰り返す。

3. 計算機実験

本稿で提案している手法を検証するために、以下の 2 変数 *Griewank* 関数最適化問題について考える。

$$\min_{x,y \in [-10,10]} \left\{ f(x,y) = \frac{x^2 + y^2}{50} + \cos x \cos \frac{y}{\sqrt{2}} + 1 \right\} \quad (3)$$

並列分散的に最適解を探索するために探索空間を 4 つの部分空間に分割し、各部分空間に対して 2 つのオートマトンを用いる (図 2)。ある時刻において、4 組 (8 個) のオートマトンは、各自の出力選択確率分布に従い出力集合から出力を決定し、それを推定値として (3) 式に代入して、それぞれの目的関数値を求める。そして、目的関数値をペナルティ応答確率として SA^{GA} への応答を決める。但し、未知環境におけるペナルティ応答確率分布は (3) 式の関数値を $[0,1]$ に正規化したものを用いる。

図3 解の精度
Fig. 3. Solution accuracy.

図4 収束時間
Fig. 4. Convergence.

ここで、提案手法 (LA^{GA}) の有効性を示すために探索空間の広さを 1) $0 \leq x,y \leq 31$, 2) $0 \leq x,y \leq 255$, 3) $0 \leq x,y \leq 1023$ と変え、収束速度を比較するために LA^{GA} の学習回数と世代数の積を LA の学習回数と同じにしている。また、実験環境として 4 ノード構成の MPI(Message-Passing Interface) 環境を用い、提案手法の実装を行っている。

実験結果を図 3, 図 4 に示す。図 3 は各探索範囲より得られた値と最適値との誤差、図 4 は各探索範囲における収束時間の比較を示している。

解の精度については、探索空間が大きくなっても提案手法は良い結果が得られていることが分かる。一方、収束時間についても著しく良い結果が得られている。

4. まとめ

本稿では、関数最適化問題に対して、遺伝的操作と学習オートマトンチームモデルを用いた並列分散型学習オートマトン計算法を提案している。

実験検証の結果より、本手法は LA^{GA} の出力数に対して良好な計算精度や高い収束性を持つことが分かる。

(平成 12 年 03 月 13 日受付)

文 献

(1) K.S. Narendra & M.A.L. Thathachar: Learning Automata - An Introduction, Prentice-Hall International(1989).
(2) 銭 飛・平田廣則: 分散型強化学習システム: 学習オートマトンのチームモデル, 電気学会論文誌, Vol. 115-C, No. 5, pp.787-793 (1998).