

大規模最適化問題における変数間依存性を考慮した協調共進化計算

吉川 太智[†] 中田 雅也[†]横浜国立大学[†]

1 はじめに

メタヒューリスティクス分野では、社会インフラや産業などの最適化対象の大規模化に伴って、高次元決定変数ベクトルを扱う大規模最適化^{††}への対処が重要課題である。この課題に対して、協調共進化は変数分割を通して低次元ベクトルを扱う部分問題を最適化することで、次元増加に対するスケラビリティを改善できる¹⁾。しかしこの方法では、分割した部分問題間に存在する変数間依存性を考慮せずに最適化を行うことになる。これは協調共進化の重大な問題として知られている。

この問題に対し、部分問題間の相関を最小化するなど、ほとんどの既存研究は「変数間依存性の影響を低減するように変数分割する」方法論を構築している¹⁾。しかし、この方法論では同依存性を最小化する最適化問題を別途解く必要があるだけでなく、線形的な変数分割のみでは問題に潜在する非線形な同依存性に対応できないという課題がある。したがって、既存方法論は解探索の非効率性に加え、変数分離不可能な問題では性能が低下する。

本稿では「変数分割後に変数間依存性を推測して最適化する」方法論を用いた協調共進化を提案する。これは、変数分割では完全に解消できない変数間依存性を推測し、最適化を促進するための探索情報として扱う点に意義がある。このために、変数間依存性が反映された推測個体に基づき解個体を生成する。

2 協調共進化法の概要

本稿では、 n 次元の問題を m 個の部分問題に等分割する最も一般的な協調共進化を用い、関数最小化問題を扱う。Fig. 1 (a)に示すように、分割された n/m 次元の部分問題毎に部分解を生成し、部分問題ごとの最良部分個体を保存した(世代数 t に対する)context vector $\mathbf{b}^t = (\mathbf{b}_1^t, \dots, \mathbf{b}_m^t)$ を用いて評価する。具体的には、世代数 t について、 i 個目の部分問題における個体集合 $\mathcal{P}_i$ において、 j 番目の部分個体 $\mathbf{x}_{ij}^t \in \mathbb{R}^{n/m}$ が生成されたとき、 $\mathbf{x}_{ij}^t$ の評価値は評価

Cooperative Co-evolutionary Algorithm Considering Variable Dependency for Large Scale Global Optimization Problems

[†] Taichi Yoshikawa, Masaya Nakata, Yokohama National University

^{††}一般的に数百から千次元程度の問題を指す。

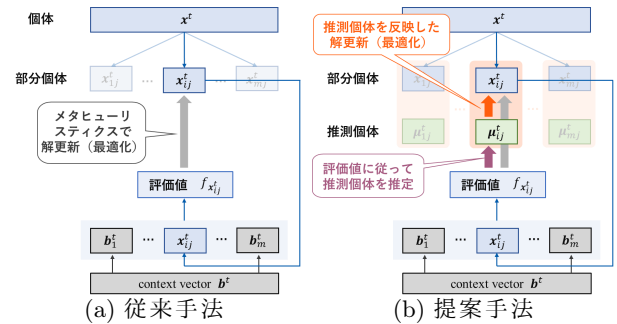


Fig. 1: 概要図

関数 $f: \mathbb{R}^n \rightarrow \mathbb{R}$ を用いて式(1)で計算され、最小化問題では現在までの最良値を下回るとき $\mathbf{b}_i^{t+1}$ を $\mathbf{x}_{ij}^t$ に置き換える。なお、 $\mathbf{x}_{ij}^t$ はGAやPSOなどを用いて生成される。

$$f_{\mathbf{x}_{ij}^t} = f(\mathbf{b}_1^t, \dots, \mathbf{b}_{i-1}^t, \mathbf{x}_{ij}^t, \mathbf{b}_{i+1}^t, \dots, \mathbf{b}_m^t) \quad (1)$$

3 提案手法

式(1)に示したように、 $\mathbf{x}_{ij}^t$ の評価は $\mathbf{b}^t$ に依存するが、 $\mathbf{x}_{ij}^t$ は $\mathbf{b}^t$ に依存せず生成される。この結果、 $\mathbf{x}_{ij}^t$ と $\mathbf{b}^t$ に存在する変数間依存性は考慮されない。そこで提案手法では、変数間依存性を考慮して $\mathbf{x}_{ij}^t$ を生成するために、 $\mathbf{b}^t$ に適した推測個体 $\mu_{ij}^t \in \mathbb{R}^{n/m}$ を推測し、 μ_{ij}^t に基づいて $\mathbf{x}_{ij}^t$ を生成する。したがって、一般的にモデル同定が困難な変数間依存性は、最適解導出を目的とする点を鑑み、 $\mathbf{b}^t$ に対し n/m 次元の変数ベクトルとして単純化できる。また、提案手法ではGrey Wolf Optimizer (GWO)²⁾を用いて $\mathbf{x}_{ij}^t$ を生成する。GWOは個体数 N を除けば1つのパラメータ(探索制御パラメータ a)のみを用いるため、パラメータ調整がボトルネックとなる実最適化問題において優れた利便性がある。また、PSOなどと比べ優れた性能を発揮することが知られている²⁾。以下では、提案手法であるCCGWO-AII (Cooperative Coevolutionary GWO Assisted by Inferred Individual)を構成する1) μ_{ij}^t を用いた解生成、2) μ_{ij}^t の更新について述べる。

1) 推測個体を用いた解生成 GWOは $\mathcal{P}$ 内の上位3個体の周辺を他の個体が探索する。協調共進化に基づく提案手法では、個体集合 $\mathcal{P}_i$ を評価値順にソートし、上位3個体 $\mathbf{x}_{i1}, \mathbf{x}_{i2}, \mathbf{x}_{i3}$ ならびに μ_{ij}^t を用いて、 $\mathbf{x}_{ij}^t$ を式(2)-(4)で更新する。

$$\mathbf{x}_{ij}^k = \mathbf{x}_{ik} - \mathbf{A}_k \cdot |\mathbf{C}_k \cdot \mathbf{x}_{ik} - \mathbf{x}_{ij}^t| \quad (2)$$

$$\mathbf{x}_{ij}' = \frac{\mathbf{x}_{ij}^1 + \mathbf{x}_{ij}^2 + \mathbf{x}_{ij}^3}{3} \quad (3)$$

$$\mathbf{x}_{ij}^{t+1} = \mathbf{x}_{ij}' + \gamma \cdot \mathbf{r}_1 \cdot (\boldsymbol{\mu}_{ij}^t - \mathbf{x}_{ij}') \quad (4)$$

γ は更新率であり、 $\mathbf{A}_k$ は $[0, a]^{n/m}$ 、 $\mathbf{C}_k$ は $[0, 2]^{n/m}$ 、 $\mathbf{r}_1$ は $[0, 1]^{n/m}$ のそれぞれ一様乱数である。なお、部分問題ごとに生成する点を除けば、簡略化した式(2)-(3)はGWOの解生成と同様である。また、式(4)より $\boldsymbol{\mu}_{ij}^t$ を考慮して $\mathbf{x}_{ij}'$ を調整し $\mathbf{x}_{ij}^{t+1}$ を生成する点が提案手法の特徴であり、 $\mathbf{b}^t$ に適した(変数間依存性を考慮した)部分個体の生成を意図している。

2) 推測個体の更新 非線形な変数間依存性におけるモデル同定の困難さを回避するために、1)分割統治法のように $\boldsymbol{\mu}_{ij}^t$ として単純化されたモデルを複数用いて同依存性を細分化して捉え、2)最適化問題の目的を踏まえ評価値を良くする変数間依存性のみを対象とする。したがって、提案手法では各部分個体に対し固有の推測個体を割り当てるとともに、各部分個体の探索結果に合わせて推測個体を逐次的に更新する。具体的には、 $\boldsymbol{\mu}_{ij}^t$ は、正規分布 $\mathcal{N}(0, a/6)$ に従う乱数 $\mathbf{r}_2 \in \mathbb{R}^{n/m}$ を用いて次式により更新する。

$$\boldsymbol{\mu}_{ij}^{t+1} = \begin{cases} \boldsymbol{\mu}_{ij}^t + \gamma \cdot (\mathbf{x}_{ij}^{t+1} - \boldsymbol{\mu}_{ij}^t) & \text{if } f_{\mathbf{x}_{ij}^{t+1}} < f_{\mathbf{x}_{ij}^t}, \\ \boldsymbol{\mu}_{ij}^t + \mathbf{r}_2 \cdot (\mathbf{x}_{ij}^{t+1} - \boldsymbol{\mu}_{ij}^t) & \text{otherwise.} \end{cases} \quad (5)$$

したがって、パーソナルベストを更新したとき($f_{\mathbf{x}_{ij}^{t+1}} < f_{\mathbf{x}_{ij}^t}$)、 $\mathbf{x}_{ij}^{t+1}$ は $\mathbf{b}^t$ に適した変数間依存性を捕捉したと扱い、 $\boldsymbol{\mu}_{ij}^t$ を $\mathbf{x}_{ij}^{t+1}$ に近づける。更新されない場合は、 $\boldsymbol{\mu}_{ij}^t$ 自体が評価値を改善する同依存性を捕捉できていないと考えられるため、 $\boldsymbol{\mu}_{ij}^{t+1}$ を $\boldsymbol{\mu}_{ij}^t$ の周辺に設定し、推測個体自体が探索をする。

4 実験

Table 1に示すように、大規模最適化におけるコンペティションで使用される15個のベンチマーク関数(IEEE CEC'2019 LSGO Benchmark Function)を用いる。なお、いずれの関数も1000次元と設定されるが、F1-F3が完全分離可能な関数、F4-F11が部分加算分離可能な関数、F12-F14が重複により変数分離不可能な関数、F15が完全変数分離不可能な関数である。**実験設定** CCGWO-AIIを、1)協調共進化に基づく通常のGWOと2)協調共進化に基づく推測個体を用いた解生成を行わないCCGWOと比較する。つまり、CCGWOは $\gamma = 0$ としたCCGWO-AIIと同じである。個体数 $N = 50$ 、最大世代数 $T = 60000$ 、分割数 $m = 10$ 、 $a = 2 - (2/T)t$ とする²⁾。CCGWO-AII固有のパラメータとして、更新率 $\gamma = 0.2$ とする。

Table 1: ベンチマーク関数の評価値の比較

Problem	GWO	CCGWO	CCGWO-AII
F1	4.3504E+10	1.0339E+11	8.9455E+10
F2	2.9308E+04	2.7573E+04	2.6041E+04
F3	2.1589E+01	2.0987E+01	2.0982E+01
F4	4.6130E+11	4.4443E+11	2.8449E+11
F5	7.3315E+06	2.5491E+07	2.4798E+07
F6	1.0609E+06	1.0402E+06	1.0405E+06
F7	1.2405E+11	8.5446E+10	1.7078E+10
F8	6.4185E+15	5.6890E+15	2.1493E+15
F9	6.3093E+08	1.9122E+09	1.7093E+09
F10	9.4104E+07	9.2592E+07	9.2390E+07
F11	1.9896E+12	6.9047E+11	4.5859E+11
F12	3.3417E+21	5.4846E+23	4.1080E+20
F13	3.8329E+11	1.2376E+11	6.5888E+10
F14	2.0554E+12	6.9060E+11	4.5672E+11
F15	5.4626E+13	2.3449E+14	3.6064E+13

実験結果 Table 1に各手法が最大世代時点で導出した最良解の評価値(25試行の中央値)を示す。また、Friedman検定およびHolm法による事後検定(Wilcoxonの順位和検定を用いた)を適用した結果、CCGWO-AIIとGWO、CCGWO-AIIとCCGWOにそれぞれ有意差($p < 0.05$)を検出したことから、CCGWO-AIIは優れた探索性能を導出することがわかる。また、同検定による有意差はないが、CCGWOはGWOと比較して10個の関数で優れた最良値を導出していることから、協調共進化によって性能が改善する傾向があると言える。しかし、完全変数分離不可能なF15をはじめとして、特に重複により変数分離不可能なF12においてCCGWOの性能はGWOよりも顕著に劣る。一方で、CCGWO-AIIは、CCGWOと比較してF6を除く14個の関数、GWOと比較してF1,F5,F9を除く12個の関数で優れた最良値を導出している。GWOと比べてCCGWOの性能が低下したF12とF15においても、最も優れた性能を導出している。以上より、CCGWO-AIIは、1)協調共進化による探索性能の改善効果を維持しつつ、2)推測個体を通して変数間依存性を考慮した解生成を行うことで、協調共進化の課題である変数分離不可能な問題においても有効であると言える。

5 まとめ

本稿では、変数分割後に変数間依存性を考慮して最適化を行うために、推測個体を用いて解生成を行う協調共進化法を提案し、その有効性を実験より確認した。今後は、分割方法を動的にした場合にも推測個体が有効的に作用する方法について検討する。

参考文献

- 1) X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, and Z. Zhu. A survey on cooperative co-evolutionary algorithms. *IEEE Trans. on Evol. Computation*, 23(3):421–441, 2018.
- 2) S. Mirjalili, S. M. Mirjalili, and A. Lewis. Grey wolf optimizer. *Advances in engineering software*, 69:46–61, 2014.