社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

生物の遺伝子制御機構に基づくネットワーク帯域幅の変動に適応可能な分散型映像分析システム消費電力最適化方式

井上誠志郎† 山内 雅明† 小南 大智† 下西 英之†† 村田 正幸†

† 大阪大学 大学院情報科学研究科 〒565–0879 大阪府吹田市山田丘 1-5
†† 大阪大学 サイバーメディアセンター 〒560-0043 大阪府豊中市待兼山町 1-32
E-mail: †{s-inoue,m-yamauchi,d-kominami,h-shimonishi,murata}@ist.osaka-u.ac.jp

あらまし 現実世界と仮想世界を高度に統合するデジタルツインの構築において、現実世界の状況を正確に認知するために大量の映像データをリアルタイムに分析する必要がある。このとき大量の映像データによるトラヒック量の増大とそれに伴う消費電力の増大が課題の一つである。これに対してエッジコンピューティング技術による分散処理が検討されているが、ネットワークの状態やアプリケーションの要求に応じて、動的に処理を分散する必要がある。我々の研究グループでは、分散型映像分析システムの消費電力最適化問題を定式化し、遺伝的アルゴリズムによる方式を提案している。ただし、遺伝的アルゴリズムでは解が収束するため、状況の変化に対応することが難しい。そこで本研究では、生物の遺伝子制御ネットワークにおいて、一つの遺伝型の変異で複数の表現型が同時に変異する特徴に着目し、遺伝子制御ネットワークに基づく分散型映像分析システムの動的制御手法を提案する。いくつかの解を表現型として記憶させた遺伝子制御ネットワークの数理モデルを用いることで、ネットワーク帯域幅が変動する環境において、記憶した解の想起による局所解からの脱出、過去の解やネットワーク構造の活用による適応能力の向上を確認した。
キーワード デジタルツイン、適応進化、最適化問題、遺伝的アルゴリズム、遺伝子制御ネットワーク

Power Consumption Optimization Method Adaptable to Network Bandwidth Fluctuation for Distributed Video Analysis System based on Gene Regulatory Network

Seishiro INOUE†, Masaaki YAMAUCHI†, Daichi KOMINAMI†, Hideyuki SHIMONISHI††, and Masayuki MURATA†

† Graduate School of Information Science and Technology, Osaka University,
1-5 Yamadaoka, Suita, Osaka, 565-0879 Japan
†† Cyber media center, Osaka University, 1-32 Machikaneyama Toyonaka, Osaka, 560-0043 Japan
E-mail: †{s-inoue,m-yamauchi,d-kominami,h-shimonishi,murata}@ist.osaka-u.ac.jp

Abstract To construct a digital twin, we have to analyze a large amount of video data in real-time to recognize real-world situations accurately. The increase in the video traffic and the accompanying increase in power consumption are problems. One solution is edge computing to distribute video analyzing tasks; it requires dynamic control of task distribution according to the network status and application requirements. Our research group has formulated an power consumption optimization problem for distributed video analysis systems and proposed a method using a genetic algorithm. However, the genetic algorithm cannot adopt to situation changes. In this study, we focus on feature of gene regulatory networks which mutate multiple phenotype bits from one-bit mutation in genotype simultaneously and propose a dynamic control method for distributed video analysis systems based on gene regulatory networks. We made a mathematical model of gene regulatory networks memorized several solutions as phenotypes. We confirmed the escape from local solutions by recalling the stored solutions and generating multiple phenotype bits mutations in dynamic network environments.
Key words Digital twin, adaptive evolution, optimization problem, genetic algorithm, gene regulatory network

1. はじめに

現実世界と仮想世界を高度に統合するデジタルツイン技術が注目を集めている。デジタルツインとは、映像やセンサを用いて現実世界から物体や環境に関するデータを収集し、デジタル世界に同じ環境をまるで双子 (ツイン) のように再現する技術である。デジタルツインの実現により、現実世界のオブジェクトやシステムの継続的な監視と最適化が可能となる。例えば、高度道路交通システム [1] やがん診断の意思決定支援ソリューション [2] といった利用シーンが考えられており、様々な業界に新たな価値をもたらすことが期待されている [3]。

デジタルツインの構築においては、現実世界の状況を正確かつリアルタイムに認知するために、現実世界から大量の映像やセンサのデータを取得し、短時間で分析する必要がある。特に映像データは、複数のカメラや LiDAR から大量のデータが生成され、ネットワークを経由してクラウドサーバ上で収集され、YOLO [4] などの AI 映像分析技術によって分析される。ユーザは、デジタルツインに反映された分析結果を、現実世界の端末から確認することができる [5]。

現在、Beyond5G/6G の研究が活発に進められており、高速で大容量の通信が低遅延で実現されることが予想される。これにより、今後、大容量のデータ通信を行うアプリケーションが急速に普及することで、ネットワークを流れるトラヒック量が急増することが見込まれている。トラヒック量の増大によってネットワークにおける消費電力が大幅に増大することが予想され [6]、ネットワークシステム全体で消費電力が低減することがデジタルツイン活用における大きな課題である [7]。

このようなトラヒック量と消費電力の増大を解決する方法として、エッジコンピューティング技術の研究が進められている。エッジコンピューティングは、データの発生源付近にエッジサーバを配置し、データ処理の一部を割り当てる技術である [8]。処理がクラウドサーバからエッジサーバに移行されることにより、ネットワーク全体のトラヒック量が抑えられ、消費電力の削減が期待される [9]。

エッジサーバを実際に運用する場合、ネットワークの状態やアプリケーションの種類に応じて、処理の分散方法を動的に決定する必要がある。定期的に状況を監視し、変化に応じて動的に分散方法を最適化することで、効率的な分散処理が可能となり、システム消費電力の低減につながる [10]。

我々の研究グループでは、分散型の映像分析システムにおける消費電力の最適化を、組合せ最適化問題として定式化し、遺伝的アルゴリズム (GA: Genetic Algorithm) [11] を用いて最適化する方式を提案している。このとき、映像分析システムは複数の計算資源とネットワークからなる非常に複雑な構造を持つ。そのため、GA を用いた最適化方式を適用することで、規模の増大に対してスケーラブルな方式として実現している。

しかし、GA では、解が収束することによって環境の変化に対応できないケースがある。そこで我々は、遺伝子制御ネットワーク (GRN: Gene Regulatory Networks) と呼ばれる、生物の遺伝子間の相互作用が形成するネットワーク構造に着目する。GRN の特徴の一つとして、進化の過程で特定の表現型を発現しやすい構造を獲得することが挙げられる。また、ネットワークの構造により、Sensitive node と呼ばれる遺伝型の特定の遺伝子が変異することで、複数の表現型が同時に変異することが知られている [12]。環境の変化に応じて表現型が大きく変化することで、変化に対応することが可能であると考えられる。さらに、今まで体験したことのないネットワークの状態やアプリケーション要求に対しても、過去の解の組み合わせから適応しやすい解候補を作ることができれば、環境変動に短時間で適応することが可能であると考えられる。

本研究では、生物の遺伝子制御機構である GRN に基づいた、ネットワーク帯域幅の変化に適応可能な分散型映像分析システムの最適化手法を提案する。GRN の数理モデルである Wagner モデル [13] を用いて、事前にいくつかの環境の最適解を表現型として学習した GRN を作成し、遺伝型および表現型の変異により適応性が向上するか検証する。ネットワークの状態が変化した際に、過去に記憶した解である表現型を GRN が想起することで、GA よりも短い時間で有利な解を発見できるか検証する。また、少ない遺伝型の変異で複数の表現型を一斉に変化させることで、局所解を脱して素早く有利な解を発見できるか検証する。

2. 関連研究

2.1 遺伝子制御ネットワーク

遺伝子制御ネットワーク (GRN) は遺伝子同士の相互作用のネットワークである [14]. 相互作用には活性を促進する作用と抑制する作用があり、遺伝子同士が互いに活性を調節することで、様々な細胞が発現し、生命活動に関わる器官が発生する。遺伝子型に対して GRN により生成された遺伝子の発現パターンを表現型と表記する。

2.1.1 GRN による環境の記憶

生物の進化では、自然選択によって生存上有利な変異のみが残される。このとき GRN は進化過程を経て表現型の分布に制約や偏りが現れる場合がある。このような制約や偏りは自然選択による影響を受けて変化したものであり、過去の表現型を記憶し想起できる分散型連想記憶、部分的な胚から完全な表現型を正確に表現する連合記憶、表現型の特徴の新たな組み合わせを生み出す一般化の三つの特性を持つことが報告されている [15]。また、Boolean Network と呼ばれる GRN の数理モデルでは、遺伝型の特定の遺伝子 (Sensitive node) に対する変異が複数の表現型を同時に変異させ、表現型を大きく変化させることが知られている [12]。本稿で対象とする、環境変動が発生するシステムの組合せ最適化問題では、部分的な環境条件から過去の記憶を想起して転用することで適応力の向上が考えられる。さらに、表現型の一般化が有効に活用されれば、未知の環境条件に対して適切な表現型を生み出すことが期待される。

2.1.2 GRN の数理モデル

本稿では Wagner 氏によるネットワークモデル [13] を用いて、GRN の構造を GA に導入する。遺伝型と表現型は長さ n の配列を、GRN は $n \times n$ の隣接行列を用いて表し、表現型は遺伝型と GRN の乗算により算出される。

2.2 BAM を用いた分散映像分析システムの消費電力最適化方式

我々の研究グループでは、分散型映像分析システムにおける消費電力最適化を、組合せ最適化問題として定式化し、遺伝的アルゴリズムを用いた最適化方式を提案した。映像分析アプリ

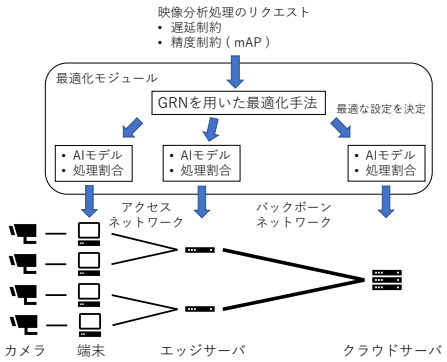


図 1　システムの全体像

ケーションから処理遅延と認識精度が制約条件として与えられ、消費電力が最小となるような分散処理のパラメータを遺伝的アルゴリズムを用いて決定する。人間の脳の認知モデルの一つとして知られる BAM (Bayesian Attractor Model) を用いて、ネットワーク環境や精度制約などの環境情報が過去の環境と類似しているかを判断し、類似した環境の解を初期個体として設定することで、環境へ適応する速度を向上する方法も提案されている [16]。ただしこの手法では、類似した環境については判断可能であるが、環境情報が部分的に類似している場合は類似環境の判断を誤ってしまうことが考えられる。本研究では、類似した表現型を解として生成可能な GRN を用いることで、環境情報を用いて類似性を判断することなく新たな環境に対して自律的に進化可能な仕組みの構築を目指すものである。

3. 提 案 手 法

3.1 システムモデル

システムの全体像を図 1 に示す。システムはカメラ、端末、エッジサーバ、クラウドサーバで構成され、端末とエッジサーバはアクセスネットワークで、エッジサーバとクラウドサーバはバックボーンネットワークで接続されている。アプリケーションが映像処理をリクエストすると、端末、エッジサーバ、クラウドサーバを 1 つずつ繋ぐ経路にセッションが生成される。最適化モジュールは、各ネットワークの通信帯域や遅延の情報を保持しており、映像処理リクエストに応じて適切な AI モデル及び処理割合を各デバイスに設定し、映像処理が開始される。最適化モジュールは、アプリケーションが要求する制約要件を満たすような処理分散方法を発見するまでは映像分析タスクを実施しない。

3.2 最適化問題の定式化

文献 [16] に基づき、上記の分散型の映像分析システムにおける消費電力の最適化を組合せ最適化問題として定式化する。

3.2.1 決 定 変 数

セッション s の映像分散処理に関する制御情報は、各デバイスに割り当てる処理割合 (W_s^t, W_s^e, W_s^c) と映像分析に使用する AI モデル (M_s^t, M_s^e, M_s^c) である。

3.2.2 制 約 条 件

本稿で対象とする最適化問題における制約条件は、処理要求に対する E2E (end-to-end) 遅延と分析精度とする。

処理遅延に関する制約条件は、式 (1) で定義する。

$$T_s(t) = \sum_{d \in D^s} T_s^d(t) + \sum_{n \in N^s} T_s^n(t) \leq T_s^{max}, \forall s \in S. \tag{1}$$

ここで、S は全セッションの集合、D^s, N^s はそれぞれセッション s が使用するデバイス (端末、エッジサーバ、クラウドサーバ) の集合とネットワークの集合、$T_s^d(t), T_s^n(t)$ はセッション s のデバイス d における映像分析にかかる処理遅延とネットワーク n の伝送遅延である。セッション s のリクエストに対する処理遅延 $T_s(t)$ がすべて、アプリケーションが設定する処理遅延の上限値 T_s^{max} を上回らないことが制約条件となる。

分析精度に関する制約条件は、式 (2) で定義する。

$$A_s = \sum_{d \in D^s} A^{M_s^d} W_s^d / |D^s| \geq A_s^{min}, \forall s \in S. \tag{2}$$

A_s はセッション s における映像分析精度であり、アプリケーションが設定する分析精度の下限値 A_s^{min} を上回ることが制約条件となる。

3.2.3 目 的 関 数

最適化問題の目的関数は、時刻 t のシステム全体の消費電力を $E(t)$ として、各デバイス d と各ネットワーク n の消費電力 $E^d(t)$ と $E^n(t)$ の合計値として式 (3) のように定義する。

$$E(t) = \sum_{d \in D^s} E^d(t) + \sum_{n \in N^s} E^n(t). \tag{3}$$

3.3 最適化手法

3.3.1 遺伝的アルゴリズム (GA)

N 個の個体を含む集団を構成し、環境への適合度が高い個体に遺伝的操作を加えながら、次の世代に保存する最適化手法である。各個体は遺伝型を保持しており、環境に有利な適合度が高い個体に遺伝的操作を適用して類似した個体を生成することで、解空間の探索を行う。アルゴリズムにおいては、ランダムな遺伝型を持つ N 個の個体で構成された初期集団を作成し、個体の適合度をもとに次世代に残る個体を N 個選択する。選択された各個体の遺伝型に対して、交叉や突然変異等の遺伝的操作を適用して、新たな遺伝型を持つ個体を生成する。

3.3.2 解の符号化

GA を用いて最適化問題を解くために、分散処理の制御情報を遺伝型に符号化する。セッション s に関する制御情報は各デバイス d における処理割合と AI モデルの選択情報から、$[W_s^d \forall s \in S, \forall d \in D^s, M_s^d \forall s \in S, \forall d \in D^s]$ で表現される。

a) 適合度の定義

個体の適合度 F を以下で定義する。

$$F = -E - \alpha \sum_{s \in S} (T_s^{max} - T_s) - \beta \sum_{s \in S} (A_s - A_s^{min}). \tag{4}$$

α, β は、遅延制約のペナルティ項 (第 2 項) と精度制約のペナルティ項 (第 3 項) が適合度に与える影響の大きさを表す。

3.4 GRN を用いた遺伝的アルゴリズム

集団内の N 個の個体が、それぞれ遺伝型と GRN の隣接行列、表現型の三つを保持する。

3.4.1 表現型の算出方法

表現型は、遺伝型と GRN の隣接行列の乗算により得られる配列である。遺伝型を $G = [g_0, g_1, \ldots, g_n](-1 \leq g_k \leq 1)$、GRN の隣接行列を $B = [b_{ij}](0 \leq i, j \leq n)$ として、表現型の発現過程を式 (5) で表す。

$$\begin{cases} p_i(0) & = g_i, \\ p_i(k+1) & = p_i(k) + \tau_1 \sigma \left(\sum_{j=0}^{n} b_{ij} p_j(k) \right) - \tau_2 p_i(k). \end{cases} \tag{5}$$

k ステップの発現過程によって得られた表現型は $P(k) = [p_0(k), p_1(k), \ldots, p_n(k)]$ で表され、個体の適合度はパラメータ k^* による $\sigma(p_i(k^*))$ を用いて算出する。また τ_1 は遺伝子間の相互作用が発現パターンに与える影響の大きさ、τ_2 は発現パターンの減衰率、σ はシグモイドであり、tanh 関数を用いる。

3.4.2 アルゴリズムの流れ

提案手法はランダムな遺伝型を持つ N 個の個体で構成された初期集団に対して、以下の手順を繰り返すことで進化を進め、解を探索する。

（1） 個体の適合度を算出し、上位 N_1 個をエリート個体として保存

（2） $N - N_1$ 個の個体から子集団に含める個体を N_2 個生成
- （a） 選択方式に従って個体を一つ選択
- （b） 選択した個体の表現型に突然変異と交叉を適用して新たな個体を生成
- （c） 個体が N_2 個生成されるまで (a)(b) を繰り返す

（3） さらに $N - N_1$ 個の個体から子集団に含める個体を $N - N_1 - N_2$ 個生成
- （a） 選択方式に従って個体を一つ選択
- （b） 選択した個体の遺伝型に突然変異と交叉を適用して新たな個体を生成
- （c） 個体が $N - N_1 - N_2$ 個生成されるまで (a)(b) を繰り返す

3.4.3 GRN の事前学習

GRN による表現型の学習では、文献 [15] の方法を利用する。学習させる表現型と、現在発現している表現型の差が小さくなるように GRN の隣接行列の要素を変異させる。具体的には、GRN に記憶させたい複数の表現型から、i 番目の表現型 P_i をランダムに選択しターゲット表現型とする。GRN によって生成される表現型 P^* の適合度を、$1 + P^* \cdot P_i$ で計算し、GRN の隣接行列に突然変異を適用させる。一定世代ごとにターゲット表現型を切り替えることで、表現型の発現に偏りが生まれ、GRN が特定の表現型を記憶した状態を構築する。

4. 評価結果

ネットワーク帯域幅を変化させた場合に、GRN によって適応性が向上すること、GRN の特性が生かされていることを確認する。帯域幅の変動によって、変動前と同一の分散方法では、制約を満たせない状態、もしくは、消費電力を過剰に消費している状態へと変化する。アプリケーションがシステムを使用している際に帯域幅の変動を発生させた場合に、変動直後からサービスを提供できる時間の長さと消費電力を評価する。それぞれ GRN を使用する提案手法と、使用しない場合における結果を比較する。

4.1 シミュレーション設定

シミュレーションのため、端末、エッジサーバ、クラウドサーバの三種類のデバイス ($D^s = \{t, e, c\}$) と、アクセスネットワークとバックボーンネットワークの二種類 ($N^s = \{n_{access}, n_{backborn}\}$) を設定する。計算資源の数とネットワーク構成は図 1 の通りで、各端末が 1 セッション、つまり合計 4 つのセッションが構築されるものとする。

表 1　各デバイスの性能と推定された性能モデルに関するパラメータ

デバイス, variables	端末 t	エッジサーバ e	クラウド c
CPU	Core i7-8700T	Xeon GOLD 6226R ×2	Core i9-10940X
IDLE 電力: E^d_{IDLE}	55.236W	334W	258.3176W
CPU TDP:$E^d_{CPU\text{-}TDP}$	35W	150W ×2	165W
GPU (Nvidia)	GeForce GTX1070	Tesla T4 ×2	RTX A5000
GPU FP32: C^d	6.463Tflops	8.141Tflops×2	27.77Tflops
GPU TDP:$E^d_{GPU\text{-}TDP}$	150W	70W ×2	230W
GPU 効率: Eff^d	0.4	0.39	0.48
FLOPS A,B:(O^A,O^B)	(1.5, 0.1)	(2.2, 0.1)	(8.1, 0.4)
(α^E, β^E)	(0.97, 0.78)	(0.97, 0.67)	(0.29, 0.81)

4.1.1 デバイスに関する設定

シミュレーションに使用するデバイスや各デバイスの処理遅延と消費電力モデルを、文献 [16] に基づき、実測値から推定されたモデルを用いて表 1 のように設定する。

まず、時刻 t のデバイス d における GPU 処理負荷率 $L^d(t)$ は式 (6) で推定される。

$$L^d(t) = \sum\nolimits_{s \in S^d} O^d_s(t) / C^d Eff^d. \tag{6}$$

また、$O^d_s(t)$ をデバイス d においてセッション s にかかる処理負荷として、式 (7) で算出する。

$$O^d_s(t) = \left((O^{M^d_s(t)} + O^A) W^d_s(t) + O^B \right) \text{(FPS)}. \tag{7}$$

式 (6), (7) を用いて、デバイスにおける処理遅延と消費電力を算出する。処理遅延について、$T^d_s(t)$ は式 (8) で算出する。

$$T^d_s(t) = \left(O^{M^d_s(t)} + O^A + O^B \right) / C^d Eff^d L^d(t). \tag{8}$$

またデバイス d における消費電力 $E^d(t)$ は式 (9) で算出する。

$$E^d(t) = E^d_{IDLE} + \alpha^E E^d_{CPU\text{-}TDP} + \beta^E E^d_{GPU\text{-}TDP} L^d(t), \tag{9}$$

4.1.2 ネットワークに関する設定

帯域幅を変動させる際のネットワーク環境について、表 2 のように設定する。帯域幅の変動は 180 秒ごとに 49 回発生させ、最初の環境と合わせて 50 通りの環境で実施する。同一の環境設定でシミュレーションを 10 回実施する。

ネットワークの伝送遅延と消費電力は、それぞれ式 (10) と (11) で算出する [16]。

$$T^n_s(t) = R \left(1 - \sum\nolimits_{d \in D^{s,n}_{pass}} W^d_s(t) \right) / B^n(t) + T^n_{prop}, \tag{10}$$

$$E^n(t) = \sum\nolimits_{s \in S} R \left(1 - \sum\nolimits_{d \in D^{s,n}_{pass}} W^d_s(t) \right) E_{bit}. \tag{11}$$

ここで $D^{s,n}_{pass}$ はセッション s がネットワーク n に到達するまでに通過したデバイスの集合であり、$B^n(t)$ は時刻 t においてネットワーク n で利用可能な帯域幅で、$B^n(t) \leq B^n_{max}$ を満たす。

4.1.3 アプリケーションの性能要求の事前設定

事前に三種類の性能要求 A, B, C を設定し、それらを満たしつつ消費電力を低減させる解を GRN に表現型として学習させた (表 3)。学習させる表現型の選択では、全ての解の適合度の評価に膨大な時間を要するため、SGA による解探索を複数回実施し、最も適合度が高かった解を選択した。その際、n_{access} と

表 2　シミュレーション時のネットワーク環境設定

ネットワーク	n_{access}	$n_{backborn}$
帯域幅: B^n_{max}	10 – 50 Mbps	10–500 Mbps
伝搬遅延: T^n_{prop}	10 ms	10 ms
1 ビットあたりの消費電力: E_{bit}	193×10^{-9} W	60×10^{-9} W
1 フレームあたりのビット数: R	470 Kbit	

表 3　事前に設定した性能要求の値と GRN に学習させた表現型

各セッションの性能要求	遅延制約: T^{max}_s	精度制約: A^{min}_s (mAP)	記憶させた表現型	適合度: F
A1	0.04238 s	26.855 %	[0,0,6,4,6,2,	-365.696
A2	0.12856 s	20.926 %	0,0,3,2,2,4,	
A3	0.11831 s	27.375 %	0,0,6,7,6,6,	
A4	0.10322 s	22.440 %	0,0,5,1,3,7]	
B1	0.06481 s	34.535 %	[0,0,5,6,7,5,	-389.289
B2	0.05974 s	41.178 %	0,0,6,7,0,5,	
B3	0.04144 s	26.532 %	6,0,0,0,7,3,	
B4	0.09141 s	37.337 %	0,0,5,1,2,4]	
C1	0.12314 s	48.583 %	[0,0,7,0,0,4,	-379.944
C2	0.03937 s	31.529 %	7,0,0,0,0,2,	
C3	0.12405 s	33.166 %	0,0,5,3,7,4,	
C4	0.11723 s	23.430 %	0,0,5,1,3,0]	

表 4　文献 [17] に基づく映像分析に使用する AI モデルの性能値

Model	Accuracy: $A^{M^d_s}$	FLOPS: $O^{M^d_s(t)}$	M^d_s
YOLOv3-tiny	33.1 %	5.6 B	$\{0,1,2\}$
YOLOv3	55.3 %	65.9 B	$\{3,4,5\}$
YOLOv3-spp	60.6 %	141.5 B	$\{6,7\}$

$n_{backborn}$ の帯域幅はそれぞれ 25 Mbps と 250 Mbps とした。

4.1.4　評価指標

環境変動直後に、制約条件を満たす解が発見されずにサービスが提供できない時間の長さと、サービスが提供できる時間における合計の消費電力を比較する。さらに、制約条件と消費電力を用いて算出される適合度に関して、一定時間経過時の適合度の高さを用いて、環境変動に対する適応速度を評価する。

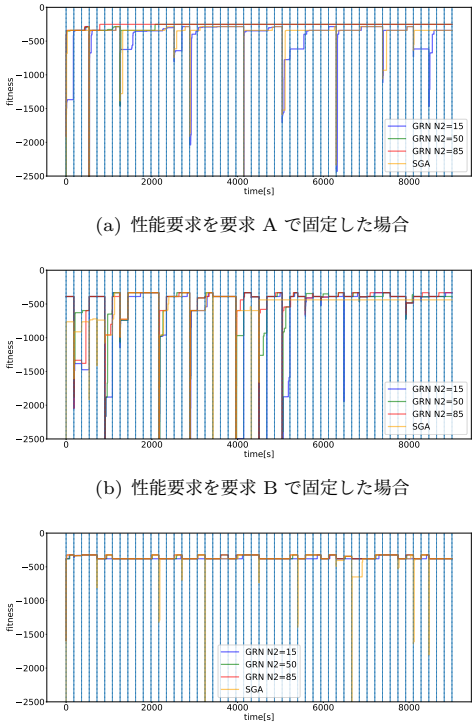
4.1.5　パラメータ設定

提案手法において生成される個体について、エリート個体と遺伝型の変異で生成される個体、表現型の変異で生成される個体の数が 5 : 45 : 50 となるよう、$N = 100$、$N_1 = 5$、$N_2 = 50$ と設定する。また、ルーレット選択とエリート生存戦略 (エリート生存割合は 0.05)、2 点交叉 (交叉率 0.2)、点突然変異を用いる。デバイスの処理割合 W^d_s と AI モデルの選択情報 M^d_s は 0–7 の範囲とする。AI モデルは表 4 の中から選択する。また、$(\alpha,\beta) = (10^5, 10^5)$、$(\tau_1,\tau_2) = (1.0, 0.2)$、$t^* = 10$ と設定する。

4.2　評価結果

性能要求を要求 A, B, C にそれぞれ設定した場合の解の適合度の時間推移を図 2 に示した。

要求 A では、提案手法の方が短時間で環境に適応し、適合度が高いケースが多く見られた。図 2(a) において、サービスが提供できなかった時間の合計が、提案手法では 149 秒、SGA では 382 秒、サービスを提供できた時間における平均の消費電力は提案手法が 288 W、SGA が 366 W と、GRN によって環境適応時間が短縮され、消費電力を削減する適合度の高い解が発見されやすくなることが分かった。尚、計 500 回の環境変動において、10 秒後に制約条件を満たした解が発見できた回数は、提案手法が 489 回で SGA が 453 回、10 秒後の適合度が提案手法の方が高かった回数は 334 回で SGA が 39 回、同じケースが 163 回と、提案手法の方が適応性が高かった。特に、図 2(a) で提案手法が 12 回目の環境変動において、多くの帯域幅で適合度が高い解を発見しており、10 回の試行のうち、8 回で同様のケースが確認された。一方で、SGA ではこのような解は発見されなかった。これは、GRN では 1 ビットの遺伝型の変異で複数の表現型が変異することから、SGA よりも短時間で、要求 A の解の周辺を広く探索できることが原因の一つである。また、GRN のネットワーク構造により、GRN が記憶した複数の表現型から新たな表現型を生成する特徴が理由の一つに挙げられる。具体的には、図 3 で示した表現型は、要求 A の解と C の解を組み合わせたような構造をしており、要求 A において高い適合度を示している。このような新たな解やその周辺を探索することで、新たな環境に適応できたと考えられる。



(a) 性能要求を要求 A で固定した場合

(b) 性能要求を要求 B で固定した場合

(c) 性能要求を要求 C で固定した場合

図 2　提案手法 (GRN N2=50) および SGA の適合度の推移。縦点線は環境変動が発生したタイミング。

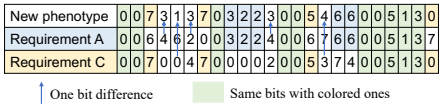


図 3　GRN が記憶を組み合わせて生成された新たな表現型

要求 B では、提案手法の方が適合度が高いケースが多かったが、制約条件を満たす解が見つからずにサービスが提供できないケースも見られた。図 2(b) において、消費電力はサービスを提供できた時間の平均電力が、提案手法が 389W、SGA が 455W と提案手法の方が低かったが、サービスを提供できない時間が、GRN は 1,294 秒で SGA が 296 秒と提案手法の方が長く、GRN による改善があまり見られなかった。尚、10 秒後の適合度が GRN の方が高かったケースが 371 回、SGA が 89 回、同じが 40 回である。また、180 秒後であれば GRN が 363 回で、SGA が 65 回、同じが 72 回である。以上により、GRN を用いる方が短時間で高い適合度を示す解を発見できることが

確認できる。これは特に、図 2(b) で SGA が 26 回目の環境変動の際に局所解に陥ったことが原因である。10 回の試行において、9 回が同様の傾向を示している。一方で、提案手法では局所解に陥ることなく、より高い適合度の解が発見されている。GRN は 1 ビットの遺伝型の変異で複数の表現型が変異し、局所解から脱しやすいことが、環境変動への適応性に良い影響を与えたケースであると考えられる。しかし、シミュレーション全体においては、提案手法が制約条件を満たす解の発見に時間がかかるケースも見られる。計 500 環境に対して、10 秒後に制約条件を満たした回数は GRN が 408 回で SGA が 467 回、180 秒後であれば GRN が 446 回で、SGA が 492 回である。一つの原因は、SGA が発見した局所解が、多くの帯域幅において常に制約条件を満たす解であったことである。また提案手法で適合度が上がりにくい原因として、GRN が記憶した解が帯域変動時に適合度が下がりやすい解であったことが考えられ、GRN に記憶させる解として、異なる帯域幅でも適合度が高い解を選択することで、改善される可能性がある。

要求 C は、提案手法の方が SGA よりも適合度が高い結果となった。図 2(c) においては、サービスを提供できなかった時間は、提案手法は 131 秒、SGA は 451 秒と、提案手法がサービスを安定的に提供できでいる。また消費電力の平均は、提案手法が 357 W、SGA が 371 W と、低減されていることが分かった。また 10 回のシミュレーションで、10 秒経過時に制約条件を満たした回数を比較すると、提案手法は 500 回中 500 回であるが、SGA では 461 回であり、提案手法が優位であることを確認している。帯域幅が変化した場合でも適合度が大きく変化しなかったことから、記憶させた解が帯域幅を変化させても適合度が高い解であったことが考えられる。GRN は安定的に記憶を想起したが、SGA では一時的に局所解に陥ったため、適応性に差がうまれている。また SGA も直前の環境の解をそのまま利用することで高い適合度となることが多く、全体として差が開かなかったと考えられる。

全体として、GRN を用いた提案手法と SGA は、同じ環境においては提案手法の方が高い適合度となった。GRN に記憶させる解によっては、環境変動全体での消費電力の改善が見られないケースもあるため、記憶させる解をどのように選択するかは今後の課題の一つである。

5. おわりに

映像分析の分散処理において、GRN の表現型の偏りや 1 ビットの遺伝型の変異による複数の表現型の変異、既知の表現型から新たな表現型を生成する特性を利用することで、GA の適応性を向上させる最適化手法を提案した。ネットワークの帯域幅が変化する環境に対して、提案手法では GRN の特性が有効に働き、SGA よりも適応性が向上することを確認した。

今後、より多様な環境変動やセッション数の多い複雑な問題についても評価を行い、GRN による GA の進化適応性の向上について確認する予定である。また、動的にパラメータを調整することで、より適応性が向上することも考えられるため、検証予定である。さらに、他の最適化手法との比較も実施する。

謝　辞

本研究開発は総務省の「ICT 重点技術の研究開発プロジェクト（JPMI00316)」によって実施した成果を含みます。

文　献

[1] J. Yang, F. Lin, C. Chakraborty, K. Yu, Z. Guo, A.-T. Nguyen, and J.J.P.C. Rodrigues, “A parallel intelligence-driven resource scheduling scheme for digital twins-based intelligent vehicular systems,” IEEE Trans. Intell. Vehicles, vol.8, no.4, pp.2770–2785, April 2023.

[2] R. Kaul, C. Ossai, A.R.M. Forkan, P.P. Jayaraman, J. Zelcer, S. Vaughan, and N. Wickramasinghe, “The role of AI for developing digital twins in healthcare: The case of cancer care,” WIREs Data Mining and Knowl. Discovery, vol.13, no.1, p.e1480, Nov. 2022.

[3] M. Singh, E. Fuenmayor, E.P. Hinchy, Y. Qiao, N. Murray, and D. Devine, “Digital twin: Origin to future,” Appl. Syst. Innov., vol.4, no.2, p.36, May 2021.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” Proc. of IEEE Conf. Comput. Vis. and Pattern Recog. (CVPR), pp.779–788, June 2016.

[5] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” Future Gener. Comput. Syst., vol.29, no.7, pp.1645–1660, Sept. 2013.

[6] V.K. Quy, A. Chehri, N.M. Quy, N.D. Han, and N.T. Ban, “Innovative trends in the 6G era: A comprehensive survey of architecture, applications, technologies, and challenges,” IEEE Access, vol.11, pp.39824–39844, April 2023.

[7] D.M. Botín-Sanabria, A.-S. Mihaita, R.E. Peimbert-García, M.A. Ramírez-Moreno, R.A. Ramírez-Mendoza, and J.d.J. Lozoya-Santos, “Digital twin technology challenges and applications: A comprehensive review,” Remote Sens., vol.14, no.6, pp.1–25, March 2022.

[8] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: Vision and challenges,” IEEE Internet of Things J., vol.3, no.5, pp.637–646, Oct. 2016.

[9] C. Caiazza, S. Giordano, V. Luconi, and A. Vecchio, “Edge computing vs centralized cloud: Impact of communication latency on the energy consumption of LTE terminal nodes,” Comput. Commun., vol.194, pp.213–225, Oct. 2022.

[10] Y. Chen, X. Shen, P. Zhang, S. Lu, L. Wang, Z. Wu, and X. Xie, “Joint optimization of uav-wpt and mixed task offloading strategies with shared mode in sag-piot: A mad4pg approach,” Internet of Things, vol.24, p.100970, Dec. 2023.

[11] D. Whitley, “A genetic algorithm tutorial,” Statistics and Comput., vol.4, pp.65–85, Oct. 1998.

[12] P.A. Dnyane, S.S. Puntambekar, and C.J. Gadgil, “Method for identification of sensitive nodes in boolean models of biological networks,” IET Syst. Biol., vol.12, no.1, pp.1–6, Feb. 2018.

[13] A. Wagner, “Does evolutionary plasticity evolve?,” Evolution, vol.50, no.3, pp.1008–1023, June 1996.

[14] E.H. Davidson and D.H. Erwin, “Gene regulatory networks and the evolution of animal body plans,” Sci., vol.311, no.5762, pp.796–800, Feb. 2006.

[15] R.A. Watson, G.P. Wagner, M. Pavlicev, D.M. Weinreich, and R. Mills, “The evolution of phenotypic correlations and “developmental memory”,” Evolution, vol.68, no.4, pp.1124–1138, Dec. 2014.

[16] H. Shimonishi, M. Murata, G. Hasegawa, and N. Techasarntikul, “Energy optimization of distributed video processing system using genetic algorithm with bayesian attractor model,” Proc. of IEEE 9th Int. Conf. Netw. Softwarization (NetSoft), pp.35–43, June 2023.

[17] Online:https://github.com/AlexeyAB/darknet. Accessed 8 August 2023.