特別研究報告

題目

モバイルネットワークのスライス埋め込み問題を対象とした量子遺伝子制御ネットワークアルゴリズムの適用手法

指導教員
村田 正幸 教授

報告者
関澤 和希

2025 年 2 月 6 日

大阪大学 基礎工学部 情報科学科

令和 6 年度 特別研究報告

モバイルネットワークのスライス埋め込み問題を対象とした
量子遺伝子制御ネットワークアルゴリズムの適用手法

関澤 和希

内容梗概

環境変動が想定される通信ネットワークの制御問題に対して、我々の研究グループでは、遺伝子制御ネットワーク（GRN: Gene Regulatory Networks）の数理モデルを用いた遺伝的アルゴリズム（GA: Genetic Algorithm）による動的な制御方法を研究してきた。GRN は、生物の遺伝子を表す遺伝型から細胞の発現を表す表現型へのマッピングを担っている。GRN を介することで、遺伝型に対して、表現型が一定の偏りを示す。生物の生存に適した細胞、つまり表現型を発現しやすいように GRN の制御情報が変化することで、特定の表現型を記憶する役割を担うことができる。また、GRN の構造の一部が変化することで表現型が大きく変化する機能や、特定の表現型を発現しやすいよう GRN の構造自体を変化させて表現型の特徴の新たな組み合わせを生み出す一般化といった機能も保持している。GA に GRN の数理モデルを導入し、これらの特性を活用することによって、GA の欠点である局所解への収束を回避するなど、動的環境における効率的なネットワーク制御の最適化を達成している。ただし、対象とする問題の規模が大きくなるほど、それに伴って GRN の規模が拡大し、計算量の増大が課題の一つであった。このような問題に対して、量子ビットの並列的な状態遷移を通じて高速演算が可能な量子コンピュータに適用できる GRN のモデルである、量子遺伝子制御ネットワーク (QGRN: Quantum Gene Regulatory Networks) に着目している。QGRN は、遺伝子間の相互作用を、ゲート型量子コンピュータにおける量子ゲートの演算を用いて表現することで、遺伝型から表現型にマッピングする、GRN の数理モデルの一つである。そのため、内部の演算方法を変更することなく、量子コンピュータに適用可能であることが利点の一つである。今後、量子コンピュータが発展し、並列演算可能なビット数の増大や、高速演算の実現により、量子コンピュータを用いることで、QGRN モデルの演算速度が向上し、課題である GRN の計算量増大にも対応できると期待される。また QGRN モデルでは量子ビットを用いており、複数の表現型を一定の確率で発現しているが、記憶させたい表現型を確率分布として扱うことができ、複数の表現型を記憶するケースにおいて性能の向上が期待される。

本報告では、モバイルネットワークのスライス埋め込み問題を対象にして、量子遺伝子制御ネットワークを用いた最適化手法を検討する。IoT をはじめとして様々な機器がネットワークに接続する昨今、通信容量、信頼性、遅延、端末数など、ネットワーク通信の要件が複雑多様化しているが、このような多種多様な要求を受容する一つの方法として、ネットワークスライシングが挙げられる。ネットワークスライシングは、1 つの物理ネットワークを仮想的にスライスに分割し、各スライスに端末およびコンピューティングリソースを割り当て、各通信要求を達成する技術である。ネットワークスライシングでは、時々刻々と変化する通信環境やスライス要件に合わせて、経路選択やコンピューティングリソースの割り当てなど、適切なスライスの埋め込み方法を動的に選択する必要がある。これらのモバイルネットワークのスライス埋め込み問題は最適化問題として設定可能である。

本報告ではまず QGRN の性能評価として、表現型の記憶能力の確認として、5 個の表現型の記憶テストを行った。QGRN は確率分布で複数の表現型を同時に記憶可能な性質によって、GRN よりも多くの表現型が記憶可能となっていることを確認した。また、モバイルネットワークのスライス埋め込み問題に量子遺伝子制御ネットワークを用いた遺伝的アルゴリズムを適用し、QGRN の性能および性質について確認した。GRN を用いない GA と比較したところ、QGRN の構造変化によって複数の表現型が同時に変化し局所解を回避する特性を活かすことで、局所解に陥りやすい環境設定であっても局所解に陥ることなく、最適解に到達可能であることを確認した。

主な用語

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
量子コンピュータ
量子遺伝子制御ネットワーク
ネットワークスライシング

目 次

図 目 次

表 目 次

1　はじめに

IoT や自動運転、スマートグリッドなど、様々な機器がネットワークに接続する昨今、ネットワークの通信状況は時々刻々と変化しているが、このような動的に環境が変動する通信ネットワークの制御問題に対して、我々の研究グループでは、遺伝子制御ネットワーク（GRN: Gene Regulatory Network）の数理モデルを用いた遺伝的アルゴリズム（GA: Genetic Algorithm）による、通信ネットワークの動的な制御方法を研究してきた。GRN は、生物の遺伝子間の相互作用をモデル化したネットワーク構造であり、生物の遺伝子を表す遺伝型から細胞の発現を表す表現型へのマッピングを担っている [1]。生物は進化の過程において、環境変化に合わせて生存に優位な細胞を生成する必要があり、特定の表現型を高い確率で発現するような構造へと GRN を変化させる。このように GRN の構造を変化させて特定の表現型を高い確率で発現する、つまり特定の表現型を記憶することができる GRN を持つ個体が、種として生存しやすい傾向にある。また GRN はその特徴的な構造から、一部の遺伝型が変化しても同じ表現型を保持する特徴や、表現型の特徴を構造的に学習し新たな表現型を生成する一般化といった特徴を持つことが知られている [2]。また、sensitive node と呼ばれる特定の遺伝型が変異することにより、表現型の一部が一斉に変化するような機構を持つことも知られており [3]、GRN を介した GA では特定の遺伝型の変化により、表現型が大きく変化することで、GA の問題点の一つである局所解に陥る点を解消している [4]。我々の研究グループでは、Wagner モデル [5] と呼ばれる GRN の数理モデルを用いて、ネットワーク状態が周期的に変動する環境における、映像分散処理システムのリソース割り当て制御の問題に対して、最適解を事前に記憶させた GRN を用いた場合、記憶させた環境と類似した環境に変化しても、短時間で新たな環境に適応可能であることを示した [4]。一方で、対象とする問題の規模が大きくなるにつれて、GRN の規模も拡大し、計算量が増大することが課題として挙げられる。

このような問題に対して、近年開発が進む量子コンピュータに適用可能な GRN のモデルである、量子遺伝子制御ネットワーク (QGRN: Quantum Gene Regulatory Network) モデルに着目している。QGRN は、遺伝子間の相互作用を、ゲート型量子コンピュータにおける量子ゲートの演算で表現し、遺伝型から表現型にマッピングするモデルである。そのため、QGRN モデルは相互作用の演算方式などを変更することなく量子コンピュータに適用可能である [6]。QGRN の演算を量子コンピュータを用いて実行することで、複数通りの状態を並列に演算し、問題規模の拡大に伴い GRN が大規模化した場合においても、演算時間の増大を抑えられると見込まれる。量子コンピュータも今後さらなる発展が想定されており、第 6 世代移動通信システム（6G）が実現される 2029 年頃には 100 万量子ビットの量子コンピュータの実現がマイルストーンとされている [7]。また、量子ゲートの高速化も進ん

でおり、IBM 量子コンピュータでは、2022 年から 2024 年にかけて、1 秒当たりの量子ゲート処理数が 950 CLOPS から 150,000 CLOPS に増加している [8]。今後、大規模かつ高速な量子コンピュータを用いることで、大規模な問題に対しても QGRN モデルが適用可能になると考えられる。さらに、既存の GRN の特性に加えて、QGRN の特性も、動的なネットワーク制御における新たな適応性に繋がると考えられる。QGRN モデルを用いてリンパ芽球の遺伝子間の相互作用を推論し、既存の GRN 推論モデルでは表現できていなかった遺伝子間相互作用が表現可能であることから [6]、既存の GRN に比べて表現能力や記憶能力の向上が期待される。また、QGRN は量子演算の特性上、複数の表現型が確率的に発現するが、記憶したい表現型も確率分布として扱うことができるため、表現型の記憶能力の向上も期待される。

本報告では、通信ネットワークの動的な制御に関する問題として、ネットワークスライスの埋め込み問題を対象とする。近年、IoT など膨大な数と種類のデバイスが同時にネットワークに接続し、通信容量、信頼性、遅延制約といった通信要件が複雑化・多様化している。また、各デバイスの通信需要や接続状況が刻々と変動する中、これらの要件をリアルタイムで把握し、ネットワークを制御する必要がある。ネットワークスライシングは、1 つの物理ネットワークを仮想的に複数に分割 (スライス) し、各スライス上の端末等の通信要求に応じて、通信経路やコンピューティングリソースを物理ネットワークに割り当てることで、要件に応じた柔軟なネットワークサービスの提供を可能とする技術である。そのため、ネットワークスライシング技術は、6G においても不可欠な要素として挙げられている [9]。ネットワークスライシングでは、通信環境やコンピューティングリソースの使用状況、スライス上の端末の通信要求といった要件を満たすように、ネットワーク経路やデータ処理を行うサーバを決定する、つまりスライスの適切な埋め込み方法を選択する必要がある。このような埋め込み方法は、様々な方法で最適化問題として定式化されている [10]。このような最適化問題では、通信帯域や遅延、アプリケーションの要件やコンピューティングリソースの状況など、様々な状況を考慮する必要があり、非常に複雑な問題構造となっている。そのため、ヒューリスティックな方法で解くことが多く見られるが、従来の GA などの方法では局所解に陥り、最適化が困難となるケースが存在する。

本報告では、量子遺伝子制御ネットワークの性能評価を行う。特に、表現型の記憶能力および、量子遺伝子制御ネットワークの数理モデルを用いた遺伝的アルゴリズムのネットワーク制御問題への適用性を評価する。本報告の構成は以下に示す。第 2 章では QGRN を用いた最適化問題の解法を提案し、QGRN の記憶性能の評価を行う。第 3 章では、その評価として、モバイルネットワークスライス埋め込み問題の定式化および適応結果について述べる。第 4 章でまとめと今後の課題について述べる。

2　量子遺伝子制御ネットワークを用いた最適化アルゴリズムと事前評価

2.1　遺伝子制御ネットワークとその量子拡張モデル

2.1.1　遺伝子制御ネットワーク

遺伝子制御ネットワークは、生物の遺伝子間の相互作用によって形成されるネットワーク構造である。生物は遺伝子の情報をそのまま細胞の情報に反映するのではなく、遺伝子間で相互に制御を行い、遺伝子の発現パターンを制御する役割を持つ。各遺伝子は、他の遺伝子の発現を促進または抑制することで、細胞内の遺伝情報の制御を行っている。GRN は遺伝子間の制御関係を表現したネットワークモデルであり、遺伝型（生物の遺伝情報）から表現型（細胞の発現状態）へのマッピングを担っている [1]。生物は長い歴史の中で常に環境に適応可能なように GRN の構造を変化させながら生存している。GRN は、環境に適応するための自然淘汰を通じて表現型の変化に関与しており、過去の選択圧の影響を受けながら発達し、表現型の特徴を「記憶」することが知られている。そして、遺伝子の一部が突然変異によって変化した場合でも、完全な表現型を想起し、種の生存を可能としている。記憶された特徴は、生物の発生過程における特定の遺伝子発現パターンが世代を超えて保存および継承され、将来の進化過程で再び現れる可能性がある。さらに、発達した GRN は、進化した表現型は環境に対して一般化され、新たな環境下でも高い適応性を示す [2]。また、sensitive node と呼ばれる特定の遺伝子がわずかに変化した場合でも、表現型が大きく変化するといった特性も知られている [3]。

生物の表現型と遺伝型のマッピングを推定するためのモデルとして、複数の GRN の数理モデルが提案されている。代表的なものとしては、Wagner モデル [5] が挙げられる。Wagner モデルでは、遺伝型と表現型は長さ n の配列として、GRN は $n \times n$ の隣接行列として表現され、表現型は遺伝型と GRN との複数回の乗算により求められる。

我々の研究グループでは、GRN の数理モデルを用いて、表現型の偏りによる記憶特性や、特定の遺伝子の変化による大規模な表現型の変異、過去の環境の表現型の想起といった GRN の統制を生かし、動的に変動するネットワーク制御の最適化問題に応用している。Wagner モデルの GRN に対して、事前に典型的なネットワーク制御パラメータを表現型として記憶させ、動的環境において遺伝型と表現型を変動させることで、典型的な解の想起や一般化により、GRN を用いない GA と比べて、適応速度が向上したことを示している [4]。ただし、GRN の規模が拡大した場合に演算時間が増大することが課題の一つであった。

2.1.2 量子遺伝子制御ネットワーク

量子遺伝子制御ネットワークは、GRN における遺伝型同士の相互作用を量子演算子を用いて表現したモデルである [6]。QGRN モデルでは、n qubit の量子ビットを用いて、各ビット間の相互作用を量子ゲートを用いて演算する。最終状態の各量子ビットを複数回観測すると、一定の確率で複数の表現型が発現するため、表現型は確率分布として得られることが特徴の一つである。文献 [6] において Roman 氏らは、リンパ芽球の RNA データに対して QGRN モデルを用いた遺伝子制御ネットワークの推論を行ったところ、既存の GRN 推論モデルでは表現されていないが、存在すると予測されていた遺伝子間の相互作用を表現したことを示しており、既存の GRN とは異なる表現能力を保持している可能性がある。そのため、既存の GRN と比較して記憶能力の向上が期待される。QGRN の演算には、ゲート型量子コンピュータ上で利用可能な量子ゲートを用いている。そのため、ゲート型の量子コンピュータであれば、演算式の変形等を行うことなく、そのまま適用可能なモデルとなっていることが利点の一つである。

具体的な数理モデルとしては、図 1 のようになっている。遺伝型に相当する L_{enc} 層と、GRN に相当する、k ビット目から他の量子ビットへの相互作用を行う L_k 層で構成されている。

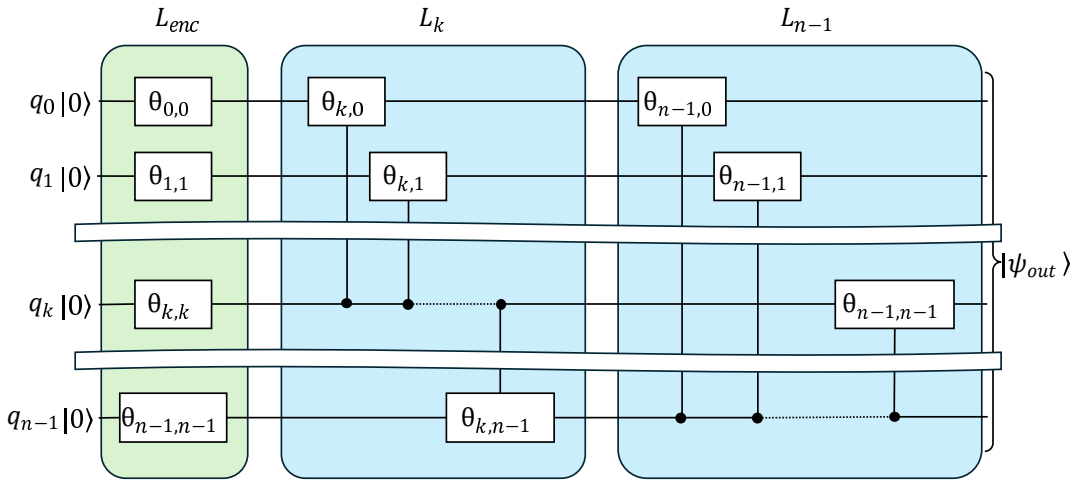


図 1: QGRN の数理モデル

まず L_{enc} 層において、量子ビットの初期状態 $|0\rangle$ に対して、式 (1) の $R_y(\theta)$ ゲートを用いて量子ビットの初期状態を変化させる。n 量子ビットに対しては式 (2) のような演算式とし

て表される。

$$R_y(\theta) = \begin{pmatrix} \frac{\cos\theta}{2} & -\frac{\sin\theta}{2} \\ \frac{\sin\theta}{2} & \frac{\cos\theta}{2} \end{pmatrix}, \tag{1}$$

$$L_{enc} = R_y(\theta_{n-1,n-1}) \otimes \cdots \otimes R_y(\theta_{1,1}) \otimes R_y(\theta_{0,0}). \tag{2}$$

次に L_k 層において、c-$R_y(\theta)$ ゲートを用いて遺伝子間の相互作用の演算を行う。c-R_y ゲートは、表 1 に示すように、制御元となる量子ビットが $|0\rangle$ であれば制御は行われないが、そうでない場合は制御先の量子ビットに対して制御演算を行うものである。また、n 量子ビットに対しては式 (3) のように表される。

表 1: c-$R_y(\theta)$ ゲート

基底状態 $\|x\rangle$	c-$R_y(\theta)\|x\rangle$
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|01\rangle$
$\|10\rangle$	$\cos\frac{\theta}{2}\|10\rangle + \sin\frac{\theta}{2}\|11\rangle$
$\|11\rangle$	$-\sin\frac{\theta}{2}\|10\rangle + \cos\frac{\theta}{2}\|11\rangle$

$$L_k = \prod_{i=0, i\neq k}^{n-1} \text{c-}R_{y,n}(\theta_{k,i}). \tag{3}$$

そのため、量子ビットの最終状態 $|\psi_{out}\rangle$ は、式 (4) で表される。

$$|\psi_{out}\rangle = \left(\prod_{k=0}^{n-1} L_k \right) L_{enc} \, |0\rangle_n \, . \tag{4}$$

このとき、式 (5) の $\boldsymbol{\theta}$ は隣接行列および遺伝型を表すパラメータとなっている。$\theta_{k,k}$ は k 量子ビット目の遺伝型を表すパラメータであり R_y ゲートの引数となっており、$\theta_{k,p}$ は、k 番目の量子ビットから p 番目の量子ビットに対する相互作用に関するパラメータであり c-R_y ゲートの引数となっている。

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_{0,0} & \theta_{0,1} & \cdots & \theta_{0,n-1} \\ \theta_{1,0} & \theta_{1,1} & \cdots & \theta_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{n-1,0} & \theta_{n-1,1} & \cdots & \theta_{n-1,n-1} \end{bmatrix}. \tag{5}$$

2.2 量子遺伝子制御ネットワークを用いた遺伝的アルゴリズム

遺伝的アルゴリズムを用いた最適化手法に QGRN を導入する。GA はある個体集団に対して選択や交叉、突然変異などの遺伝的操作を繰り返し適用し、適合度の高い個体を次世代に保存することで最適解への到達を促す手法である [11]。具体的には、QGRN を有する個体の集団を構築し、QGRN のパラメータ $\boldsymbol{\theta}$ に対して GA の突然変異および交叉といった進化計算を実施する。手順および全体像を、以下および図 2 に示す。

1. QGRN を保持する個体を N 個設定し初期化
2. 個体の適応度を評価
   (a) QGRN を観測し、b 個の表現型を取り出す
   (b) 取り出した b 個の表現型の適応度を、Fitness 関数を用いてそれぞれ算出
      - Fitness 関数は、対象とする問題に応じて別途定義
   (c) b 個の各表現型の適応度をもとに個体の適応度を算出
      - 各表現型の適応度から個体の適応度を算出する方法は、対象とする問題に応じて別途定義
3. 個体の適応度が上位 N_1 個の個体をエリート個体として保存
4. 適応度が下位 N_2 個の個体に対し、ランダムに選択したエリート個体を複製
5. エリート個体以外の $N - N_1$ 個体に遺伝的操作を実施
6. エリート個体と生成された $N - N_1$ 個の個体を次世代に継承し、2 に戻る

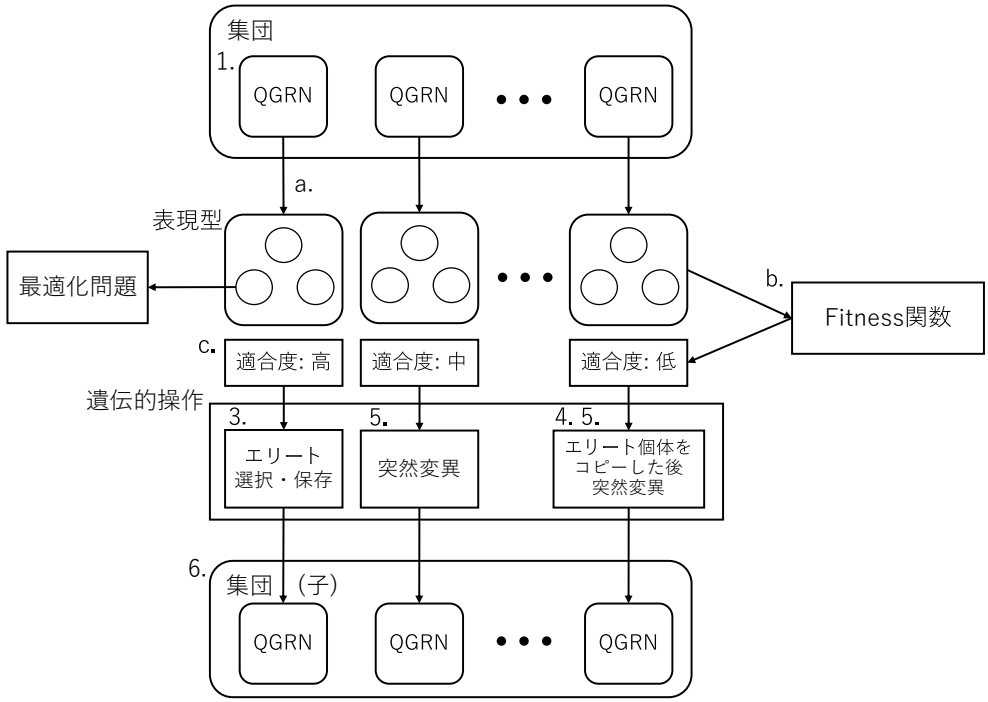


図 2: QGRN を用いた GA による最適化アルゴリズムの全体像

2.3 量子遺伝子制御ネットワークの表現型記憶能力の評価

GRN を量子モデルに拡張した QGRN の特性を確認する。本章では、QGRN の表現型の記憶能力について、GRN と比較評価する。

2.3.1 評価方法

QGRN と GRN に対して、複数の表現型のビット列を記憶させるよう GA を実施し、平均して何個の表現型が記憶可能かを比較評価する。記憶させる表現型は 14 ビットのビット列とし、表 2 に示す 5 つの表現型とする。それぞれ 100 個の QGRN および GRN の集団に対して表現型を記憶させ、Fitness 関数を基に最良の個体を選択し、最良の個体が発現する表現型の上位 6 個に、記憶させる表現型が何個含まれているかで評価する。

QGRN の表現型の記憶に際して、個体の適応度の計算方法を以下の様に設定する。記憶する 5 つの表現型が発現する確率がそれぞれ 20%、それ以外の表現型の発現確率を 0%とした確率分布 p^{target} を生成し、QGRN の表現型の発現確率分布 p^{out} がこれに近づくように Fitness 関数を設定する。具体的には、i 種類目の表現型に対して KL ダイバージェンス [6]

表 2: QGRN と GRN の記憶能力の比較検証時に記憶させる表現型のビット列

	記憶させる表現型のビット列
表現型 1	00100001111101
表現型 2	00101010011111
表現型 3	00001010111111
表現型 4	00100001101001
表現型 5	01110101110101

を用いて、

$$Fitness = -\sum_i p_i^{target} \cdot log(\frac{p_i^{target}}{p_i^{out}}), \tag{6}$$

とする。尚、p^{out} における発現確率の高い上位 6 個の表現型を記憶した表現型とする。

またGRNの数理モデルおよび記憶方法は、文献 [4] と同様とする。具体的にはまず、Wagner の GRN の数理モデルで表現される GRN の隣接行列と遺伝型のビット列を持つ 100 個体を含む集団を用意する。そして、記憶させる 5 つの表現型からランダムに一つを選び、遺伝型および GRN の隣接行列から生成される表現型のビット列と対象のビット列のハミング距離を Fitness の値とする。過学習しない程度の 20 世代の GA を実施し、GRN の隣接行列に対して変異を与える。その後、表現型をランダムに選択しなおし、同様の GA を 10 回繰り返す。上記の手順により生成された各個体において、遺伝型のビット列をランダム値に設定した場合に発現される頻度が高かった表現型の上位 6 個を記憶した表現型として比較評価に用いる。

2.3.2 評価結果

表 3 と 4 に QGRN および GRN の記憶操作を 3 回分の実行結果として、各試行において最も適応度が高い個体が記憶した表現型とその発現を示している。5 個の表現型の記憶において、QGRN における記憶平均数は 4.67 個である。また、GRN の平均は 1.33 個と、QGRN の方が記憶可能な表現型の数が多かった。これは、QGRN では表現型の確率分布を目標に GA を行うことができるため、複数の表現型を同時に記憶可能であることが要因であると考えられる。そのため、QGRN は複数の解を記憶する能力を保持していると考えられる。

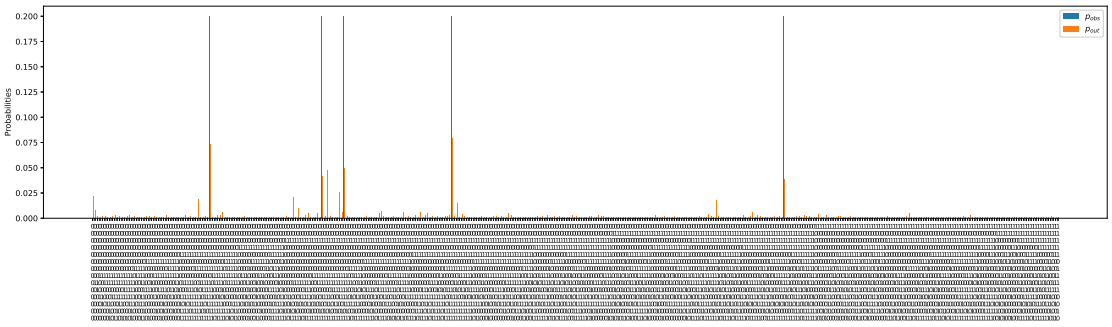
表 3: 各試行において最も適応度が高い QGRN が記憶した表現型とその発現確率

発現順位	シミュレーション 1	シミュレーション 2	シミュレーション 3
1	**00101010011111** (0.079)	**00001010111111** (0.069)	**00101010011111** (0.10)
2	**00001010111111** (0.073)	**01110101110101** (0.051)	**00100001111101** (0.081)
3	**00100001111101** (0.050)	**00100001111101** (0.037)	**00001010111111** (0.068)
4	00100001101101 (0.048)	**00101010011111** (0.031)	**00100001101001** (0.045)
5	**00100001101001** (0.042)	00100011111101 (0.029)	**01110101110101** (0.037)
6	**01110101110101** (0.039)	00000000000000 (0.025)	00000000000000 (0.028)

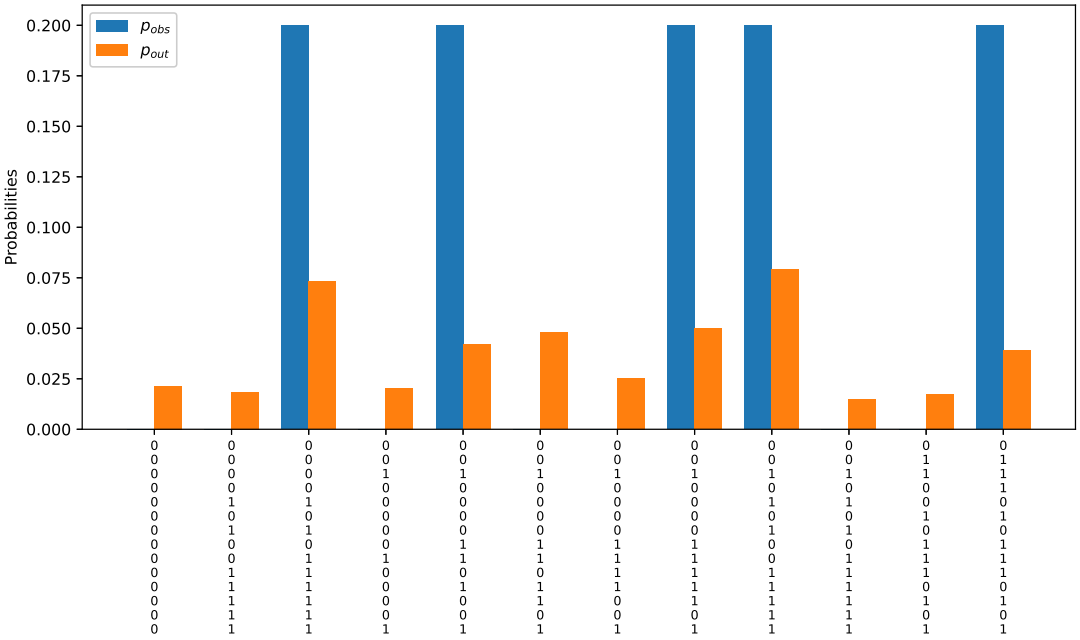
表 4: 各試行において最も適応度が高い GRN が記憶した表現型とその発現確率

順位	シミュレーション 1	シミュレーション 2	シミュレーション 3
1	**01110101110101** (0.06)	**01110101110101** (0.05)	11011110000010 (0.05)
2	10001010001010 (0.05)	01110101110001 (0.04)	**00100001111101** (0.03)
3	01100101110101 (0.03)	01110001110101 (0.03)	00110101110101 (0.03)
4	01110001011101 (0.02)	01110101111101 (0.03)	10001010001010 (0.03)
5	01110001110101 (0.02)	10001010011010 (0.03)	00110001111101 (0.02)
6	01111101110101 (0.02)	00110101110101 (0.01)	**01110101110101** (0.02)

図 3、図 4、図 5 は、学習後の QGRN 集団のうち、最も高い Fitness の値を算出する個体の表現型の発現確率分布であり、3 回の試行で得られた結果をすべて記載している。2^{14} 通りの広い表現型空間のうち、記憶対象である 5 つの表現型がいずれも高い確率で発現されており、上位 6 個のうちの 5 個が記憶対象の表現型となっている。ただし記憶対象以外の表現型についても高い確率で発現する様子も見られ、p^{target} の 20%には少し遠く、5%から 7.5%程度であることから、この QGRN から目標とする表現型を取り出す場合、最低でも 20 回程度の量子ビット観測が必要となることに注意が必要である。

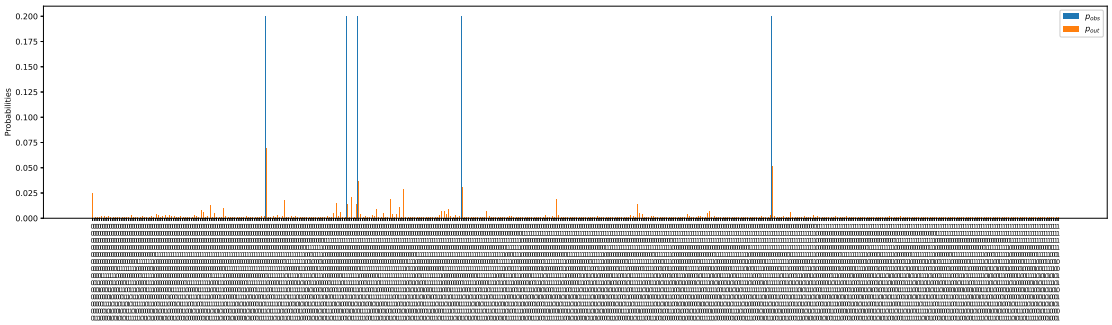


(a) 全表現型の発現分布

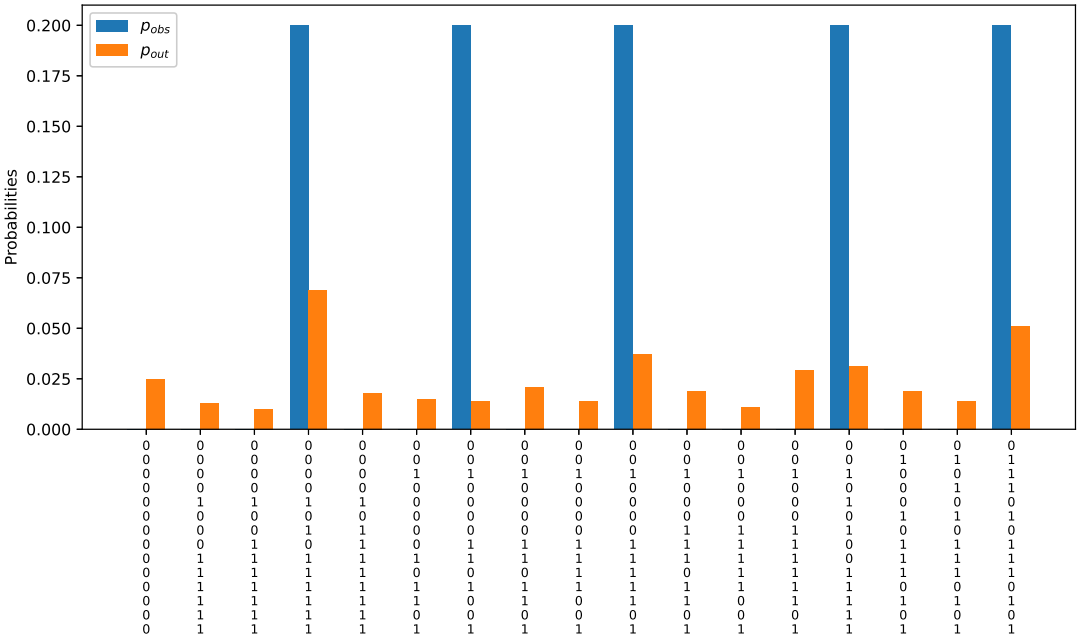


(b) 発現確率が 0.01 以上の表現型のみ

図 3: 記憶能力の検証における QGRN の表現型の発現分布 (試行 1)

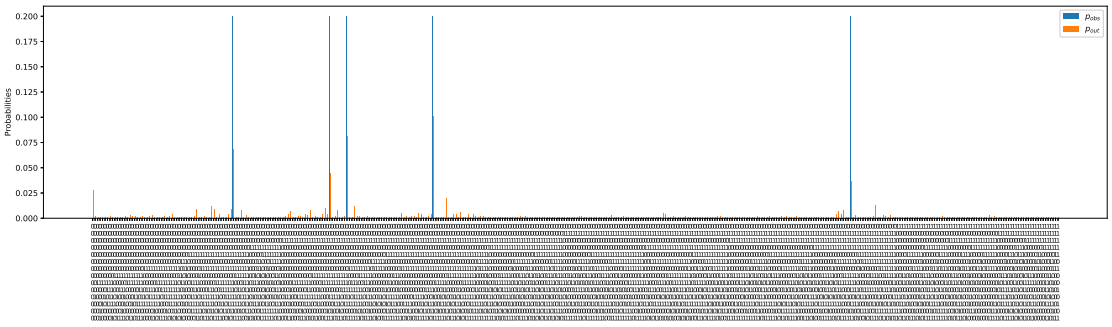


(a) 全表現型の発現分布

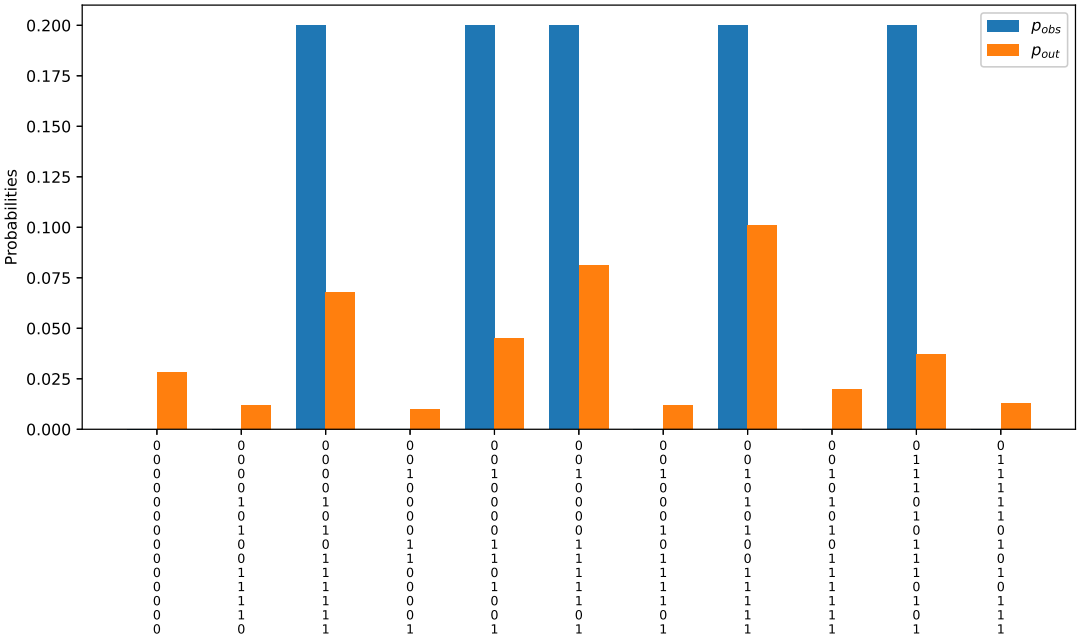


(b) 発現確率が 0.01 以上の表現型のみ

図 4: 記憶能力の検証における QGRN の表現型の発現分布 (試行 2)



(a) 全表現型の発現分布



(b) 発現確率が 0.01 以上の表現型のみ

図 5: 記憶能力の検証における QGRN の表現型の発現分布 (試行 3)

また、QGRN の記憶の中で一般化の傾向も確認することができる。記憶させる表現型の各ビットに着目し、各ビットの値の平均値をとり、それを小数点第一位の位で四捨五入した表現型は、“00100001111101” となる。この時、表 3 で高い確率で発現している表現型のうち “00000000000000” を除いた二つの表現型と、これのハミング距離を比較すると、いずれも 1 ビット差となっている。これは、QGRN が表現型の傾向と似た解を生成する一般化によるものであると考えられる。

図 6 は、QGRN の学習過程において最も高い適応度となった個体の Fitness の遷移である。初期状態から約 1,000 世代程度の短期間で Fitness が大幅に向上している様子が確認で

きる。このように短期間で大きく Fitness が向上し適応することは、ネットワーク環境の変動に対して動的に適応するために必要な特性である。また、1,000 世代以降も Fitness は上昇するものの、その変化は次第に緩やかになり、10,000 世代以降では Fitness はほぼ横ばいである。これに対応して、QGRN から観測される表現型の分布にも顕著な変化は見られなくなることを確認している。

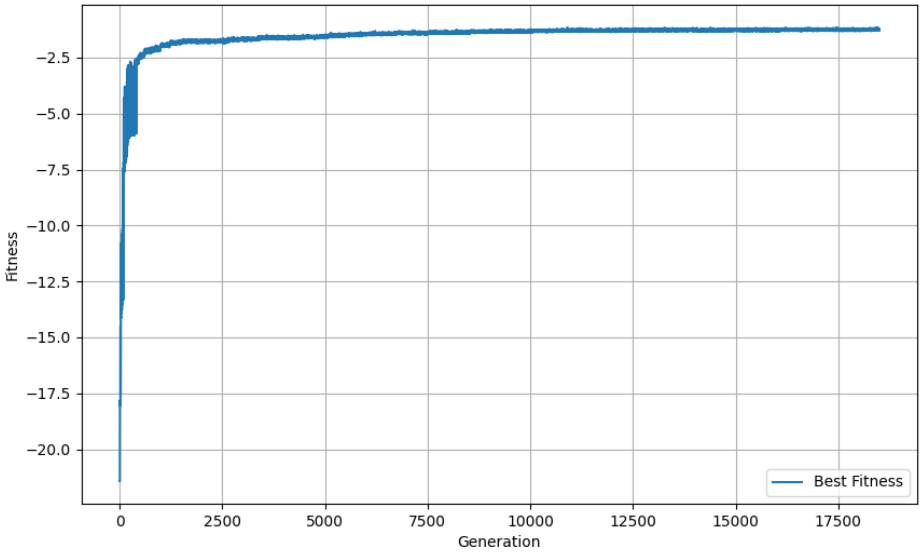


図 6: 記憶能力の検証における QGRN の Fitness の推移

図 7 は GRN における表現型の発現分布を示している。特に今回の GRN モデルでは、複数の表現型を記憶しようとする場合、GA を行う世代数が多くなると過学習となり数個程度の表現型に収束し、世代数が少ないと、偏りのない発現分布となる。今回の GRN モデルでは、記憶能力は世代数によって非常にセンシティブに変化しており、何度も試行すれば複数の表現型が記憶されることもあるが、記憶の不安定さが見られる。記憶が安定的に可能であることも、QGRN の特性の一つであると考えられる。

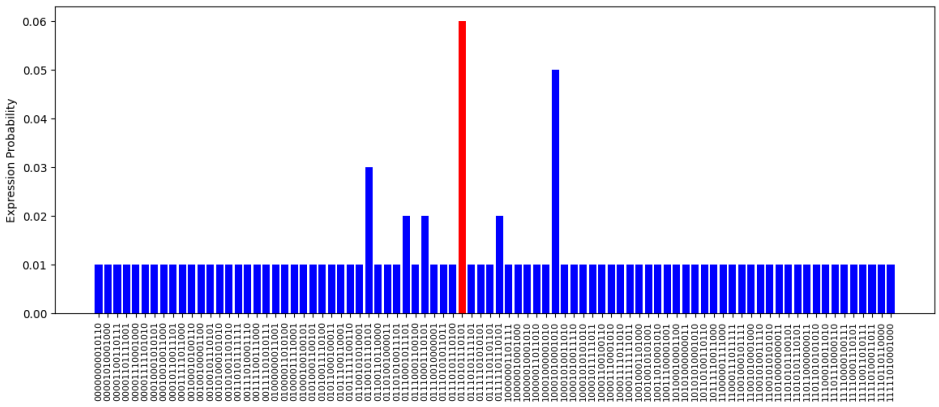


図 7: 記憶能力の検証における GRN の表現型の発現分布

3　モバイルネットワークのスライス埋め込み問題に対する適用検証

ネットワークスライシングとは、異なる通信要件に対して、1 つの物理ネットワークを仮想ネットワークに分割し、それぞれの通信要件を満たすように端末や通信帯域を割り当てる技術である。本章では、ネットワークスライス埋め込み問題の定式化を行う。また、ネットワークやコンピューティングリソースの状況等が変動する環境を設定し、変動する環境下において QGRN を動的に学習させ、環境が切り替わる際の QGRN の挙動を確認し、QGRN の適用性について評価する。

3.1　最適化問題の定式化

文献 [10] のモバイルネットワークのスライス埋め込み問題をもとに、最適化問題を定式化する。

その際、以下の 4 種類のリソースを定義する。

- T: 通信リンクのスループット
  - T_j^s: 物理リンク e_j
  - T_i^k: i 番目の仮想リンク l_i^k
- L: 通信遅延
  - L_j^s: 物理リンク e_j
  - L_i^k: i 番目の仮想リンク l_i^k
- P: 計算能力
  - P_w^s: w 番目の物理クラウドノード c_w
  - P_m^k: k 番目のネットワークスライスのアプリケーション a_m^k
- M: クラウドサーバのメモリ容量
  - M_w^s: w 番目のクラウドノード c_w
  - M_m^k: k 番目のネットワークスライスのアプリケーション a_m^k

3.1.1 物理ネットワークの設定

物理ネットワークのグラフ $\mathcal{N} = (\mathcal{U}, \mathcal{C}, \mathcal{E})$ を無向グラフ $G = (\mathcal{V}, \mathcal{E})$ と頂点 $\mathcal{V} := \mathcal{U} \cup \mathcal{C}$ と定義する。ただし、$\mathcal{U} = \{u_1, \ldots, u_n\}$ はユーザ端末、$\mathcal{C} = \{c_1, \ldots, c_m\}$ をコンピューティングクラウドノードとしてと定義し、リンクの集合は $\mathcal{E} \subseteq \{u_i c_j, c_k c_l\}$ と表記する。

具体的には、図 8 のような物理ネットワークを想定している。ユーザ端末、エッジ、クラウドの階層構造を想定しており、ユーザ端末 $\{u_0, u_1\}$、エッジサーバ $\{c_0, c_1\}$、クラウド $\{c_2, c_3\}$ とする。また、ユーザ端末とエッジサーバ間の通信は無線通信を、エッジクラウド間の通信は有線通信を想定する。上記想定に基づき、コンピューティングリソースおよび通信性能を設定する。

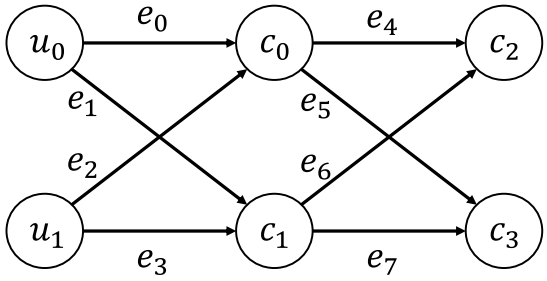


図 8: 評価に用いる物理ネットワークの構成図

3.1.2 仮想ネットワークスライスの設定

仮想ネットワークでは k 番目 $(k = 1, \ldots, n)$ のネットワークスライスを、無向グラフ $\mathcal{N}_k = (\mathcal{U}_k, \mathcal{A}_k, \mathcal{L}_k)$ と表記する。$\mathcal{U}_k \subset \mathcal{U}$ は k 番目のスライスのユーザ端末の集合、$a_m^k \in \mathcal{A}_k$ は k 番目のネットワークスライスに割り当てられている m 番目のアプリケーションノード、$l_i^k \in \mathcal{L}_k$ は k 番目のネットワークスライスの i 番目の仮想リンクとする。また、仮想パス $P_r \in \mathcal{P}_{vw}$ は、始点 u_v から終点 c_w 間の経路を表す。

具体的には、図 9 と 10 の 2 つのネットワークスライスの埋め込みを行う。リンク l_i^k は図 8 の物理ネットワーク上の経路（$\{e_0, e_0e_4, e_0e_5, e_1, e_1e_6, e_1e_7, e_2, e_2e_4, e_2e_5, e_3, e_3e_6, e_3e_7\}$ のいずれかに割り当てられる。また、アプリケーション a_w^k は、図 8 物理ネットワーク上のクラウド $\{c_0, c_1, c_2, c_3\}$ のいずれかに割り当てられる。

3.1.3 変数定義

目的関数および制約条件の制定のため、以下のように変数を定義し、物理ネットワークと仮想ネットワークスライスの関係を表現する。$a2c_{mw}^k$ はネットワークスライス k において

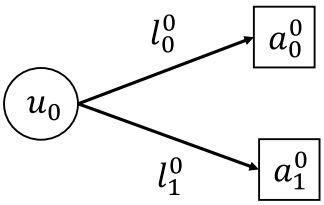


図 9: 評価に用いるネットワークスライス 1

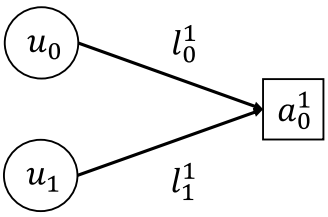


図 10: 評価に用いるネットワークスライス 2

m 番目のアプリケーションノードがクラウドノード c_w にマッピングされているかを示すバイナリ変数であり、$l2e_{ij}^k$ はネットワークスライス k において仮想リンク i が物理リンク j にマッピングされているかを示すバイナリ変数である。

- アプリケーションノードと物理クラウドノードのマッピングを表す変数:

$$a2c_{mw}^k := \begin{cases} 1 & \text{if } a_m^k \text{ is mapped on } c_w, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

- 仮想リンクと仮想パスのマッピングを表す変数:

$$l2p_{ir}^k := \begin{cases} 1 & \text{if } l_i^k \text{ is mapped on } P_r \in \mathcal{P}_{vw}, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

- 仮想パスと物理リンクのマッピングを表す変数:

$$p2e_{rj} := \begin{cases} 1 & \text{if } e_j \text{ is used in } P_r, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

- 仮想リンクと物理リンクのマッピングを表す変数:

$$l2e_{ij}^k := \sum_r (l2p_{ir}^k \cdot p2e_{rj}). \tag{10}$$

3.1.4 意思決定変数

ネットワークスライス上のリンク l_i^k を、物理ネットワーク上でどのエッジに埋め込むか、また、ネットワークスライス上のアプリケーション a_w^k を、物理ネットワーク上のどのクラウドノードに埋め込むかを決定変数とする。

3.1.5 目的関数

すべての仮想ネットワークが制約条件を満たして正常に埋め込まれ、かつ、ネットワーク全体の消費電力が低減することを目的とし、目的関数を、

$$Fitness = \begin{cases} -\alpha & \text{if } \delta = 0, \\ \\ -\sum_k \sum_i (\omega_i \cdot T_i^k) - \sum_k (\omega_m \cdot P_m^k + B_m) & \text{if } \delta = 1, \end{cases} \tag{11}$$

とする。このとき δ は、すべての仮想ネットワークの埋め込みが制約条件を満たしたときに $\delta = 1$、失敗した場合に $\delta = 0$ となる変数である。また B_m は、アプリケーション a_w^k の埋め込み先クラウド c_m のベース電力を、ω_m はその重みを表す。また ω_i は、物理リンク e_i の重みを示しており、コンピュータ使用時の消費電力とネットワーク使用時の消費電力を計算に含めている。消費電力の計算では、アプリケーションが要求する計算能力に応じてコンピュータ使用時の消費電力が増加し、埋め込み対象のリンクが要求する通信帯域によってネットワーク使用時の消費電力が増加する。一方、使用されていない物理リンクやクラウドノードについては消費電力計算から除いている。なお、本報告では定数 $\alpha = 10000$ を採用している。

3.1.6 制約条件

ネットワークおよびコンピューティングリソースに関する制約条件を設定する。ネットワークに関しては、スループットおよび遅延が、コンピューティングリソースに関しては、計算能力およびメモリ容量が、上限値を超えないことである。以降では各項目について記載する。

まずネットワークの制約条件として、スループットの制約を、式 (12) のように表記する。これは、物理ネットワーク内の各リンクの総スループット容量が、そのリンクにマッピングされた全仮想リンクのスループット要求の合計を超えないことを意味する。

$$\sum_k \sum_i l2e_{ij}^k \cdot T_i^k \leq T_j^s, \quad \forall j \tag{12}$$

また、遅延制約は式 (13) と設定する。これは、各ネットワークスライスの仮想リンクに許容される遅延が、対応する物理経路におけるエッジ遅延の合計以上でなければならないことを示している。

$$\sum_j l2e_{ij}^k \cdot L_j^s \leq L_i^k, \quad \forall k, \forall i \tag{13}$$

次に、コンピューティングリソースの制約条件である計算能力およびメモリ容量の制約は、それぞれ式 (14) および式 (15) のように設定する。これらは、物理ネットワーク内のクラウドノードが、マッピングされたアプリケーションノードによる要求の合計を超えないことを示している。

$$\sum_{k}\sum_{m} a2c_{mw}^{k} \cdot P_{m}^{k} \leq P_{w}^{s}, \quad \forall w \tag{14}$$

$$\sum_{k}\sum_{m} a2c_{mw}^{k} \cdot M_{m}^{k} \leq M_{w}^{s}, \quad \forall w \tag{15}$$

3.2 評価環境の設定

3.2.1 QGRN の表現型とネットワーク埋め込み方法とのマッピング

図 11 のような 14bit の表現型を設定し、QGRN の表現型から、ネットワークスライシングの埋め込み方法へのマッピング方法とする。上位 8bit は 1 つ目のスライスに対応し、下位 6bit は 2 つ目のスライスに対応する。リンク l_i^k の埋め込み経路は 2bit で表記し、上位 bit は $\{c_0, c_1\}$ のどちらを通る経路を選択するか設定しており、0 のときは c_0 に、1 のときは c_1 へ向かう経路を選択し、下位 bit は $\{c_2, c_3\}$ のどちらを通る経路を選択するか設置しており、0 のときは c_2 に、1 のときは c_3 へ向かう経路を選択する。また、アプリケーション a_w^k の埋め込み先は 2bit で表記し、2bit を十進数に変換した数値 x に対し、c_x に埋め込む。

	リンクl_0^0の経路 2bit(4経路)		リンクl_1^0の経路 2bit(4経路)		アプリa_0^0の埋め込み先クラウド 2bit(4個)		アプリa_1^0の埋め込み先クラウド 2bit(4個)	
slice1	0	1	0	1	0	1	0	1
	リンクl_0^1の経路 2bit(4経路)		リンクl_1^1の経路 2bit(4経路)		アプリa_0^1の埋め込み先クラウド 2bit(4個)			
slice2	0	1	0	1	0	1		

図 11: シミュレーション上での表現型とネットワーク埋め込み方法とのマッピング設定

3.2.2 変動する環境の設定

本評価では、物理ネットワークにおける通信帯域と通信遅延、クラウドノードにおける計算能力とメモリ容量、スライスにおける通信帯域と遅延要件および計算能力とメモリ使用

量、が 3 パターンに変化することを想定する。変動する環境として、表 5 のような 3 種類の環境を設定する。また、各環境における全ての埋め込みパターンにおける Fitness の値を算出し、最適解となる表現型が表 6 のようになることを確認している。尚、環境 A,B,C におけるネットワークスライスの埋め込み方法はそれぞれ、図 12、図 13、図 14 のように異なるものとなっている。

表 5: 変動するネットワーク環境設定

設定値	環境 A	環境 B	環境 C
物理ネットワーク設定			
通信帯域 (T_j^s)	(596, 623, 537, 588, 2235, 2311, 2445, 2051)	(731, 506, 528, 526, 2391, 2464, 2436, 2435)	(742, 590, 681, 548, 2375, 2140, 2290, 2118)
通信遅延 (L_j^s)	(13, 13, 13, 12, 6, 4, 6, 5)	(9, 11, 10, 9, 6, 7, 7, 6)	(10, 12, 10, 10, 4, 7, 7, 5)
通信経路の重み (ω_i)	(0.36, 0.37, 0.34, 0.38, 0.16, 0.11, 0.17, 0.17)	(0.32, 0.32, 0.34, 0.38, 0.15, 0.20, 0.16, 0.15)	(0.33, 0.36, 0.31, 0.39, 0.15, 0.20, 0.12, 0.11)
計算能力 (P_w^s)	(482, 657, 15000, 23000)	(415, 504, 15000, 23000)	(455, 524, 15000, 23000)
メモリ容量 (M_w^s)	(742, 745, 6000, 7500)	(742, 736, 6000, 7500)	(671, 732, 6000, 7500)
ベース電力 (B_m)	(485, 464, 1088, 1150)	(460, 471, 1088, 1189)	(474, 485, 1251, 1304)
クラウドの重み (ω_m)	(1.22, 1.62, 2.45, 2.28)	(1.33, 1.37, 2.05, 2.79)	(1.49, 1.25, 2.50, 2.02)
リンク要件			
通信帯域 (T_i^k)	(427, 233, 63, 81)	(363, 306, 316, 236)	(303, 402, 377, 147)
遅延要件 (L_i^k)	(23, 16, 15, 14)	(19, 19, 16, 15)	(14, 15, 17, 16)
アプリケーション要件			
計算能力 (P_m^k)	(325, 253, 132)	(766, 776, 761)	(461, 565, 507)
メモリ使用量 (M_m^k)	(145, 75, 127)	(700, 547, 695)	(455, 666, 573)

表 6: 環境 A, B, C における最適解の表現型と Fitness。表現型において $*$ は 0 または 1 のいずれでもよい。

環境	最適解の表現型	Fitness
A	0∗1∗00011∗1∗01	-2263.22
B	00101010011111	-8168.18
C	00001010111111	-6733.23

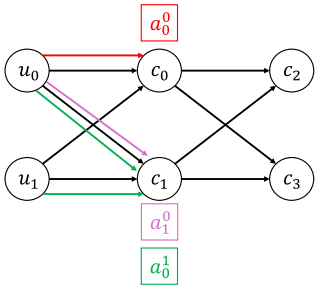


図 12: 環境 A での最適なネットワークスライス埋め込み方法

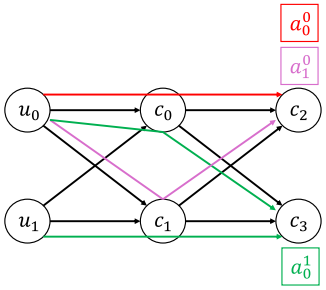


図 13: 環境 B での最適なネットワークスライス埋め込み方法

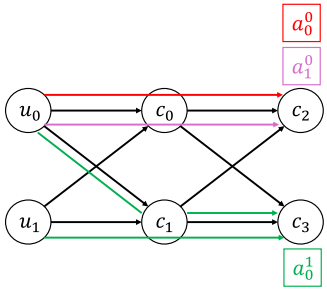


図 14: 環境 C での最適なネットワークスライス埋め込み方法

これらの環境を 30 分毎に環境 A、環境 B、環境 C、環境 A、⋯ と 24 時間周期的に変動させたとき、QGRN が環境変動後に短い時間で最適解となる表現型を発現可能か確認する。

3.2.3 変数設定

本アルゴリズムでは、QGRN の最適化に GA を用いている。ネットワークスライス埋め込み問題への適用に際し、表 7 のように設定を行っている。また、比較対象として用いる QGRN を使用しない GA の設定については、表 8 に記載している。

表 7: ネットワークスライス埋め込み問題に適用する際の QGRN における GA の設定

項目	内容
個体数	100 個体
エリート個体数	10 個体
コピー戦略	fitness が下位 20 個体はエリート個体のコピー
遺伝的操作の対象	エリート個体を除く 90 個体
遺伝的操作の種類	突然変異のみ
突然変異の詳細	14×14 行列である QGRN の各要素に対し、突然変異率 0.05 でランダム値（$-\frac{\pi}{2}$ から $\frac{\pi}{2}$ の範囲）に変更
QGRN から発現する表現型の数	5 個
Fitness 関数	発現した表現型をネットワークスライス埋め込み問題の目的関数に適用し、その中で最大の Fitness を QGRN の適応度に設定

表 8: QGRN を用いない GA の設定

項目	内容
個体数	100 個体
エリート個体数	10 個体
選択方式	トーナメント+エリート選択
交叉	1 点交叉
突然変異	1bit 反転
交叉率	0.6
突然変異率	0.05

3.2.4 実行環境

QGRN の演算には、量子コンピューティングフレームワークである Qiskit の量子シミュレータを使用する。評価に用いた計算機の詳細は表 9 に示す。

表 9: 評価に用いた計算機とソフトウェアの設定

項目	内容
プロセッサ名	Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz
スレッド数	64 スレッド
コア数	32 コア (16 コア × 2 ソケット)
メモリ容量	131.65 GB
OS	Ubuntu 20.04.6 LTS
使用したソフトウェア	Python 3.11.6

3.3 評価結果

前節までの環境設定に基づき、シミュレーションを 3 回実行した結果を図 15、 16、 17 に示す。また、環境変動における最適解到達率および最適解に到達した場合の到達時間についての評価を表 10 にまとめた。

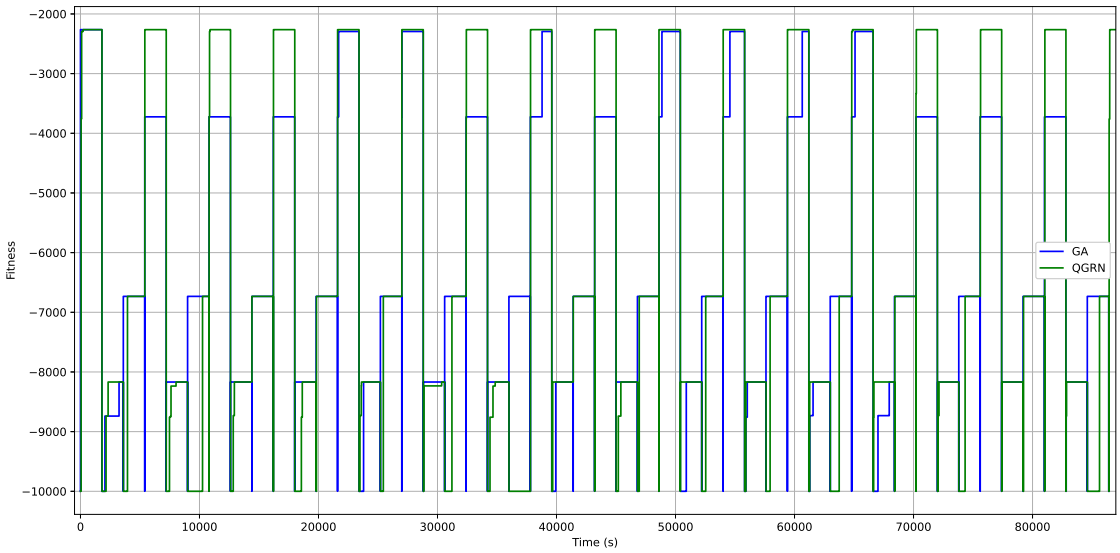


図 15: 環境 A,B,C が 30 分ごとに変動する環境下における fitness の推移（試行 1）

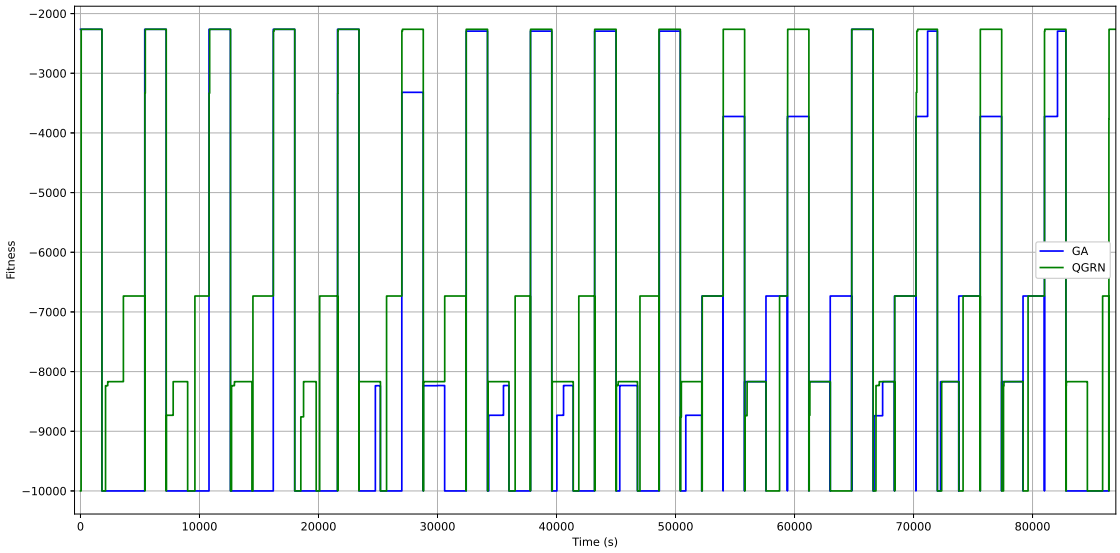


図 16: 環境 A,B,C が 30 分ごとに変動する環境下における fitness の推移（試行 2）

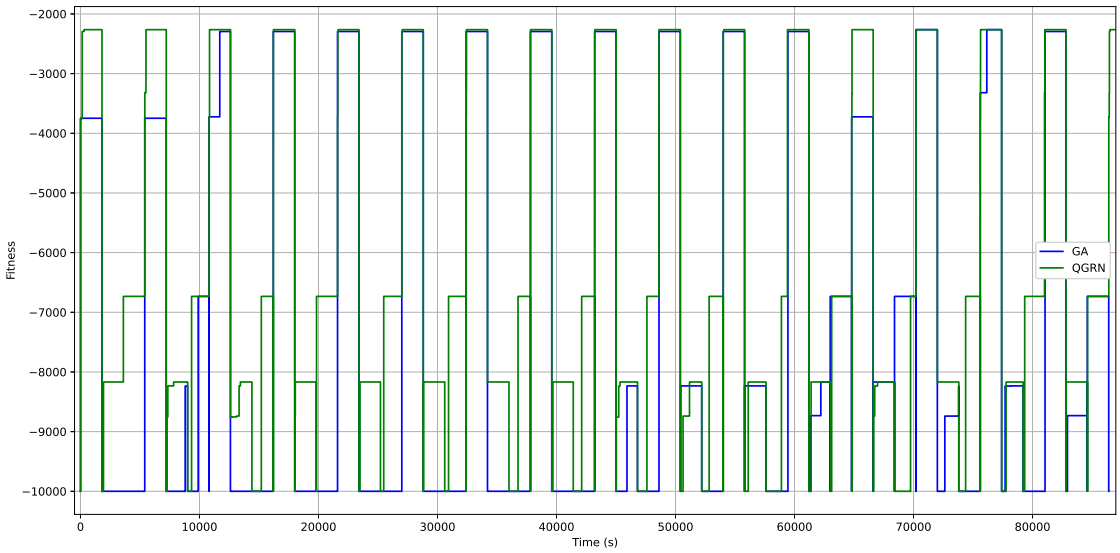


図 17: 環境 A,B,C が 30 分ごとに変動する環境下における fitness の推移（試行 3 ）

表 10: 環境変動時における QGRN と GA の最適解到達時間

	評価方法	**環境 C→A**	**環境 A→B**	**環境 B→C**
QGRN	最適解到達率	100% (48/48 回)	100% (48/48 回)	95.83% (46/48 回)
	平均値（秒）	37.13	431.83	427.23
	中央値（秒）	22.99	322.43	334.70
	標準偏差（秒）	56.12	412.11	398.59
GA	最適解到達率	18.75% (9/48 回)	25% (12/48 回)	54.17% (26/48 回)
	平均値（秒）	37.25	709.61	29.58
	中央値（秒）	0	596.34	0.15
	標準偏差（秒）	143.68	444.47	157.31

表 10 より、環境が変動した場合において、QGRN は高い確率で最適解に到達することが可能である。一方、GA は最適解に到達することが少なく、図 15、図 16、図 17 によると局所解に陥っている様子が多くみられる。これは QGRN が、従来の GRN モデルと同様に、遺伝型の相互作用が少し変動した際に表現型が大きく変動することで、局所解への収束が回避可能であったことが要因である。

特に環境 C から環境 A に変化する場合が最も顕著な差が見られるため、GA における環境 C における最適解の表現型と環境 A における局所解および最適解の表現型の関係性について調査した結果を、図 18 に示す。環境 C の最適解から環境 A の局所解 1 または局所解 4

への遷移が多く観測されている。これは、環境 C の最適解から局所解 1 のハミング距離は 3、局所解 1 から局所解 4 へのハミング距離は 2 と小さく、解同士の距離が近いことにより環境 C の最適解から遷移しやすいことが原因である。環境 A の最適解へ到達するには、局所解 2 または局所解 3 を経るか、直接環境 A の最適解に到達するパターンが確認されている。これらは環境 C の最適解からのハミング距離が大きいため、到達頻度が低いと考えられる。また、環境 C から環境 A のケースに限らず、他の複数のケースにおいても同様の結果が観測されている。このような GA では最適解に到達することが困難な状況下においても、QGRN を用いた GA では、相互作用の変動により複数の表現型の変異を発生させ、最適解に到達することを可能としている。

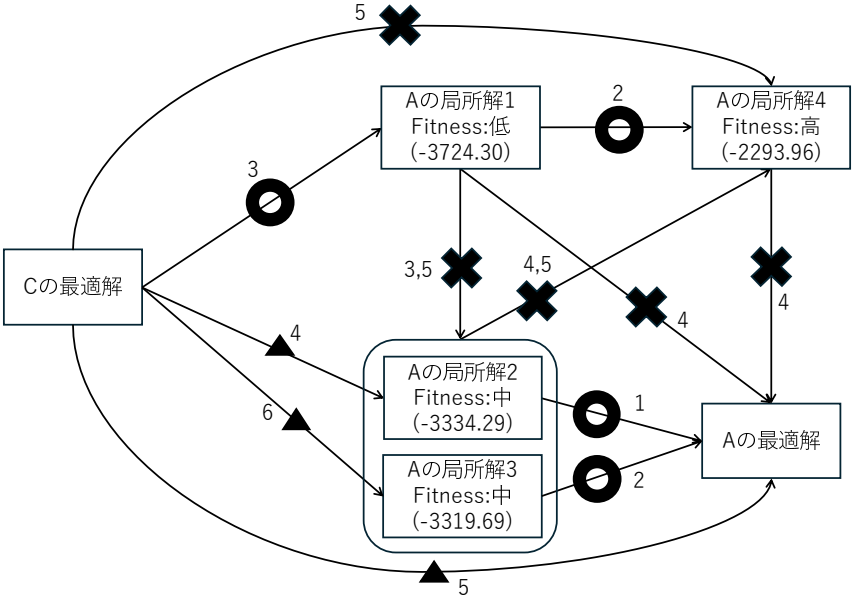


図 18: 環境 C の最適解と環境 A において出現回数の多い解（最適解および局所解）の関係性。図中の数字は解同士のハミング距離を表し、〇、△、×はそれぞれ遷移のしやすさ（〇は頻繁に観測された遷移、△は数回観測された遷移、×は一度も観測されなかった遷移）を示す。

一方で、環境 B から環境 C に変化するケースにおいては、GA は最適解に到達した割合は低いものの、到達した場合は平均 29.58 秒で中央値が 0.15 秒と、QGRN の平均 427.23 秒、中央値 334.70 秒に比べて短い時間で最適解に到達する傾向がある。これは環境 B と環境 C の最適解の表現型のハミング距離が 2 と小さく、GA が到達しやすい解の関係であることが要因と考えられる。QGRN においては、以前に環境 C にて到達した最適解の表現型が記憶されていることが理想である。しかし、今回の最適化手法の設定において、QGRN の記憶

が保持されることはごく一部のケースに限られていた。これは、QGRN が変動した環境に適応する解一つのみを記憶するよう作用したことが原因であると考えられる。Fitness 関数等の工夫により、以前の環境を保持するように QGRN の最適化を行うことで、過去の記憶を想起し、このようなケースにおいても QGRN が短時間で最適解に到達することができないかと考えている。

尚、GA において、環境 C から A に変動するケースで、中央値が 0 秒となっているが、これは、一度環境 A で最適解に到達し、環境 B および環境 C において適切な解が発見されず、環境 A の解が保持されたまま再び環境 A に戻る現象が多く発生したためである。この結果は、GA のパラメータ設定が環境変動に対して十分な適応力を持たない可能性を示しており、GA のパラメータ調整が今後の課題である。

環境 A から環境 B への変動においても、QGRN が GA よりも短時間で最適解に到達しているが、環境 A と B の最適解の表現型は最小でも 5 ビットの差があり、GA では時間がかかったが、QGRN では複数の表現型ビットの同時変異により探索時間が短縮されたことから短い時間で最適解に到達したと考えられる。

3.4 考察

これらの結果から、QGRN において、既存の GRN モデルと同様に、GRN の遺伝子間の制御作用の変動による複数の表現型ビットの同時変動によって、局所解を回避して最適解に到達する様子が確認された。

一方で、変動環境における記憶機能および記憶機能による一般化された表現型の発現能力については限定的なものとなった。今後の課題として、Fitness 関数の作り方や、GA の方法を工夫することで、これらの様子が確認できないか検討する必要がある。

今回の評価では、量子コンピュータの実行時間および利用料の観点から、量子シミュレータである Qiskit を用いてシミュレーションを行った。そのため、ビット数と演算時間の制約から、小規模なネットワークモデルでの評価となっている。将来的な量子コンピュータのビット数拡大や演算速度の向上および一般化によって、より大きな規模でのネットワークモデルにも適用できるようになると考えられる。具体的には、6G 通信システムの実現が見込まれる 2029 年頃には、100 万量子ビットの量子コンピュータの実現がマイルストーンとされている [7]。対象とするアプリケーションやネットワーク構成によって大きく異なるものの、100 万量子ビットを用いた場合、1 端末と 1 アプリケーションの最も単純なスライス構成において、最大で端末 10 万台、エッジサーバ 1,000 台、クラウドサーバ 100 台規模のネットワークを表現型として表現可能となる。6G では、1 平方キロメートルあたり最大 1000 万台の接続端末を目標としている [12] ことから、100 万量子ビットを用いることで、最も端末の

密度が高い地域においても、100 m四方の範囲内での通信要求に対応できる規模感となる。実際には、QGRN のパラメータ $\theta_{i,j}$ の要素数も増大するなど、適用可能性は未知ではあるが、大規模なネットワークモデルでの検証も今後の課題の一つである。

4　おわりに

本報告では、量子遺伝子制御ネットワークを用いた最適化アルゴリズムを提案した。複数の表現型を記憶する能力について評価し、量子遺伝子制御ネットワークモデルが Wagner モデルの遺伝子制御ネットワークモデルよりも安定的に高い記憶能力を示すことを確認した。また、量子遺伝子制御ネットワークが、遺伝子制御ネットワークモデルと同様に、一般化の能力を有することも確認した。環境変動が想定されるネットワークスライシング埋め込み問題を対象に、量子遺伝子制御ネットワークを活用した遺伝的アルゴリズム手法の性能評価を行った。環境変動後に最適解に到達する時間を計測したところ、量子遺伝子制御ネットワークを用いることで、最適解への到達確率が向上した。これは、量子遺伝子制御ネットワークにおいても、遺伝子制御ネットワークと同様に、制御ネットワークの一部の変動によって表現型が大きく変化し、遺伝的アルゴリズムの問題点である局所解への収束を回避できたことを意味している。一方、局所解に陥ることがなく、表現型の数ビット程度の変異で最適解に到達できるような場合は、量子遺伝子制御ネットワークを用いない遺伝的アルゴリズムの方が最適解に短時間で到達した。数ビット程度の変異であれば短時間で演算可能な単純な遺伝的アルゴリズムの方が早いためである。

以上の結果から、量子遺伝子制御ネットワークが遺伝子制御ネットワークの特徴を有していることが分かった。そのため、量子遺伝子制御ネットワークモデルを用いることで、今後の量子コンピュータの発展および一般化に伴い、量子コンピュータを用いることで、大規模な問題に対する遺伝子制御ネットワークの演算時間の増大に対処できることが見込まれる。

今回の変動環境では、量子遺伝子制御ネットワークが過去の表現型を記憶する機構が作用しにくかった。Fitness 関数の作り方の工夫などにより、このようなケースでも量子遺伝子制御ネットワークが短時間で最適解を発見することも可能になると考えられるため、今後の課題の一つである。

また本評価は、量子コンピュータの制約により、量子演算を古典コンピュータ上で実施したため、量子ビット数が少なく規模が小さいネットワークにて性能評価を実施した。将来的な量子コンピュータの性能向上や一般への普及により、より演算可能な量子ビット数や演算コストが軽減されると考えられるため、大規模なネットワークに対する量子遺伝子制御ネットワークの適応結果については今後の課題である。

また、QGRN の特徴である c-R_y ゲートを用いた表現能力の効果については今回の報告では評価できていない点の一つであり、今後の課題である。

謝辞

本報告の遂行にあたり、ご多忙の中で多大かつ貴重なご助言を賜りました大阪大学大学院情報科学研究科の村田正幸教授に心より深く感謝申し上げます。さらに、大阪大学大学院情報科学研究科の山内雅明特任助教には、研究の着想から調査、論文執筆に至るまで、多大なるご助力とご指導をいただき、心よりお礼申し上げます。また、平素よりご指導・ご教示を賜った大阪大学 D3 センターの下西英之教授、大阪大学大学院情報科学研究科の荒川伸一准教授、大阪大学先導的学際研究機構の小南大智准教授、ならびに大阪大学大学院経済学研究科の大歳達也助教にも深く感謝いたします。最後に、日頃から支えてくれた家族、友人、そして研究室の皆様に対して、心より感謝の意を表し、謝辞とさせていただきます。

参考文献

[1] E. H. Davidson and D. H. Erwin, "Gene regulatory networks and the evolution of animal body plans," *Science*, vol. 311, pp. 796–800, Feb. 2006.

[2] 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, pp. 1124–1138, Apr. 2014.

[3] P. A. Dnyane, S. S. Puntambekar, and C. J. Gadgil, "Method for identification of sensitive nodes in boolean models of biological networks," *IET Systems Biology*, vol. 12, pp. 1–6, Feb. 2018.

[4] S. Inoue, M. Yamauchi, D. Kominami, H. Shimonishi, and M. Murata, "Genetic algorithm with gene regulatory networks based optimization method for distributed video analysis system," in *Proceedings of 27th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, pp. 257–264, Mar. 2024.

[5] A. Wagner, "Does evolutionary plasticity evolve?," *Evolution*, vol. 50, pp. 1008–1023, June 1996.

[6] C. Roman-Vicharra and J. J. Cai, "Quantum gene regulatory networks," *npj Quantum Information*, vol. 9, July 2023.

[7] IBM, "Quantum roadmap." Accessed: 2024-12-24.

[8] R. M. Jay Gambetta, "Ibm quantum delivers on performance challenge made two years ago." Accessed: 2025-2-4.

[9] "6G requirements and design considerations," white paper, NGMN alliance - white paper, Feb. 2023. Accessed: 2024-12-24.

[10] A. Fendt, S. Lohmuller, L. C. Schmelz, and B. Bauer, "A network slice resource allocation and optimization model for end-to-end mobile networks," in *Proceedings of 2018 IEEE 5G World Forum (5GWF)*, pp. 262–267, 2018.

[11] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, pp. 8091—8126, Oct. 2020.

[12] “ドコモ 6G ホワイトペーパー 5.0 版,” white paper, 株式会社 NTT ドコモ, Nov. 2022. Accessed: 2025-2-6.