# 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を用いた最適化手法

関澤 和希[1], 山内 雅明[1], 小南 大智[1], 下西 英之[2], 大歳 達也[3], 村田 正幸[1]

[1]大阪大学大学院情報科学研究科

[2]大阪大学D3センター

[3]大阪大学大学院経済学研究科

NS研究会　2025.5.16

# 目次




- **研究背景**
  - 遺伝子制御ネットワーク
  - 先行研究の課題
- **研究目的とアプローチ**
- **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**
- **記憶性能の評価**
- **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**

# 遺伝子制御ネットワーク (GRN: Gene Regulatory Networks)



- **生物の遺伝子間の調節的相互作用を記述したネットワーク構造[2]**
  - 遺伝子型 (DNAに含まれる生物の遺伝情報) から
    表現型 (細胞や臓器等) へのマッピング
- **特定の表現型を発現しやすい形状に変化し、表現型を記憶[3]**
  - 生物は周囲の環境が大きく変動した際、環境に適応できる細胞（表現型）を構築できるよう、遺伝子制御ネットワークを変化させ、進化する
  - 過去環境で有効であった制御パターンがGRNの内部構造に保持 (記憶) され、同様の環境に変動した際、想起可能[1]
- **一か所の遺伝子の変異により、複数の表現型が同時に変化[3]**
  - 環境に適応するための進化が効率的に行える

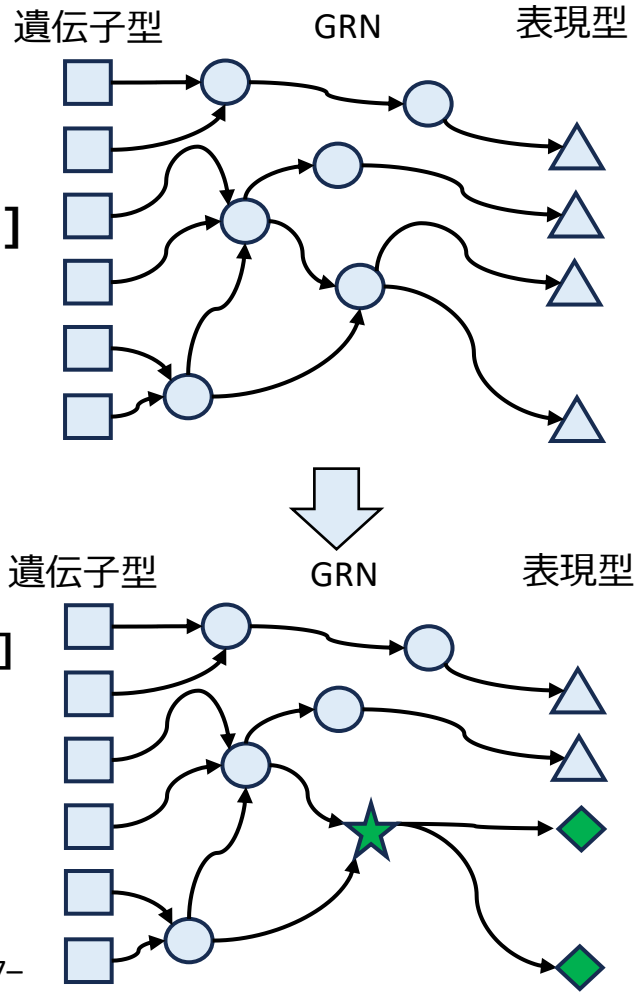


図1：一部の遺伝子の変化により表現型が大規模に変化

[1] 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.
[2] 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.
[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.

# GRNを用いたNW最適化と課題



- **GRNを用いた遺伝的アルゴリズム (GA) による動的なNW制御手法[1]**
  - 映像分散処理システムのリソース割り当て制御の最適化問題に適用
    - 特定のNW環境における割り当て最適解と対応するビット列を、GRNが発現しやすいよう事前学習
    - 遺伝子型をGAで変異させることで、最適解が想起され、類似のNW環境に短時間で適応可能
    - 遺伝子型の1つのビットの変異により、表現型の複数ビットが同時に変異し、局所解から脱出
      - GAでは、複数ビットの表現型を同時に変異できず、局所解に陥り抜け出せなくなった
- **課題**
  - GRNモデルが記憶できる表現型の数は限定的で、構築には複数回の試行が必要
  - 事前にGRNモデルの構築が必要

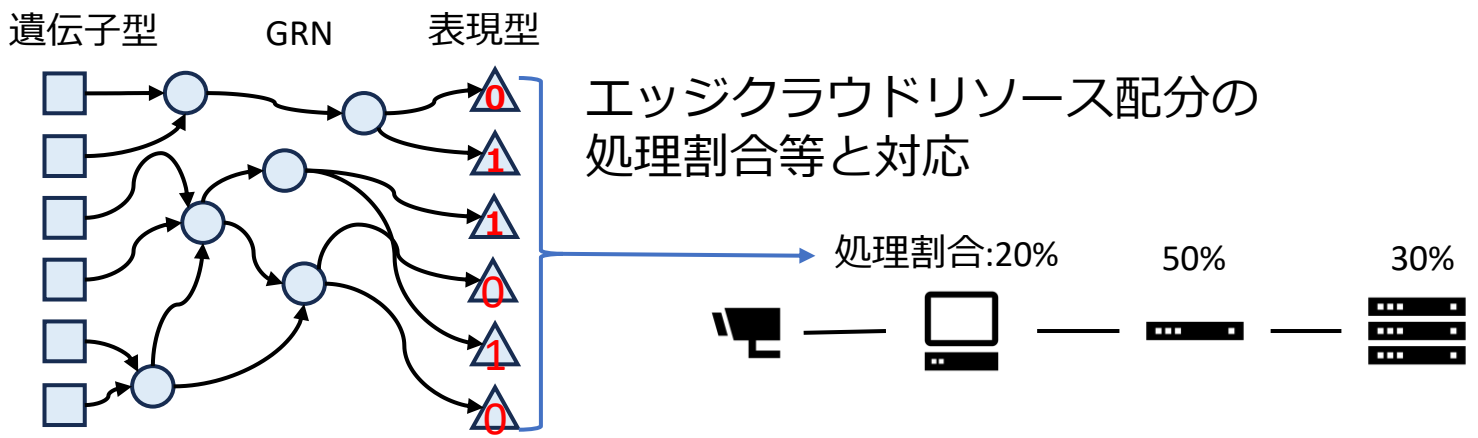


図2 : GRNを用いたGAによるNW制御手法

[1] 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.

# 研究目的とアプローチ



- **研究目的**
  - 複数の表現型を安定的に記憶できるGRNモデルの構築
  - 従来のGRNと同様、局所解に陥ることなく最適解に到達
- **アプローチ**
  - 複数の表現型を確率的に発現可能なQuantum GRNの適用
    - 複数の表現型を安定的に記憶できるのではないか
  - 動的なNW環境でのネットワークスライス埋め込み問題で最適解に到達可能か検証
    - ネットワークスライシング：1つの物理ネットワークを通信要求に応じて仮想的にスライスし、リソースを物理ネットワークに割り当てる技術

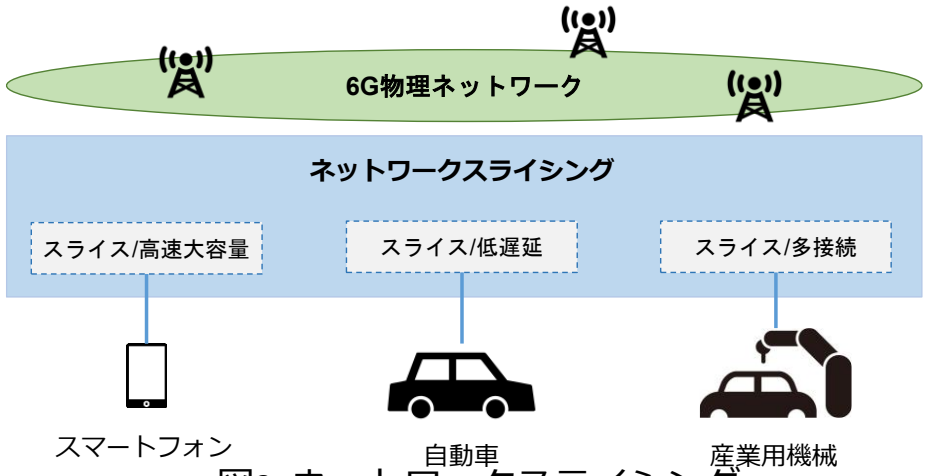


図3: ネットワークスライシング

# 量子遺伝子制御ネットワーク (QGRN: Quantum GRN)[5]



- **遺伝子間の相互作用を量子ゲート演算として表現**
  - 発現する表現型および記憶する表現型を確率分布として扱うことが可能
    - 既存のGRNと比較して、複数の表現型の記憶に適している
  - 量子コンピュータに適用可能なモデルであるため、将来的には高速な計算が期待できる

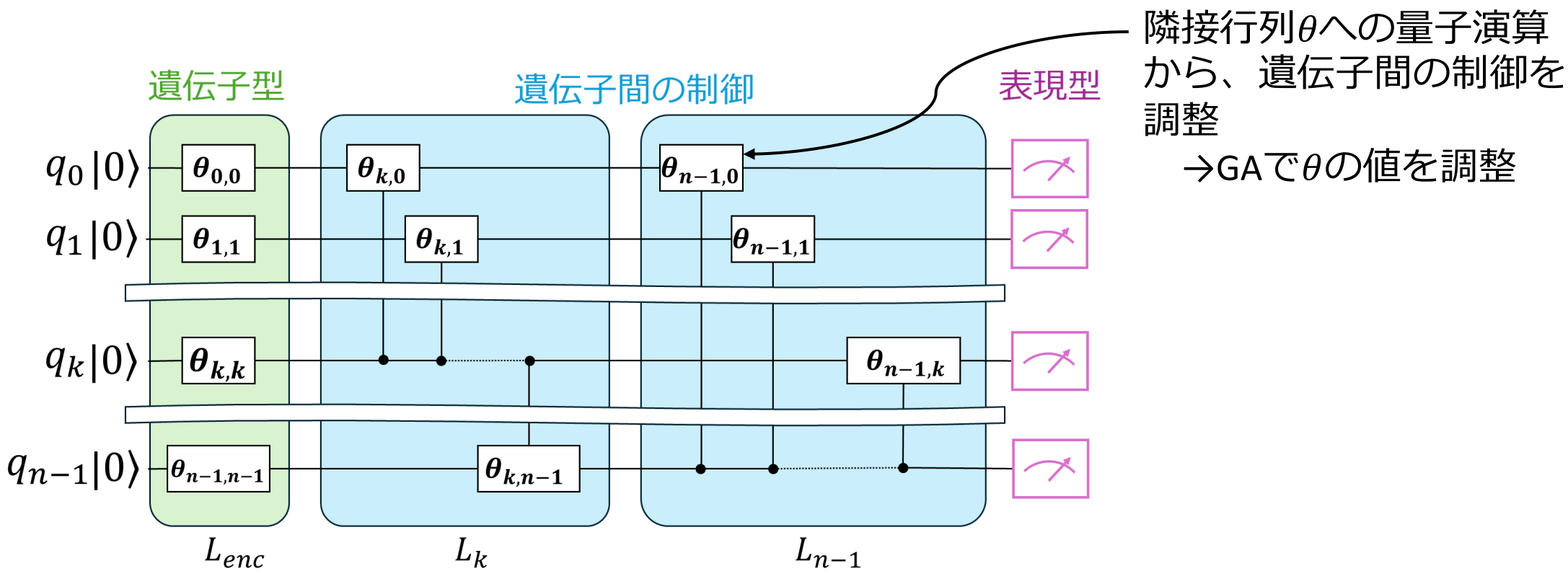


図4 : QGRNの数理モデル

[5]C. Roman-Vicharra and J. J. Cai, “Quantum gene regulatory networks,” npj Quantum Information, vol. 9, July 2023.

# 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



- **目的**
  - 最適化問題に対して適応度の高い表現型を発現するQGRNを構築
- **アルゴリズムの流れ**
  1. QGRNを個体とする集団を形成
  2. 各個体の適応度を評価
     a. QGRNの量子ビットの観測を行い、上位n個の表現型を取得
     b. 各表現型の適応度を算出し、各個体の適応度を算出
     c. 最も適応度が高い表現型を最適化問題に適用
  3. 遺伝的操作によるQGRNの最適化
     A) 適応度の高いQGRNをエリート保存し、適応度の低い個体へ複製
     B) 制御行列$\theta$に対して突然変異を適用（エリート個体を除く）
  4. 2. 3.を繰り返し、適応度の高い表現型を発現するQGRNを構築

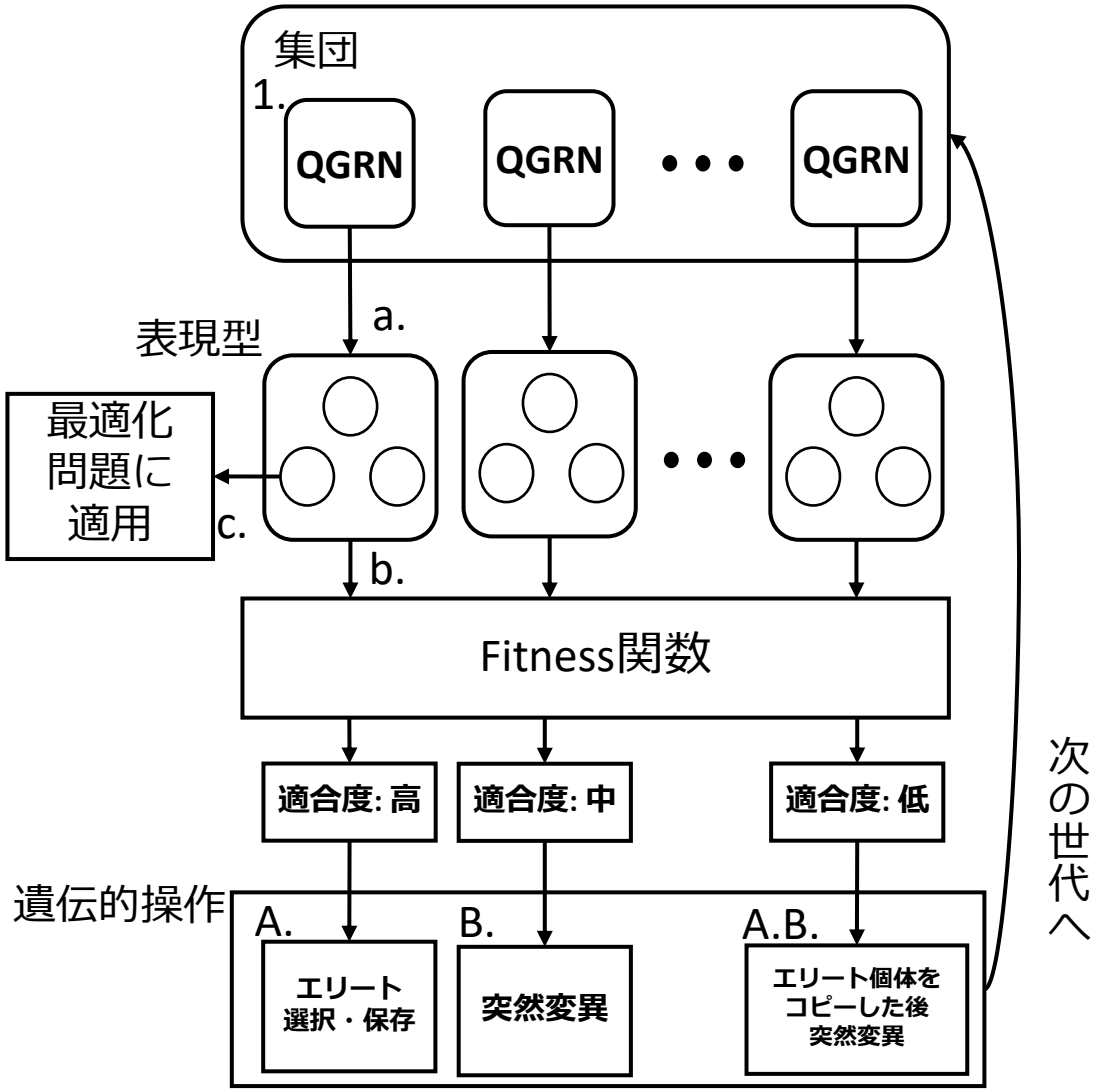


図5：最適化アルゴリズムの概略図

# 検証１：記憶性能の検証



- **概要**
  - 5つの表現型をGRN・QGRNにそれぞれ100個体に記憶させる[4]
    - 5つの表現型：スライス埋め込み問題において解候補となるもの
    - 「記憶した表現型」：1個体において、発現確率が上位6位の表現型

表1：記憶させる表現型

| | 記憶させる表現型 |
|---|---|
| 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 |

- **比較対象**
  - QGRN：提案手法の適用
    - Fitness関数：KLダイバージェンス (目的の確率分布に近づける)
      - $p_i^{target}$ ： $i$ 種類目の表現型に対して、表1の表現型の発現確率がそれぞれ20%で発現する確率分布
      - $p_i^{out}$ ：表現型の発現確率

$$Fitness = -\sum_i p_i^{target} \cdot log\left(\frac{p_i^{target}}{p_i^{out}}\right).$$

  - GRN：先行研究の手法[1]
    - Fitness関数：5つからランダムに選択した表現型とのハミング距離
      - 20世代学習後、表現型をランダムに選択しなおし、学習を10回繰り返す

[1] 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.
[4] 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.

# 検証１：記憶性能の検証結果



- **QGRNは従来のGRNと比較して記憶可能な表現型が増加**
  - QGRN：平均4.33個
  - GRN：平均1.33個
    - 既存研究の課題である複数の表現型を記憶するモデル構築の不安定性は解消
- **QGRNにおいて発現確率は低い**
  - QGRNにおいて、発現確率は5%~7.5%程
    - QGRNから記憶対象の表現型を取り出す場合、発現確率を考慮した観測回数が必要

表2：記憶させた表現型と記憶できた個体数

| | 記憶させた表現型 | QGRN | GRN |
|---|---|---|---|
| 表現型1 | 00100001111101 | 5個 / 6個 | 3個 / 6個 |
| 表現型2 | 00101010011111 | 6個 / 6個 | 1個 / 6個 |
| 表現型3 | 00001010111111 | 5個 / 6個 | 0個 / 6個 |
| 表現型4 | 00100001101001 | 4個 / 6個 | 1個 / 6個 |
| 表現型5 | 01110101110101 | 6個 / 6個 | 3個 / 6個 |
| 平均個数 | | 4.33個 / 6個 > | 1.33個 / 6個 |

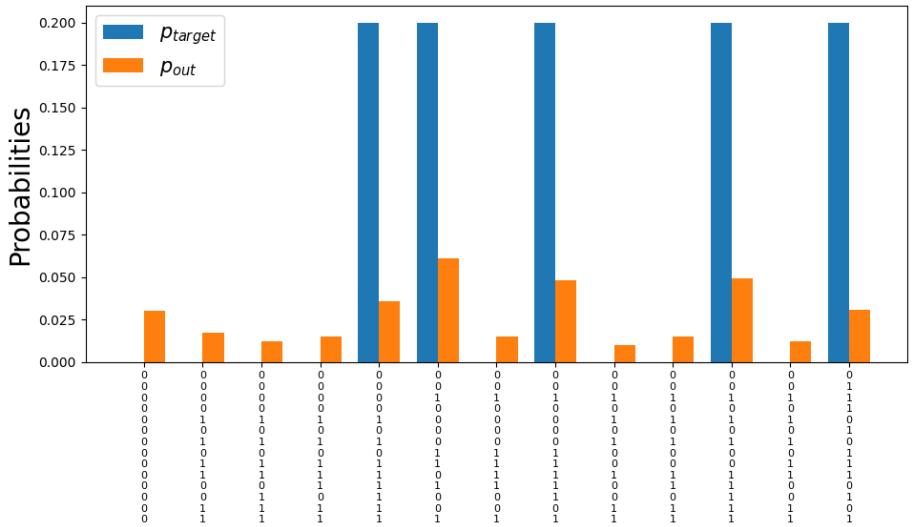


図6：QGRN最良個体の表現型の確率分布(1%以上のみ)

# 検証２：NW最適化問題への適応（定式化）



- **ネットワークの設定**
  - 物理ネットワーク
    - ユーザ端末2台、エッジサーバ2台、クラウド2台の階層構造
  - 仮想ネットワークスライス
    - 2つのスライス方法を設定
    - 仮想リンクとアプリケーションを、制約条件を基にそれぞれ物理ネットワーク上に埋め込む

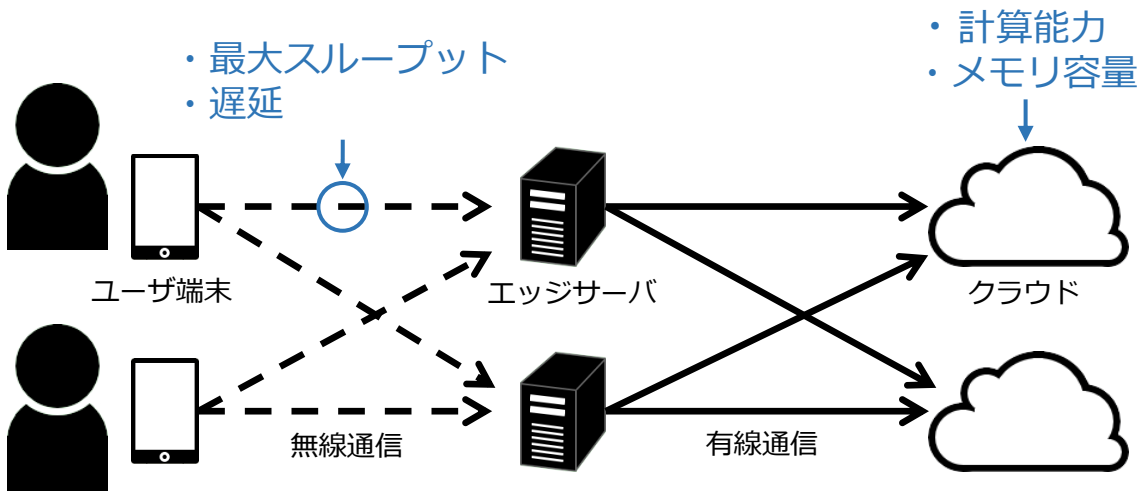


図7 : 物理ネットワークの設定

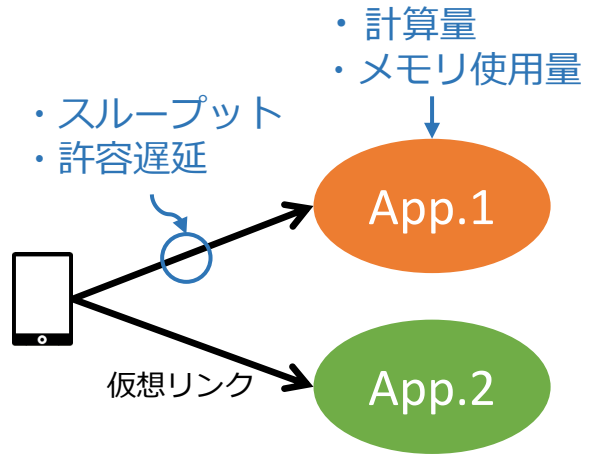


図8 : ネットワークスライス1

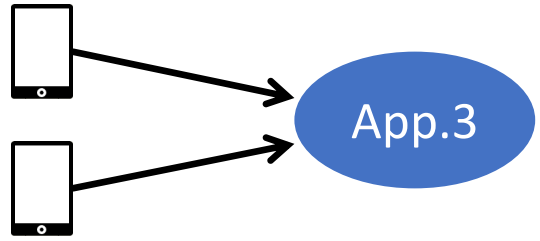


図9 : ネットワークスライス2

# 表現型と埋め込み方法の対応付け



- **表現型は14bitで表現し、スライスの埋め込み先を決定する**
  - アプリケーションの埋め込み先はエッジ・クラウドの数字に対応

[0 0　0 1　1 0　1 1　1 0　1 1　0 1]

0 0：リンク
0 1：リンク
1 0：App.1 → $c_2$へ
1 1：App.2 → $c_3$へ
1 0　1 1：リンク
0 1：App.3 → $c_1$へ

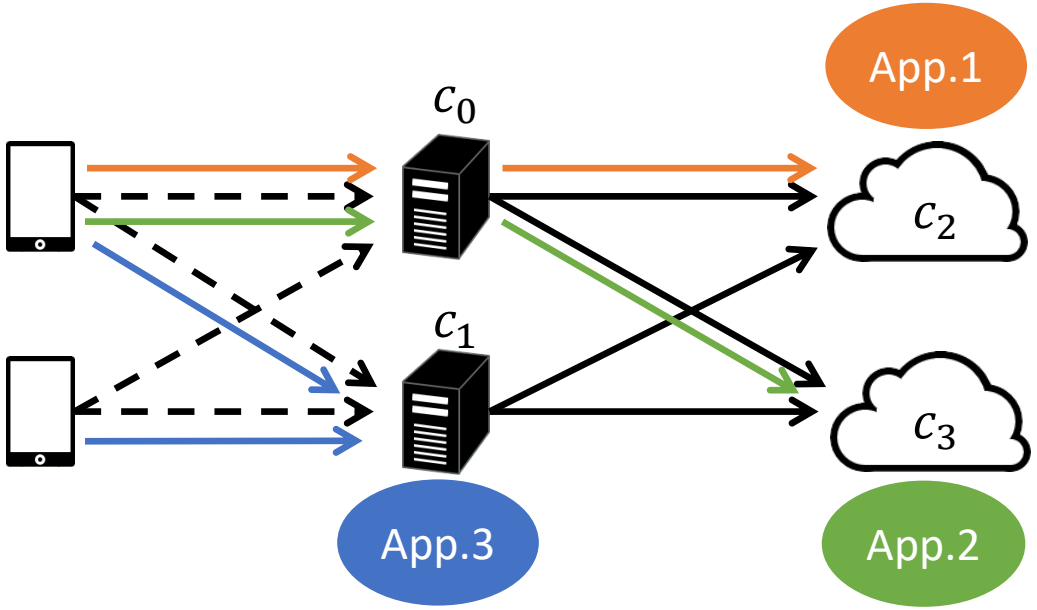


図10 : 表現型とスライス埋め込みの対応

表3 : リンクを表すビットと対応する経路

| リンクを表現する2bit | 経路 |
|---|---|
| 00 | エッジ$c_0$ → クラウド$c_2$ |
| 01 | エッジ$c_0$ → クラウド$c_3$ |
| 10 | エッジ$c_1$ → クラウド$c_2$ |
| 11 | エッジ$c_1$ → クラウド$c_3$ |

# 目的関数と制約条件



- **目的関数**
  - 目的：ネットワーク全体の消費電力が低減すること

スライスが一つでも
埋め込めない場合にペナルティ

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

物理ネットワークに埋め込んだ
仮想リンクの消費電力

使用したエッジや
クラウドの消費電力

- **制約条件**
  - 通信帯域

    埋め込まれた仮想リンクの帯域の和が、各物理リンクの最大帯域を超えない
  - 遅延

    各仮想リンクの許容遅延が、埋め込み先の物理リンクの遅延値の和を超えない
  - 計算能力

    各エッジおよびクラウドの計算能力が、
    埋め込まれた全アプリケーションの計算量の和を超えない
  - メモリ

    各エッジおよびクラウドのメモリ容量が、
    埋め込まれた全アプリケーションのメモリ使用量の和を超えない

表4：変数とその役割

| 変数 | 役割 |
|---|---|
| $\alpha$ | 埋め込み失敗時のペナルティ定数 |
| $\omega_i$ | 物理リンク$e_i$の消費電力係数 |
| $T_i^k$ | 仮想リンク$l_i^k$に必要なスループット |
| $\omega_m$ | 埋め込み先ノード$c_m$の消費電力係数 |
| $P_w^k$ | アプリケーション$a_w^k$に必要な計算量 |
| $B_m$ | アプリケーション$a_w^k$の埋め込み先ノード$c_m$のベース電力 |

# 評価環境



- **環境変動設定**
  - 環境A→環境B→環境C のサイクルを30分間隔で繰り返す
- **実行環境**
  - 量子フレームワークのQiskitと量子シミュレータのQiskit-Aerを用いて古典コンピュータ上で実行
- **比較手法**
  - GA (QGRNを用いない) とQGRNとの比較
    - QGRNのFitness関数

      発現確率上位5表現型のうち、最大の目的関数値

表5 : 実測に用いた実行環境

| 項目 | 内容 |
|---|---|
| プロセッサ名 | Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz |
| スレッド、コア数 | 64スレッド 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 |
| プログラム言語 | Python 3.11.6 |
| 主要パッケージ | Qiskit 1.3.1, Qiskit-Aer 0.15.1 |

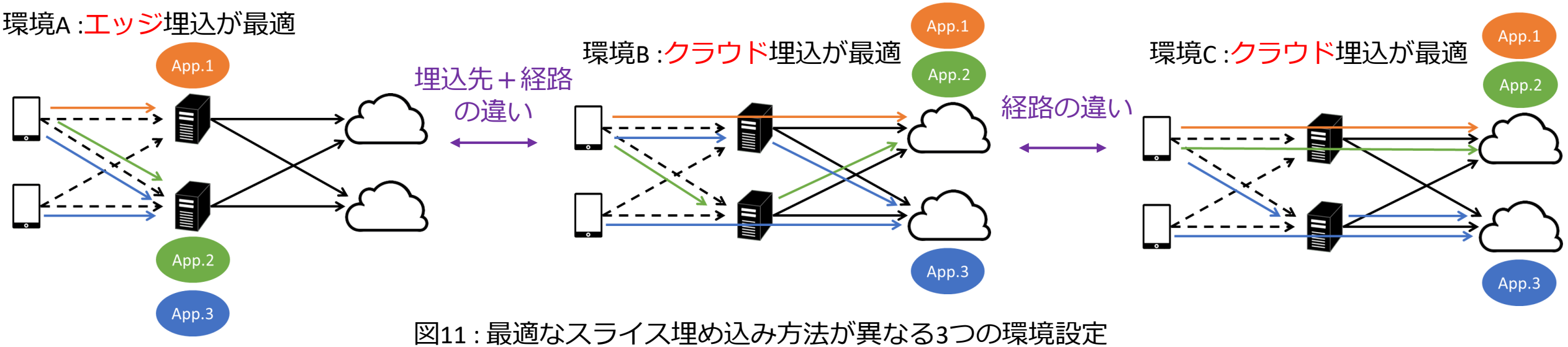


図11 : 最適なスライス埋め込み方法が異なる3つの環境設定

# 評価結果：実時間に対する最適解到達の評価



- **QGRN：短時間で最適解へ到達 ↔ GA：時間内に到達困難**
  - QGRN：少ない突然変異回数で表現型の複数ビットが同時変異、局所解脱出
  - GA：局所解 (埋め込みに成功するが最適でない解) に陥り、時間内に脱せず

表6：環境ごとの最適なFitnessの値

| 環境 | 最適解のFitness |
|---|---|
| 環境A | -2263.22 |
| 環境B | -8168.18 |
| 環境C | -6733.23 |

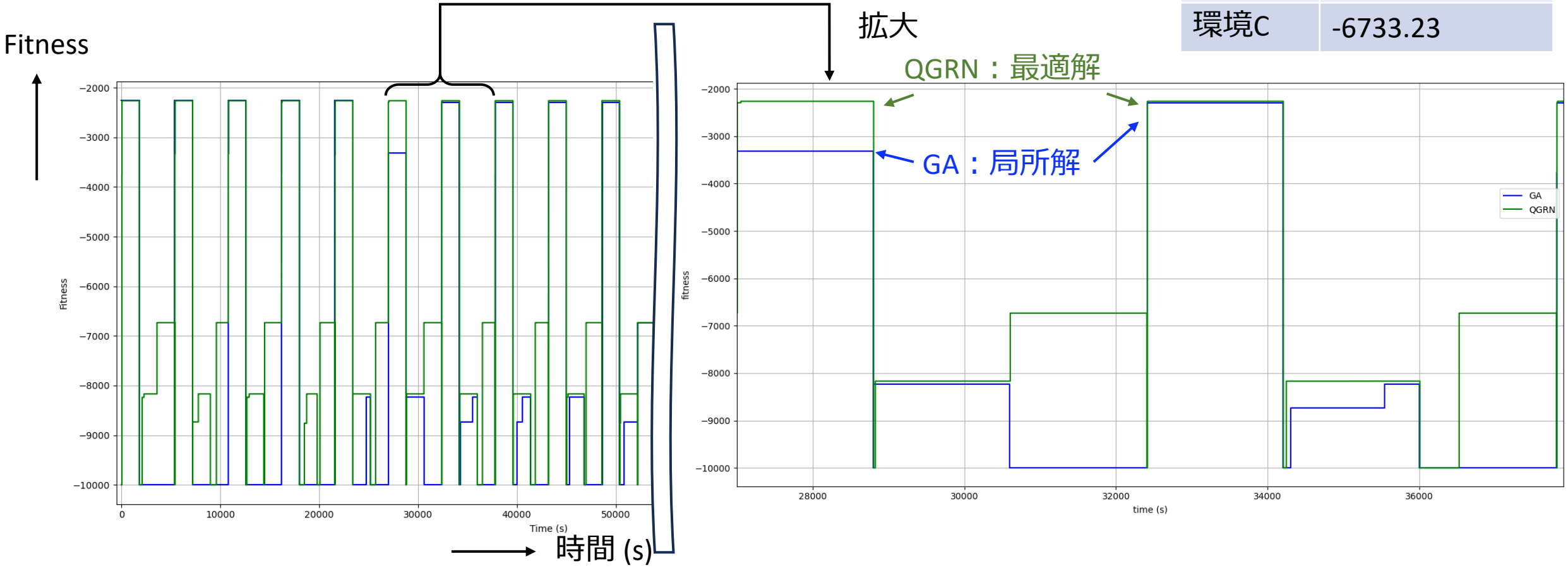


図12: 環境A,B,Cが30分ごとに変動する環境下でのfitnessの推移

# 評価結果：最適解および局所解への到達率



- **最適解到達率は、QGRN：高 ↔ GA：低**
  - 最適解到達率：最適解に到達した回数/環境変動回数
  - 一度の変異で表現型の複数ビットが同時変異するQGRNの特性により、局所解への収束を回避したため

表7：環境変動時におけるQGRNとGAの最適解到達率
(最適解に到達したケースのみ)と局所解到達率

| | 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) |
| | 局所解到達率 | 100%(48/48回) | 100%(48/48回) | 95.83%(46/48回) |
| GA | 最適解到達率 | 18.75%(9/48回) | 25%(12/48回) | 54.17%(26/48回) |
| | 局所解到達率 | 100%(48/48回) | 75%(36/48回) | 54.17%(26/48回) |

# 評価結果：環境C→Aで最適解に到達するまでの時間



- **QGRN は1分以内に最適解への到達することが多かったが、GA は局所解に陥り30分以内に最適解に到達できない事が多かった**
  - 環境C→A：最適解同士のハミング距離が５bitと大きく、表現型の複数ビットの変異が必要なため
  - QGRNは比較的広く解空間を探索可能
  - 環境A→Bでも同様の傾向

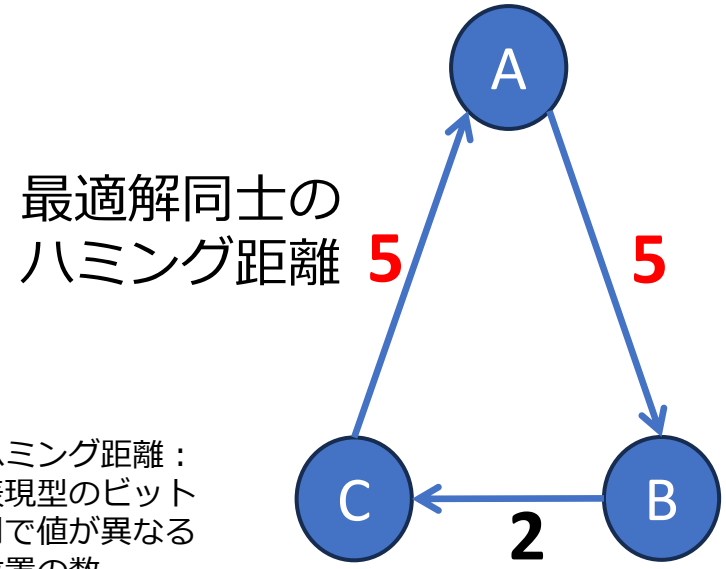


図 13 : 各環境同士の解のハミング距離の関係性

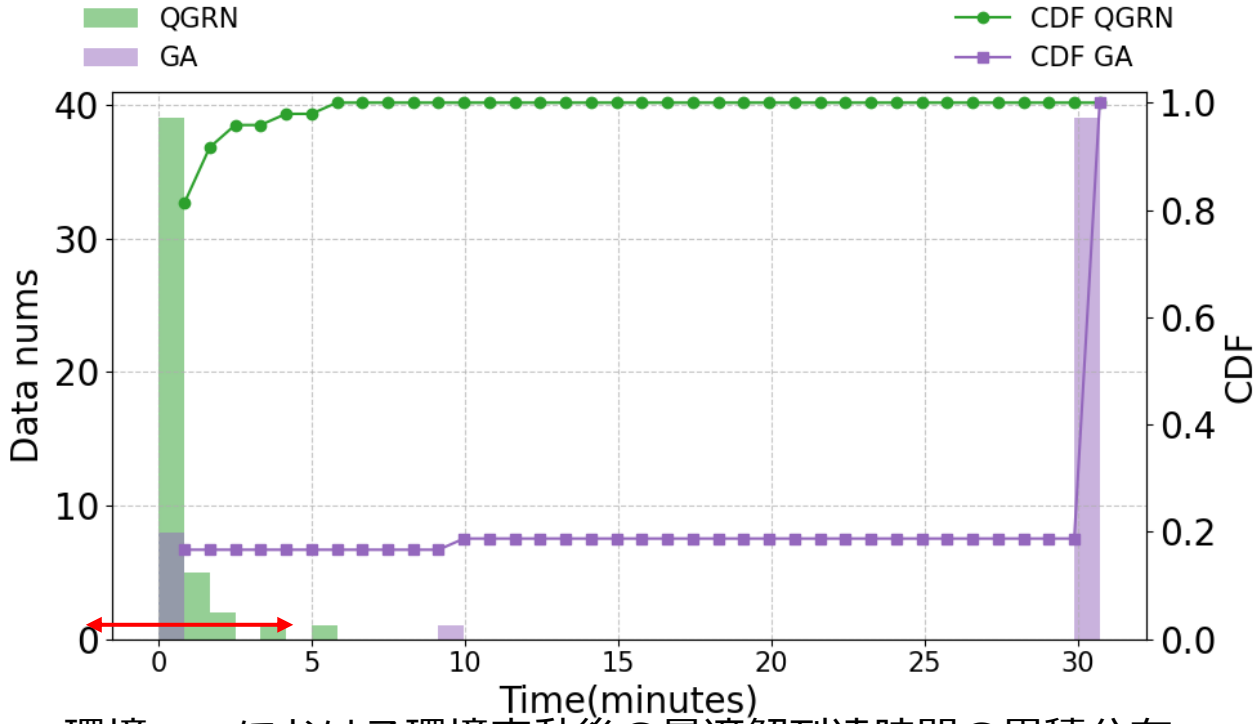


図 14 : 環境C→Aにおける環境変動後の最適解到達時間の累積分布
ただしヒストグラムの“30”は、30分間経過しても最適解に到達不能という意

# 評価結果：環境B→Cで最適解に到達するまでの時間



- **1分以内に最適解に到達したケースは、QGRNよりGAの方が多かった**
  - 環境B→C：最適解同士のハミング距離が２bitと小さく、GAは短時間で到達可能
    - 今回のQGRNは近い解同士の遷移は苦手
  - ただし、GAでは最適解に到達できないケースも多かった

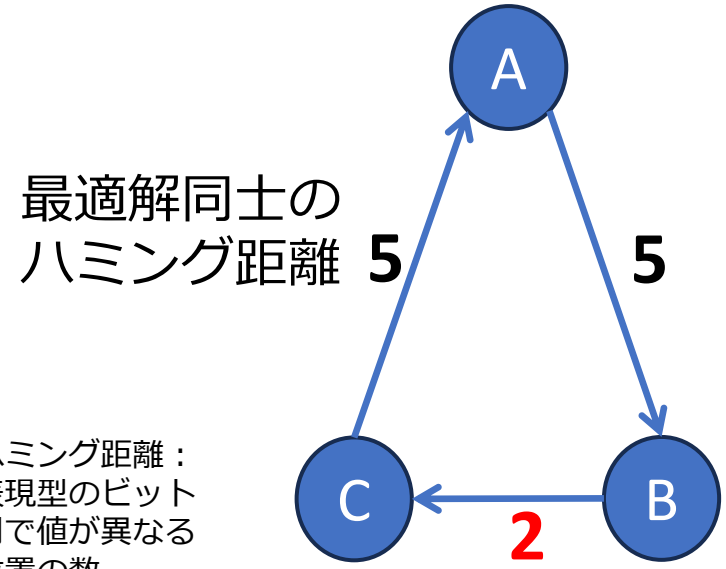


図 12 : 各環境同士の解のハミング距離の関係性

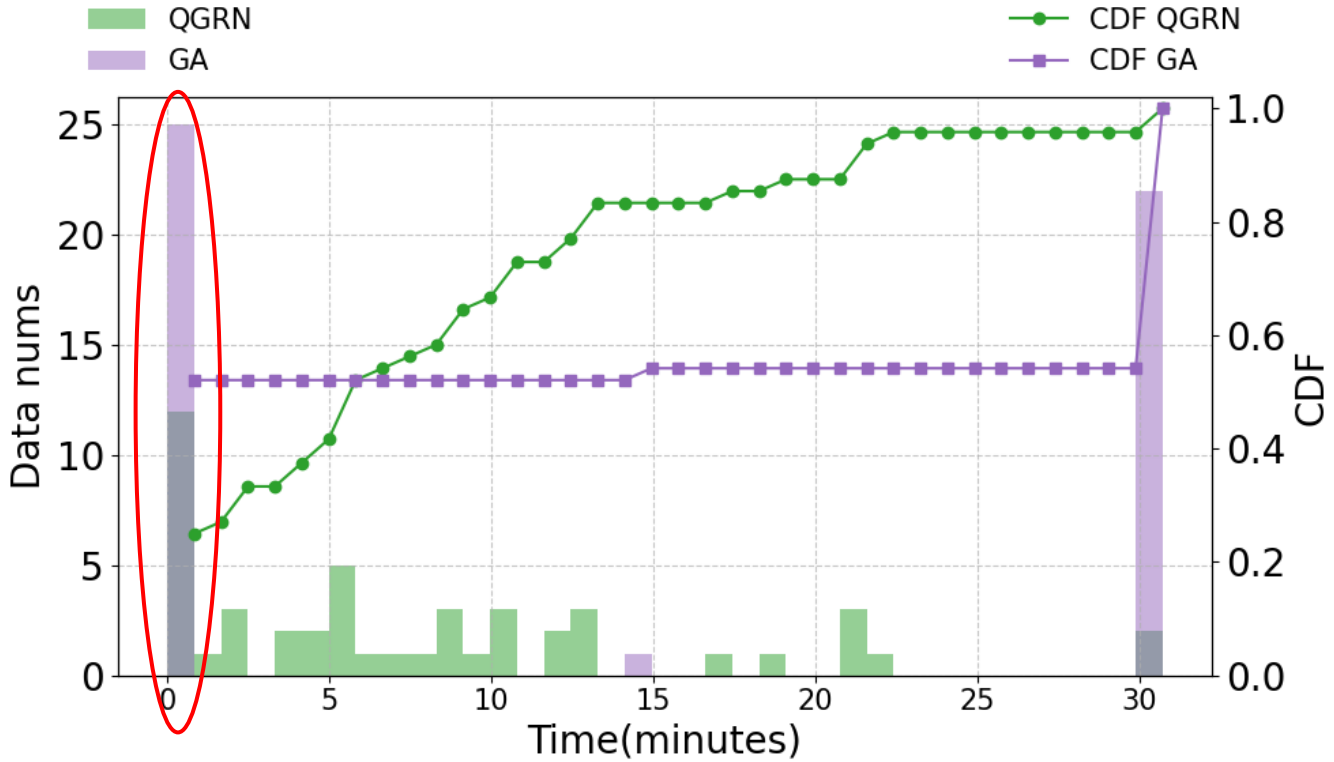


図 15 : 環境B→Cにおける環境変動後の最適解到達時間の累積分布
ただしヒストグラムの“30”は、30分間経過しても最適解に到達不能という意

# まとめと今後の課題



- **QGRNを用いてGAを拡張した最適化手法の考案**
- **QGRNモデルの表現型の記憶性能を検証**
  - 既存のGRNと比較して、記憶可能な表現型の個数が増加
  - ➢ 複数の表現型を安定的に記憶できるGRNモデルである
- **動的なNW制御問題への適用検証**
  - GAと比べて最適解への到達率が高かった
    - 遺伝子型の一つのビットの変異により表現型の複数ビットが同時に変異することで、局所解に陥らなかったため
  - ➢ 従来のGRNと同様、局所解に陥ることなく最適解に到達可能
  - ただし、環境変動のたびにQGRNの最適化をやり直すため、事前に最適解を記憶した既存手法に比べて非効率的
- **今後の課題**
  - 動的環境下で最適化を行いつつ、過去の最適解を記憶したQGRNを活用すること
    - Fitness関数やQGRNに適用する遺伝的操作の設定、エリート保存の工夫などにより対応できないか