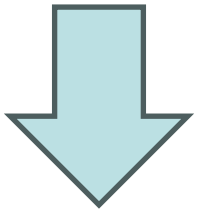
動的な需要に対する分散量子演算の強化学習を用いた割り当て法

田中　信元　　上山　憲昭
立命館大学

研究背景

- **量子計算の規模拡大と分散化**
  - 大規模量子回路の実行には、単一QPU（量子処理装置）の**規模制約がボトルネック**となる
  - 複数の小規模QPUを量子ネットワークで接続し、協調動作させる**分散量子演算（DQC）**が有望な解決策



- **DQC環境における主要課題**
  - ノード間の**エンタングルメント資源は有限**であり、生成に時間を要する
  - 非局所ゲート操作（Teleportation等）に伴う**通信遅延・忠実度劣化**が発生

問題点：従来研究の限界と課題

- 従来研究のアプローチ
  - 要求が事前に全て既知である**「静的状況」を前提**とする研究が主流
  - 数理最適化（混合整数計画法など）や固定的なルールベース手法により最適化
  - 局所的なルールに基づき、その場限りの最適化を行うのみ

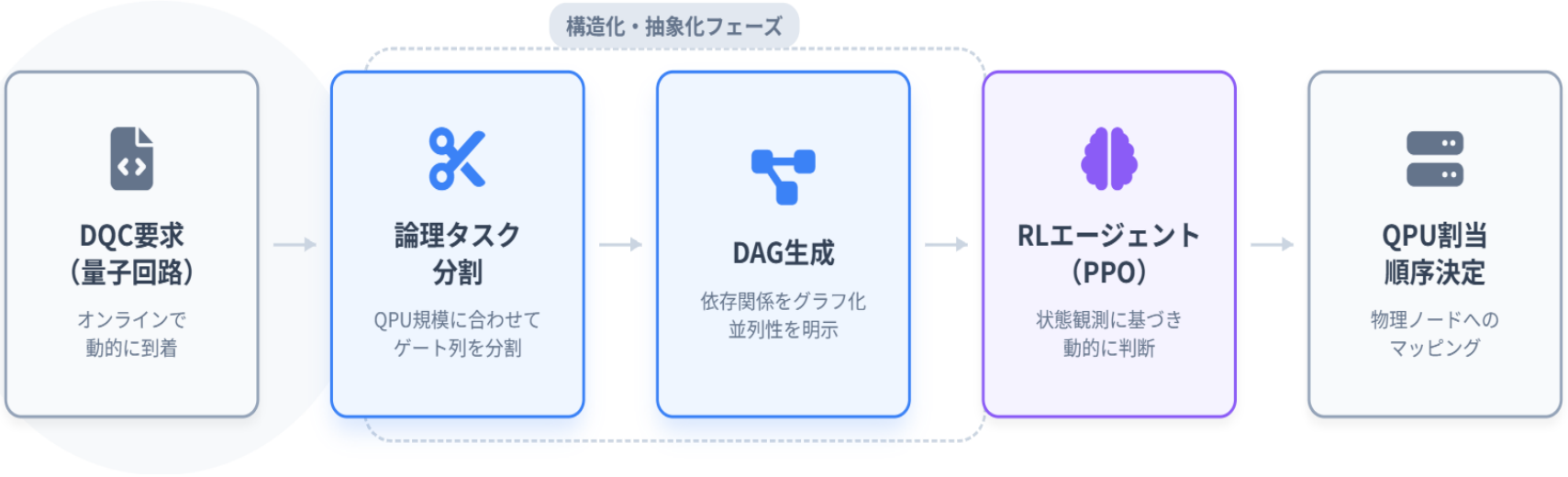
- **動的環境における問題点**
  - 要求がオンラインで到着する環境では、**静的最適化は適用困難**
  - ルールベースでは、資源の枯渇や通信遅延の変動といった**複雑な状況変化に適応できない**
  - 適応性の欠如が、結果として大幅な**性能劣化（スループット低下・忠実度悪化）**を招く

本研究の目的

- **動的DQC環境における目的**
  - DQCネットワーク全体における**平均処理時間の短縮**
  - 非局所操作に伴う**量子忠実度（Fidelity）の維持**
  - 上記2つの要件を**両立する資源割当**を実現する

- **提案手法のアプローチ**
  - 量子回路の依存関係を**有向非巡回グラフ（DAG）化**して構造化
  - **強化学習**を用いた動的かつ逐次的な資源割当
  - オンラインで到着する要求に対し、**即応性の高い最適化**を行う

提案方式の全体像



- 構造の明示化（DAG）

物理的な制約（量子ビット数など）や論理的な依存関係（ゲート順序、エンタングルメント）は、事前の構造化フェーズで**DAGとして静的に解決**する。

- 動的最適化の委譲

資源の競合解決やタイミングの調整といった複雑で動的な判断のみを、**強化学習（Reinforcement Learning）に集中**させる。

論理タスクの分割

- 連続ゲート列のグループ化

同一QPU上で実行可能な一連の量子ゲート操作をまとめて1つの論理タスクとします。通信オーバーヘッドを削減する。

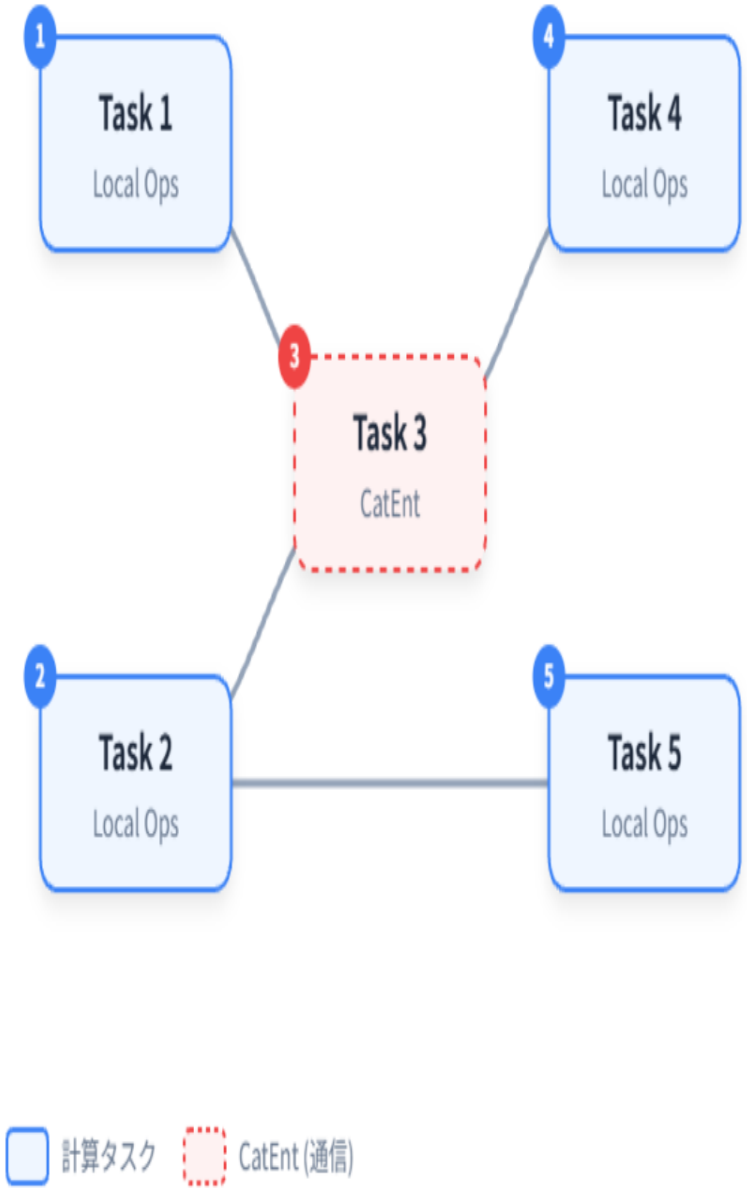
- CatEnt操作の分離

異なるQPU間のエンタングルメント生成（CatEnt）は通信遅延を伴うため、必ず単独のタスクとして切り出す。

- スケジューリング単位の抽象化

個々のゲート単位ではなくタスク単位で扱うことで、強化学習エージェントの決定空間を削減する。

DAGの生成



ノード：論理タスク

分割された各論理タスクをグラフのノードとして定義します。各ノードは実行に必要なQPU時間や量子ビット数などの情報を保持する。

エッジ：実行順序依存

同一量子ビットを共有するタスク間や、CatEnt（エンタングルメント生成）の結果を利用するタスク間に有向エッジを張る。

グラフの最適化

直接的な依存関係のみを残し、推移的な冗長エッジ（A→B→Cがある場合のA→Cなど）を削除して最小依存集合を構築します。

並列実行性の顕在化

DAG構造化により、依存関係のないタスク同士（ブランチ）の並列実行可能性を明確にし、スケジューリングの自由度を高めます。

強化学習の設計①

状態空間（State）

- 実行可能タスク集合

依存関係が解消され、実行待ち状態にある論理タスクのリスト

- QPU空き時刻

各ノードが現在処理中のタスクを完了し、利用可能になる時刻

- 処理中ジョブ情報

現在ネットワーク内で実行されているタスクの配置状況と進捗

- コヒーレンス残存時間

量子ビットの情報の寿命。これが尽きる前に処理を完了する必要がある

強化学習の設計②

行動＆制約（Action）

- タスク割り当て

行動 a = (τ , q) : タスク τ を QPU q に割り当てる

- アクションマスク（制約）

物理的に実行不可能な行動を事前に除外（マスク）し、探索効率を向上させる。

  - **実行可能タスク制約**
    - **DAG上で実行可能状態のノードのみ選択可**
  - **量子ビット数制約**
    - **QPUの量子ビット容量を超える割当は不可**
  - **忠実度制約**
    - **忠実度が閾値を下回る割当は実質不可**

報酬関数の設計

$$r_t = -\alpha \cdot T_{wait} - \beta \cdot \Delta F - \gamma \cdot P_{violation}$$

- **待ち時間ペナルティ（α）**
  - タスクの実行開始までの待機時間を最小化
  - QPUのアイドル時間を減らし、スループットを向上させる
- **忠実度劣化ペナルティ（β）**
  - デコヒーレンスや通信ゲートによる忠実度低下を抑制
  - 計算結果の信頼性を担保する
- **制約違反ペナルティ（γ）**
  - 量子ビット数上限やコヒーレンス時間切れ等の物理制約違反
  - 実行不可能な割り当てを罰する

評価環境

- **量子ネットワーク構成**
  - **QPUノード数：30台**

小規模～中規模ネットワークを想定

  - **接続トポロジー：Erdős–Rényi (ER) グラフ**

接続確率p=0.6でランダム生成

  - **各QPUの規模：15 ～ 25 qubits**

不均一な資源分布を模擬

- **ワークロードと学習設計**
  - **要求発生：ポアソン過程**

到着率λを可変パラメータとして評価

  - **学習規模：1エピソード 200要求**

合計 1,000 エピソードの学習を実施

  - ノイズモデルの導入

通信距離に応じた**遅延・忠実度劣化**を考慮

比較対象手法①　MILPを用いたQPU割り当て

- QPU割当（MILP）＋バッチ型スケジューリング
- 量子通信遅延による誤差コストの最小化
- 割当成功数の最大化

- ε 制約法で単一目的化
  - 通信コストを最小化
  - 割り当て成功率を制約として扱う
- MILPソルバで最適解算出

比較対象手法②　戦略選択型RL

タスク選択

- SJF / 最小量子ビット優先
  - スループット向上
  - 大規模ジョブが飢餓状態になる可能性
- クリティカルパス優先
  - 全体makespan短縮に有利
  - 局所最適化に偏る可能性
- 忠実度優先
  - 忠実度低下抑制
  - 負荷集中の可能性

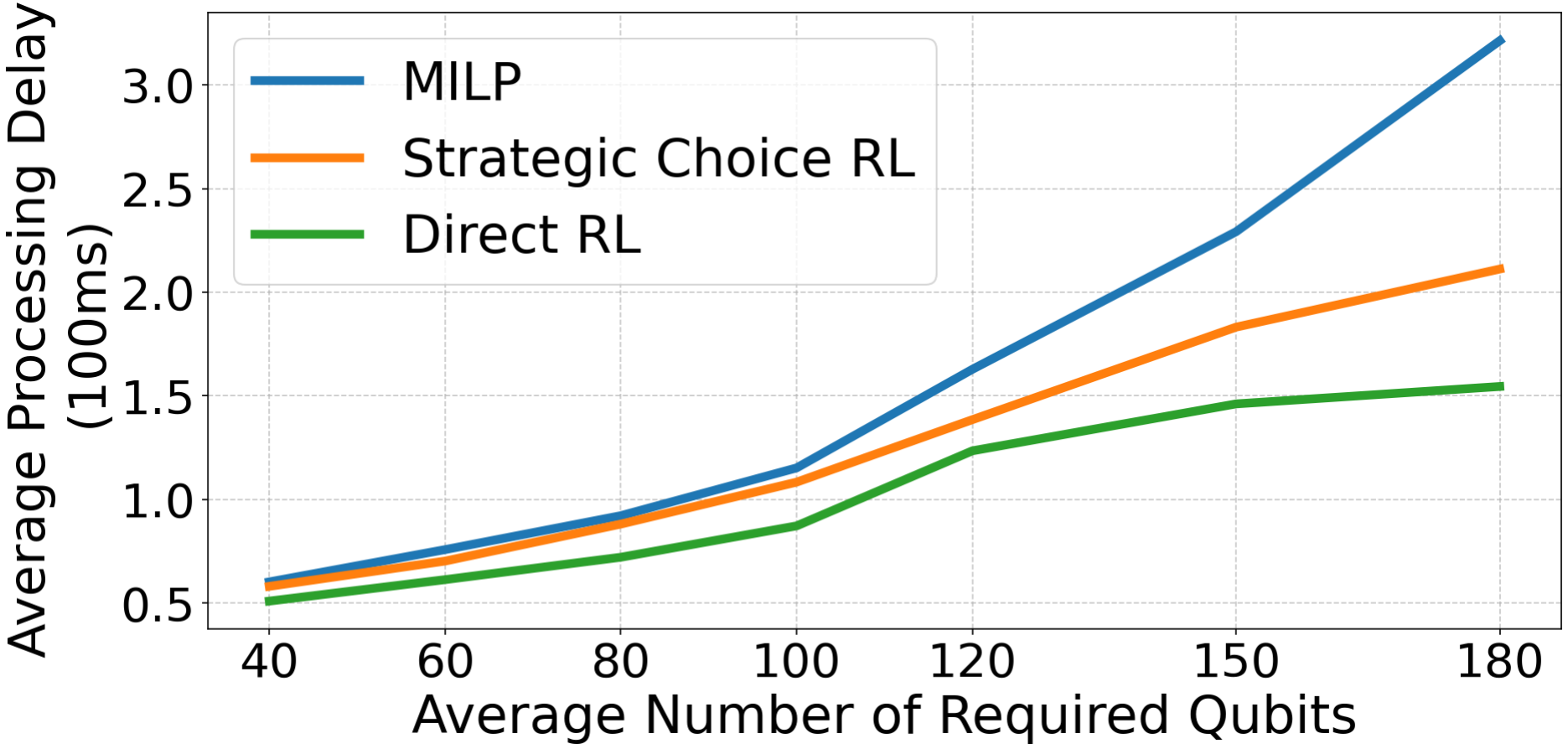
.
.
.

QPU割り当て

- 最早空きQPU割当
  - 混雑回避型
  - 忠実度悪化リスク
- 通信コスト最小QPU割当
  - CatEnt削減
  - 計算量微増
- 量子ビット余裕最大QPU割当
  - 将来拡張性高い
  - 現在待ち時間の無視

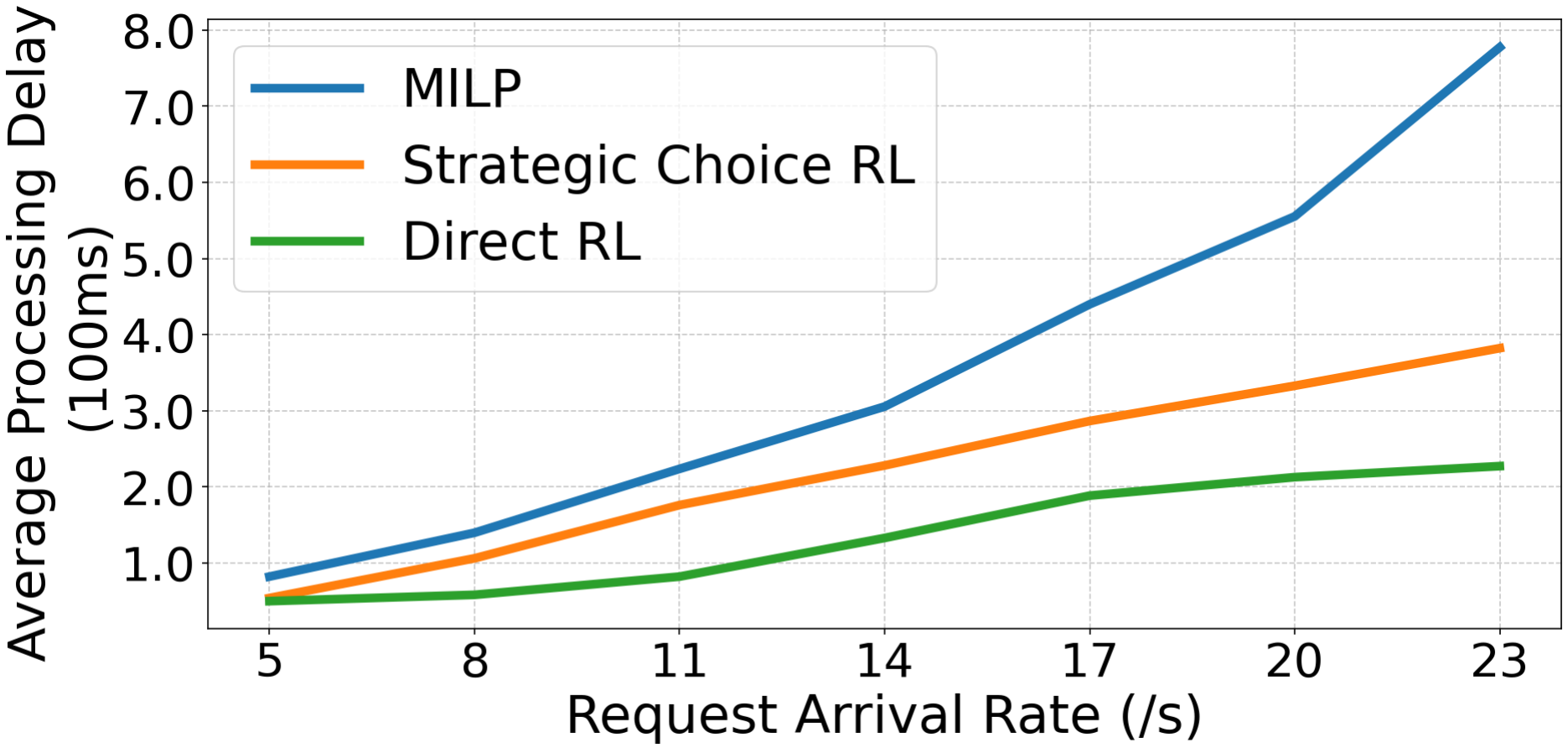
.
.
.

実験結果①：量子ビット数に対する平均処理時間



- スケーラビリティの確保
- 最適解とのギャップ
- 戦略選択型RLの課題

実験結果②：到着率に対する平均処理時間



- 高負荷環境への耐性
- 待ち時間の抑制効果
- 既存手法の限界

まとめ・今後の課題

- 本研究の成果
  - **DAG構造化と強化学習の融合**

物理制約の解決（DAG）と動的資源割当（RL）を分離・統合し、複雑なDQC環境に適応可能なアーキテクチャを確立

  - **動的環境でのスケーラビリティ**

静的最適解（MILP）より良い処理時間を達成しつつ、量子ビット数が増加しても性能劣化を最小限に抑えることに成功

  - **安全性と高速化の両立**

物理制約（量子ビット数、CatEnt資源）を報酬設計に組み込むことで、実行可能性を保証しながらクリティカルパスを短縮

- 今後の課題
  - **現実的なノイズモデルの導入**
  - **大規模ネットワークでの分散学習**