2021 年度　修士論文

# 初期状態に依存した量子回路最適化手法と超伝導量子トリットにおける制御ゲート実装手法の開発 (改訂版)
# (Development of quantum circuit optimization depending on initial states and controlled gates on superconducting qutrits)

東京大学大学院
理学系研究科 物理学専攻
澤田研究室

張 元豪

2022 年 1 月 28 日 (改訂 10 月 18 日)

# 概要

欧州合同原子核研究機構（CERN）が建設した大型ハドロン衝突型加速器（The Large Hdron Collider: LHC）では新物理の発見を促進するため、陽子陽子衝突頻度が上がる High Luminosity LHC（HL-LHC) に向けて準備が進められている。その中で予想される膨大なデータ量を処理するコンピューティングリソース不足が懸念されている。解決策の一つとして量子計算機の応用が検討されている。古典計算機では非効率な問題を効率よく解いたり、機械学習による感度を高めることが期待される。既に物理解析における飛跡の再構成、パートンシャワー、事象選択などへの応用を目指した研究が行われている。

現状の量子計算機はノイズがあり量子ビット数も限られ、NISQ (Noisy Intermediate-Scale Quantum) デバイスと呼ばれる。そこで量子計算機の特性を理解した上で量子ノイズの影響を抑えることが重要である。超伝導量子計算機である IBM Quantum では CNOT ゲートが最も大きなノイズ源となっている。特に多量子ビットゲートは CNOT ゲートを大量に消費する。以上の背景から多量子ビットゲートをより少ない CNOT 数で実装することは重要な研究テーマとなる。

本研究ではソフトウェアとミドルウェアからのアプローチにより多量子ビットゲートの効率的な実装を試みた。ソフトウェアからは初期状態に依存した量子回路最適化手法の開発を行った。制御ビットがエンタングルしている場合でも多項式計算量で一部の不要なビット制御を削除することが可能になった。また量子状態における低振幅の基底を意図的に無視した近似回路の生成も実証した。ミドルウェアからは独自のマイクロ波パルス操作によって二量子トリットゲートのパルスシークエンスの開発を行い、多量子ビットゲートを大幅に少ない CNOT 数で実装する可能性を開いた。

# 目次

# 第 1 章

# 序論

## 1.1 背景

### 1.1.1 高エネルギーと量子計算

LHC では世界最高衝突エネルギーを用いた陽子陽子衝突実験が行われている。ATLAS 実験 [1] では測定した大量の粒子の情報を用いて物理解析を行い、ヒッグス粒子の精密測定や超対称性粒子の探索が行われているが、標準模型を超える兆候は未だに得られていない。原因の一つは陽子陽子衝突に特徴的な QCD 由来の大量の背景事象である。そこでデータ量を増やし統計誤差を減らすことが必須である。2027 年から HL-LHC にアップグレードし衝突頻度を増やす予定となっている [2]。現在の計算機資源では HL-LHC によって生じるデータ処理に十分ではないと予測されており [3]、計算機科学におけるブレークスルーが期待されている。一つの可能性として量子計算 [4] があり、ショアの素因数分解 [5] やグローバーの探索アルゴリズム [6] では古典計算よりも少ない計算量で解くことができると考えられている。近年高エネルギー分野でも活発に研究され始めており、物理解析に近いものでは飛跡の再構成 [7, 8] やシミュレーション [9, 10]、背景事象と信号事象の分離 [11, 12, 13, 14] 等が研究されている。

### 1.1.2 NISQ

量子計算の恩恵を受けるためには量子計算を可能とするハードウェアが必要である。これを量子計算機と呼び活発に開発が進められている。量子情報を扱うためには量子力学に従う量子ビットとこれを操作するための手法が必須である。量子計算機はエンタングルメントや干渉の性質を持ち、多項式量のビット数で指数増加空間を表現できる。

量子計算機は様々な手法で実現されており、超伝導回路 [15]、光 [16]、イオントラップ [17] などがある。量子制御は非常に困難で様々なノイズが生じ、量子ビット数にも限りがあるため、現在の量子計算機は NISQ と呼ばれる [18]。理論的には量子誤り訂正によって任意の計算精度を達成できるが [19, 20] が、大量の量子ビットとゲートを要するため、未だ限定的な誤り訂正のみが実証されている [21]。そのため NISQ を用いた研究が盛んに進められている [22]。

量子計算関連の研究には量子アルゴリズムの考案や量子計算機の改善研究が主なものとして存在するが、量子アルゴリズムを既存の量子計算機においてより効率良く計算するための手法を開発するミドル

ウェア研究が存在する。代表的なものとして、測定エラー緩和 [23] や CNOT のエラー緩和 [24] や、量子回路を特定の実機で実行できる形にするトランスパイルがある [25, 26, 27]。トランスパイルには様々な要素が含まれ、実機で実現できる基本ゲートへの分解、限られた量子ビット結合という制限を満たすためのスワップ操作の追加、より少ないゲートで等価な量子計算をするための量子回路最適化などがある。量子回路をパラメータ付きアンザッツの組み合わせで表現し、元の回路を良く近似するパラメータを機械学習によって探索する手法もある [28, 29, 30, 31]。元の回路の等価生を完全に保つことは難しいが少ないゲート数で近似できるので、回路の等価性の損失よりもノイズの影響を抑えるメリットの方が大きい。このような手法は NISQ 時代では非常に有効な手法であるが、本研究では解析的な量子回路最適化手法について考える。

最近は IBM などのベンダーによるソフトウェア開発により、ユーザー側でマイクロ波パルスの制御のパラメータを自由に設定し、独自の基本ゲートを作ることができるようになっている [32, 33, 34, 35]。量子回路最適化は定められた基本ゲート群の制限下で行われるため、量子回路レベルでの改善には限界がある。そのため新しい基本ゲートの開発はそれ自身でも多くのメリットがあるが、基本ゲートの追加によってさらなる量子回路最適化を促すという効果もある。

特に IBM Quantum については単一量子ビットゲートのエラー率が $10^{-4}$、二量子ビットゲートである CNOT のエラー率が $10^{-2}$ であり、NISQ では CNOT が主なエラー源となっている [36]。特にトフォリゲートのような多量子ビットゲートは大量の CNOT を要する。よってミドルウェア研究の主な目標は、CNOT 実行数を抑えることにある。CNOT 数を減らすアプローチの例を 3 つ紹介する。一つ目は量子回路レベルで不要な制御ゲートやビット制御を削除することで、基本ゲートに分解後の CNOT 数を大幅に減らす [37, 38, 39] ことである。二つ目は量子トリットを用いた新しい分解方法によって CNOT を減らす [40, 41, 42, 43, 44, 45, 46, 47, 33] ことである。三つ目は様々な二量子ビットゲートを直接実装することである [35, 34, 33]。これらは計算コストを削減させるため、NISQ の性能を最大限に引き出すだけでなく、量子誤り訂正の高速化も狙える重要な分野である。1 つ目の背景は 1.1.3 節で、2 つ目と 3 つ目は 1.1.4 節でも述べる。

### 1.1.3　量子回路最適化

人間が設計する量子回路は大抵の場合非効率であるが、愚直に効率的な量子回路を見つけることは総当たり探索であることから現実的ではない。そこで多項式計算量の回路最適化手法が必要である。既に様々な最適化手法が提案されており、それらを組み合わせたモジュール Qiskit Transpiler [25] や t|ket> [27] などが開発されている。

ここでは不要な制御ゲートやビット制御を消す手法に注目する。量子回路のユニタリ行列の等価性を保ちながら最適化する手法としては可逆回路合成の分野がある [48]。トフォリからビット制御が消失するエラーと冗長なビット制御が加わるエラーを総称して crosspoint fault と呼ぶ。$n$ 個の制御ビットが取りうる $2^n$ 個の全基底に対して、元々のトフォリと crosspoint fault を意図的に導入したトフォリが全く同じ出力をした場合は等価性を保っていることになる。もし意図的に導入した crosspoint fault が全てビット制御消失エラーであった場合は、不要なビット制御であるということになるので消去することができる。Crosspoint fault がビット制御の消失や冗長なビット制御が同数である場合でも、ビット制御数は

減らなくても別の形の回路にすることで、他の最適化手法によってさらにシンプルになる可能性が生じる [49, 50]。また繰り返し用いられるビット制御の情報を作業ビットを保存しておくことで、後続する制御ゲートのビット制御を削除する手法も存在する [51]。

しかし量子回路を実機で実行する際は必ず初期状態 $|0\rangle^{\otimes n}$ があるため、たった一つの出力状態のみしか得られない。つまり必ずしも量子回路の等価性を保つ必要はなく、出力のみが等しくなる量子回路に変換してしまっても良い。このように初期状態依存の解析的な最適化手法としては ZX-calculus のうち Copy rule [37] と Relaxed Peephole Optimization [39] がある。どちらの手法も制御ゲート直前の量子状態を確認し指定の条件を満たす場合にのみ最適化を行うが、制御ビットが他の量子ビットとエンタングルしていると多項式時間で不要な制御操作を削除できない。そこで本研究では、多項式計算量で制御ゲート直前の量子状態に関してより多くの情報を得る手法を開発し、最適化条件を拡張することに成功した。

### 1.1.4　超伝導量子ビットへのマイクロ波パルス制御

超伝導量子コンピュータでは、量子ゲート操作はトランズモンへのマイクロ波パルスの照射によって行われる [15]。パルスのパラメータを自由に設定すれば独自の基本ゲートを作ることが可能となる [32, 52]。

ここではトランズモンを量子トリットとして扱う手法について紹介する。パルスの振動数をトランズモンの第一励起状態と第二励起状態間 $|1\rangle - |2\rangle$ の振動数に設定することで $|2\rangle$ も操作可能となり、トランズモンを量子トリットとして扱うことも可能となる [33, 42, 53, 54]。エネルギー高準位を利用することは量子ビットの自然放出時間 $T_1$ や脱位相時間 $T_2$ が短くなるという実験的な制約があるが [55]、IBM Quantum を用いた量子トリットにおけるトフォリゲート実装が行われた [42]。しかし第二励起状態における電荷分散と交差共鳴 (Cross Resonance: CR) における AC-Stark shift 由来の位相エラーが原因で、ゲート忠実度に殆ど改善が見られなかった。さらにこの先行研究では二量子トリットゲートの実装に CNOT2 回分のコストがかかっている。より高精度なトフォリゲートを実装するためには、二量子トリットゲートの実装コスト削減と上記のエラーに剛健なパルスシークエンスの開発が必須である。またトランズモンを用いた量子トリットの挙動は未だ未知な部分が多く、これらの理解も進めて行く必要がある。

## 1.2　目的

素粒子実験の物理解析において量子計算機を応用と量子誤り訂正を加速するため、本研究では CNOT 数を減らすことを目的に、初期状態に依存した量子回路最適化手法とトランズモンにおける量子トリット上のトフォリゲートの開発を行う。

量子回路最適化では、任意の制御ゲートに対する不要な制御操作削除条件を導出する。そのうち多項式計算量で可能かつ先行研究の最適化条件の十分条件を絞り出し、これに従った最適化プロトコルを AQCEL と呼び Qiskit を用いて実装する。AQCEL は最適化過程の中で量子計算機を用いた測定を要求するため、NISQ を測定に用いても AQCEL が実際に有効であること、低振幅基底を棄却することで効率の良い近似回路を得ることができることを実証をするために、パートンシャワー量子アルゴリズムをに適用する。また IBM Quantum を量子トリットとして扱い、自ら考案した現実的なトフォリ分解と二量子トリットゲー

トを実装する。第二励起状態への漏れ (leakage) と、相対位相を無視した Z 基底測定による古典忠実度を測定することで量子トリットの挙動を調べ、今後の発展の方向性を議論する。

## 1.3　構成

第 2 章では [4, 15, 56] を参考に量子回路と超伝導量子コンピュータとマイクロ波パルスによる制御手法、量子エラーについて説明する。第 3 章では本研究で開発した量子回路最適化手法 AQCEL とパートンシャワー量子アルゴリズム [9] に適用した結果を紹介する。なおこの成果は [57] で公表しており、本節もこの論文を参考にしている。第 4 章では量子トリットおける様々なトフォリゲート分解とそれを実現するために必要なゲート開発と、完成した新しいトフォリゲートを紹介する。なおこの成果の一部は [43, 58] で公表しており、本節もこの論文を一部参考にしている。第 5 章で本研究の結論と今後の展望を述べる。また AQCEL の内、もう一つの構成要素である量子回路繰り返しゲート群の識別を付録 A で紹介する。

# 第 2 章

# 量子計算

## 2.1 量子回路

量子アルゴリズムを実機で実行するためにはまず共通言語で表現する必要があり、このうち最も高水準言語が量子回路である。量子回路は量子ビット、量子ゲート、測定、初期化からなる。ハードウェアの実現方法は 1.1.2 節で紹介したように様々あるが、全て同じゲート型量子計算機であり、その計算方法はどれも量子回路で表現可能である。なお本節では量子情報単位が量子ビットか量子トリットによって、重要な量子ゲートや multi-controlled gate の分解方法、相対位相などに違いがあるためどちらも紹介する。

### 2.1.1 量子情報単位

量子情報単位は、用いられる計算基底数によって呼び方が異なる。量子ビット (Qubit) は $|0\rangle$ と $|1\rangle$ の 2 つ、量子トリット (Qutrit) はこれらに $|2\rangle$ を加えた 3 つの計算基底を用いる。なお $d$ 個の計算基底を用いる場合は $d$-量子ディット ($d$-Qudit) と呼ぶ。実際には扱いやすさと、 2.2.1 節で説明するようなエラーの存在から、量子ビットが使われることが殆どである。

量子ビットは、二つの正規直交基底を計算基底として扱い、これらの複素数の係数を用いた線型結合もしくは重ね合わせによる 2 次元複素ベクトル空間で表現される。

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{2.1}$$

ただし $|\alpha|^2 + |\beta|^2 = 1$ であり、これを正規化条件という。つまり量子ビットにおける量子状態は、図 2.1 にあるように 2 次元複素ベクトル空間における半径 1 のブロッホ球上の点で表される。もしくは中心から長さ 1 のベクトルである。なおブロッホ球の点を極座標で

$$|\psi\rangle = e^{i\gamma}(\cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle) \tag{2.2}$$

というように表すこともできる。$e^{i\gamma}$ は絶対位相と呼び、$\gamma$ の値によって量子状態は区別されないため、物理的には全く意味のないパラメータである。$e^{i\phi}$ は相対位相と呼び、$\phi$ の値に応じて測定基底によっては異なった測定が得られたりするため、物理的に意味のあるパラメータである。そのため絶対位相は省略される。

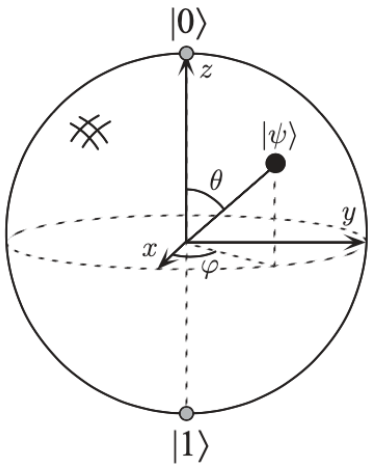



図 2.1　量子ビットのブロッホ球による表現 [4]。

同様に量子トリットも

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle + \gamma\,|2\rangle \tag{2.3}$$

ただし $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$ である。極座標で表すと

$$|\psi\rangle = \sin\frac{\eta}{2}\cos\frac{\theta}{2}\,|0\rangle + e^{i\phi_1}\sin\frac{\eta}{2}\sin\frac{\theta}{2}\,|1\rangle + e^{i\phi_2}\cos\frac{\eta}{2}\,|2\rangle \tag{2.4}$$

となり、相対位相が 2 つになっていることがわかる [59]。

### 2.1.2　量子ゲート

量子ビットは量子ゲートを作用させることで量子状態を操作する。量子ゲートが作用した後の量子ビットも正規化条件を満たす必要があるため、量子ゲートは等長変換である。複素ベクトル空間における等長変換はユニタリ変換であるので、量子ゲートはユニタリ演算子である。ゲート作用は行列で表現されることが多いので、まず量子ビットの行列表示を行う。

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix}, |1\rangle = \begin{bmatrix}0\\1\end{bmatrix}, \tag{2.5}$$

ここで量子ビットゲートの生成子はパウリ行列（$I$、$\sigma_x$、$\sigma_y$、$\sigma_z$）であり、

$$I = |0\rangle\langle 0| + |1\rangle\langle 1| = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} \tag{2.6}$$

$$\sigma_x = |0\rangle\langle 1| + |1\rangle\langle 0| = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} \tag{2.7}$$

$$\sigma_y = -i\,|0\rangle\langle 1| + i\,|1\rangle\langle 0| = \begin{bmatrix}0 & -i\\i & 0\end{bmatrix} \tag{2.8}$$

$$\sigma_z = |0\rangle\langle 0| - |1\rangle\langle 1| = \begin{bmatrix}1 & 0\\0 & -1\end{bmatrix} \tag{2.9}$$

図 2.1 に表されるブロッホ球上の x、y、z 軸周りの回転ゲートを $Rx$、$Ry$、$Rz$ ゲートと呼び、パウリ行列を用いて

$$Rx(\theta) = e^{-i\sigma_x \frac{\theta}{2}} = \cos\frac{\theta}{2} I - i\sin\frac{\theta}{2}\sigma_x = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \tag{2.10}$$

$$Ry(\theta) = e^{-i\sigma_y \frac{\theta}{2}} = \cos\frac{\theta}{2} I - i\sin\frac{\theta}{2}\sigma_y = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \tag{2.11}$$

$$Rz(\theta) = e^{-i\sigma_z \frac{\theta}{2}} = \cos\frac{\theta}{2} I - i\sin\frac{\theta}{2}\sigma_z = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix} \tag{2.12}$$

と定義される。$Rz$ ゲートの最後の等式では絶対位相を除いた。例えば $\theta = \pi$ である時に $Rx$ ゲートは

$$Rx(\pi) = e^{-i\sigma_x \frac{\pi}{2}} = -i\sigma_x = \begin{bmatrix} 0 & -i \\ -i & 0 \end{bmatrix} \tag{2.13}$$

であり、$-i$ の部分は絶対位相に当たるからこれは $X$ ゲートである。$Y$、$Z$ ゲートについても同様である。$X$ ゲートは $|0\rangle$ と $|1\rangle$ を反転させる作用をし、$Z$ ゲートは $|0\rangle$ に対しては作用せず $|1\rangle$ に $\pi$ 回転分の相対位相シフトを与える。他にはアダマール（H）ゲート

$$H = \frac{X+Z}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.14}$$

などがある。なお任意の単一量子ビットゲートは $Rx$、$Ry$、$Rz$ から任意の 2 つのゲートを選び、それらの組み合わせで表現することが可能である [4]。

次に代表的な二量子ビットゲートである CNOT（$CX$）は

$$CX = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.15}$$

と表される。CNOT は制御ビットの状態が $|0\rangle$ である時は作用せず、$|1\rangle$ の時のみ標的ビットに対して $X$ ゲートを作用させる。なお任意の二量子ビットゲートは二つの CNOT と複数の単一量子ビットゲートで表現することができる [4]。

CNOT はビット制御数が 1 であったが、2 以上のビット制御数を持つ制御ゲートを多量子ビットゲートと呼ぶ。その中でもトフォリゲート（$CCX$）は多用され、

$$CCX = (|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 10|) \otimes I + |11\rangle\langle 11| \otimes X \tag{2.16}$$

と表される。トフォリは制御ビットの状態が $|11\rangle$ の時のみ標的ビットに対して $X$ ゲートを作用させる。トフォリは CNOT と単一量子ビットゲートのみから構成することが可能である。ビット制御数が 3 以上の制御ゲートについては全て、多項式個数のトフォリと二量子ビットゲートのみを用いて分解することが可能である [60, 61, 62]。ただし多量子ビットゲートの分解方法は非常に多くあり、特に分解する時に補助として用いられる作業ビットの数や作業ビットの状態に応じて異なる。以上より任意の量子ビットゲー

トゲートは、例えば $Rx$、$Rz$ と CNOT があれば全て構成できることが保証される。実際にこれらの量子ビットゲートが超伝導量子コンピュータで実装可能であることを 2.2 章で示す。

ここから量子トリットゲートについて紹介する。量子トリットゲートは $|0\rangle$、$|1\rangle$、$|2\rangle$ の 3 つ全ての計算基底状態に作用させる必要はなく、このうち 2 つの基底間にのみ作用するゲートのみで任意の単一量子トリットゲートを構成することが可能である [63]。量子トリットゲートではどの 2 つの計算基底間のゲートかを見分けるために $U_{kl}$ と書き、$U$ はユニタリ行列で $k,l$ は計算基底を指し、$k,l=0,1,2(k\neq l)$ である。例えば $|0\rangle$、$|1\rangle$、$|2\rangle$ のうち 2 つの基底間にのみ作用する任意の単一量子トリットゲートは、$X_{12}$ ゲートと単一量子ビットゲートで実現可能である。これは $|0\rangle$、$|1\rangle$ 間で保証されている任意の単一量子ビットゲート操作を利用している。$|0\rangle$、$|2\rangle$ 間もしくは $|1\rangle$、$|2\rangle$ 間の操作がしたい場合はまずそれらが $|0\rangle$、$|1\rangle$ になるよう基底変換する。$|1\rangle$、$|2\rangle$ の基底変換は $X_{12}$ ゲートを用いる。$|0\rangle$、$|2\rangle$ の基底変換は $X_{01}$ と $X_{12}$ を用いて式 (2.18) によって実現できる。

$$\begin{bmatrix}|0\rangle & |2\rangle & |1\rangle\end{bmatrix} = X_{12}\begin{bmatrix}|0\rangle & |1\rangle & |2\rangle\end{bmatrix} \tag{2.17}$$

$$\begin{bmatrix}|2\rangle & |1\rangle & |0\rangle\end{bmatrix} = X_{01}X_{12}X_{01}\begin{bmatrix}|0\rangle & |1\rangle & |2\rangle\end{bmatrix} \tag{2.18}$$

以上より、任意の量子トリットゲートは $|0\rangle-|1\rangle$ 空間における量子ビットゲートと $X_{12}$ ゲートで実現可能である。さらに任意の二量子トリットゲートも二量子トリットゲートである CNOT と単一量子トリットゲートから構成可能である [64]。

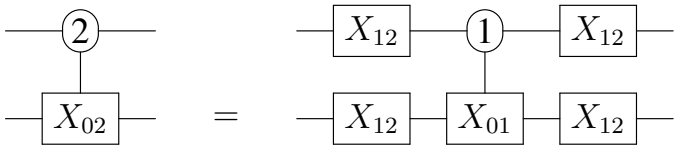



図 2.2　$C_2X_{12}$ の二量子トリットゲートである CNOT と単一量子トリットゲートを用いた分解方法。

量子トリットとして扱う場合も正規化条件が成り立つので量子トリットゲートもユニタリ演算子である。ここでまず量子トリットの行列表示を行う。

$$|0\rangle=\begin{bmatrix}1\\0\\0\end{bmatrix},|1\rangle=\begin{bmatrix}0\\1\\0\end{bmatrix},|2\rangle=\begin{bmatrix}0\\0\\1\end{bmatrix}, \tag{2.19}$$

ここで量子トリットにおける生成子はゲルマン行列が用いられ、その一部を紹介すると

$$I=\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix},\lambda_1=\begin{bmatrix}0&1&0\\1&0&0\\0&0&0\end{bmatrix},\lambda_3=\begin{bmatrix}1&0&0\\0&-1&0\\0&0&0\end{bmatrix} \tag{2.20}$$

$$\lambda_6=\begin{bmatrix}0&0&0\\0&0&1\\0&1&0\end{bmatrix},\lambda_8=\frac{1}{\sqrt{3}}\begin{bmatrix}1&0&0\\0&1&0\\0&0&-2\end{bmatrix} \tag{2.21}$$

例として式 (2.10) にある $R_x$ ゲートは量子トリット表記では

$$Rx_{01}(\theta)=e^{-i\lambda_1\frac{\theta}{2}}=\begin{bmatrix}0&0&0\\0&0&0\\0&0&1\end{bmatrix}+\cos\frac{\theta}{2}I-i\sin\frac{\theta}{2}\lambda_1=\begin{bmatrix}\cos\frac{\theta}{2}&-i\sin\frac{\theta}{2}&0\\-i\sin\frac{\theta}{2}&\cos\frac{\theta}{2}&0\\0&0&1\end{bmatrix} \tag{2.22}$$

となる。例えば $\theta = \pi$ の時、量子ビットでは行列の成分 $-i$ は絶対位相としてキャンセルすることができたが、量子トリットでは残ってしまうことがわかる。これはベリー位相と呼ばれ、実験でも確認することができるため量子トリットでは考慮しなければならない。

### 2.1.3　測定と初期化

量子回路の構成要素として最後に測定と初期化がある。測定は量子ビットの情報を取り出すが、一回の測定でビット列が一つ得られるだけである。量子ビットの量子状態が式 (2.1) で表される場合、Z 基底で測定した時に得られるビット列は 0 か 1 であり、それぞれが得られる確率は $|\alpha|^2$ と $|\beta|^2$ である。$n$ 個の量子ビットを同時に測定する場合は、測定されうる基底数は $2^n$ に増加し、十分な回数測定を繰り返すと各基底の振幅の二乗に当たる確率分布が得られる。しかしあくまで Z 基底で測定した確率分布は量子ビットの基底とその振幅を二乗した確率分布のみであり、相対位相の情報は得られていない。相対位相の情報も測定するためには 2.3.2 節で見るように量子状態トモグラフィを行う必要がある。

量子ビットの初期状態は $|0\rangle$ とする。測定が終わればその量子ビットを初期状態に戻す必要がある。これを量子ビットの初期化と呼ぶが、あくまで実機で行うべき操作であるため、量子回路では省略される。ただし測定と初期化は非常に時間のかかる操作であるため、量子誤り訂正などでは律速段階になっている [43]。

## 2.2　超伝導量子コンピュータ

この章では超伝導量子コンピュータとマイクロ波パルスによる操作の基本について [15, 56] を参考にしながら説明する。

### 2.2.1　トランズモン

初めにインダクタとキャパシタのみから構成される単純な LC 回路について考える [15]。$L$ をインダクタンス、$C$ をキャパシタンス、$Q$ を電荷、$\Phi$ を流束とする。LC 回路の古典力学におけるハミルトニアンをブランチフラックス法によって求め、これを量子化すると

$$\hat{H}_{\mathrm{LC}} = \frac{\hat{Q}^2}{2C} + \frac{\hat{\Phi}^2}{2L} \tag{2.23}$$

ここで $h$ はプランク定数、$e$ は電荷素量、$\Phi_0 = h/2e$ とし、換算電荷 $\hat{n} = \hat{Q}/2e$、換算流束 $\hat{\phi} = 2\pi\hat{\Phi}/\Phi_0$ を定義すると

$$\hat{H}_{\mathrm{LC}} = 4E_{\mathrm{c}}\hat{n}^2 + \frac{1}{2}E_{\mathrm{L}}\hat{\phi}^2 \tag{2.24}$$

ただし $E_{\mathrm{c}} = e^2/2C$ は電荷エネルギー、$E_{\mathrm{L}} = (\Phi_0/2\pi)^2/L$ は誘導エネルギーとして定義される。これは明らかに 1 次元の量子調和子のハミルトニアンの形であるから、生成消滅演算子を用いて $[\hat{\phi}, \hat{n}] = i$ を用いて

$$\hat{H}_{\mathrm{LC}} = \hbar\omega(\hat{a}^\dagger\hat{a} + \frac{1}{2}) \tag{2.25}$$

で表すことができる。ただし $\omega = 1/\sqrt{LC}$。LC 回路のエネルギー準位は

$$E_n = \hbar\omega(n + \frac{1}{2}) \tag{2.26}$$

しかしこれでは各エネルギー準位間隔が等しくなってしまい、量子ビットをどのエネルギー準位間に等しい振動数のマイクロ波パルスで操作したとしても、全てのエネルギー準位間に作用することになってしまう。

トランズモンではインダクタの代わりにジョセフソン結合を導入する。この時のハミルトニアンは

$$\hat{H}_{\mathrm{tr}} = 4E_{\mathrm{c}}\hat{n}^2 - E_{\mathrm{J}}\cos\phi \tag{2.27}$$

ただし $E_{\mathrm{J}} = I_0\Phi_0/2\pi$ はジョセフソンエネルギーである。$\cos\phi$ を展開して 6 次以上の項を無視し、生成消滅演算子で置き換えると

$$\hat{H}_{\mathrm{tr}} = \hbar(\omega - \frac{\delta}{2})\hat{c}^\dagger\hat{c} + \hbar\frac{\delta}{2}\hat{c}^\dagger\hat{c}\hat{c}^\dagger\hat{c} + \frac{1}{2}\hbar\omega \tag{2.28}$$

ただし、$\delta = -E_{\mathrm{c}} < 0$ である。生成消滅演算子の定義式 $\hat{c}^\dagger\hat{c} = \sum_n n\,|n\rangle\langle n|$ からトランズモンのエネルギー準位は

$$E_n = \hbar(\omega - \frac{\delta}{2})n + \hbar\frac{\delta}{2}n^2 + \frac{1}{2}\hbar\omega \tag{2.29}$$

となる。例えば基底状態、第一励起状態、第二励起状態のそれぞれのエネルギー準位は

$$E_0, E_1, E_2 = \frac{1}{2}\hbar\omega, \frac{3}{2}\hbar\omega, \frac{5}{2}\hbar\omega + \hbar\delta \equiv \hbar\omega_0, \hbar\omega_1, \hbar\omega_2 \tag{2.30}$$

となり、それぞれのエネルギー差に該当する振動数は

$$\omega_{01} = \omega_1 - \omega_0 = \omega \tag{2.31}$$
$$\omega_{12} = \omega_2 - \omega_1 = \omega + \delta \tag{2.32}$$
$$\omega_{02} = \omega_2 - \omega_0 = 2\omega + \delta \tag{2.33}$$

なお $\delta < 0$ であったから。、$\omega_{12} < \omega_{01}$ である。以上よりエネルギー準位間隔に非線形性が導入され、各エネルギー準位が全て異なる値を取っている。よってトランズモンではそれぞれのエネルギー準位間を選んで操作をすることが可能である。一般的には基底状態と第一励起状態をそれぞれ $|0\rangle$、$|1\rangle$ とラベル付けし計算基底状態として用いる。これを量子ビットといい、$\omega_{01}$ を量子ビットの振動数と呼ぶ。量子ビットとして扱っているトランズモンが、意図せず第二励起状態 $|2\rangle$ も含めた重ね合わせ状態にある場合は漏れエラーとしてみなされる。

逆の見方をすれば、パルスの振動数次第で意図的にエネルギー高準位にもアクセスできる。例えば $|2\rangle$ にアクセスする方法として (2.33) のように $|0\rangle$ から $|2\rangle$ に励起する $X_{02}$ ゲートを作る方法と、 (2.32) のように $|1\rangle$ から $|2\rangle$ に励起する $X_{12}$ ゲートを作る二つの方法がある。$X_{02}$ は大きな振動数を要求するため、IBM Quantum では $\omega_{02}/2$ の振動数をもつ二光子吸収過程により仮想準位を介した励起を待つ必要が

ある [65, 33]。二光子吸収過程の確率は光の強度の二乗に比例するので、通常のゲートのような一光子吸収過程に比べて長いゲート時間にするかパルスの振幅を大きくする必要があり、どちらも量子エラーを増幅してしまう。よって本研究では一量子トリットゲートとして $X_{12}$ を採用している。

d-量子ディット（d-Qudit）では表現できる基底数が $d^n$ と大きくなるため、少ないゲート操作で等価な量子計算をすることが可能である。しかし高準位ほど自然放出や電荷分散（charge dispersion）による脱位相エラーが大きくなるため、実験の観点から制限なく利用することは現実的でない [55]。つまり高準位も用いる場合はエラー増幅とゲート数減少のトレードオフがある。ただし基本的には量子ビットとして扱い、計算途中で一時的に量子トリットとして扱うハイブリッド型はメリットの方が大きい場合がある。このような場合では量子トリットとして扱っている部分の入力と出力が量子ビット空間であることが特徴である。例えばトフォリゲートなどの多量子ビットゲートの計算時のみ量子トリットとして扱い、効率よく分解する手法が知られている。

### 2.2.2　マイクロ波パルスによる量子ビット制御

この節ではトランズモンの時間発展をシュレディンガー方程式 $i\hbar\frac{\partial}{\partial t}|\psi(t)\rangle = \hat{H}_{\rm tr}|\psi(t)\rangle$ を解くことで考えてみる。トランズモンの自由場におけるハミルトニアンと量子状態

$$\hat{H}_{\rm tr}/\hbar = \sum_n \omega_n |n\rangle\langle n|, \ |\psi(t)\rangle = \sum_n c_n(t)|n\rangle \tag{2.34}$$

をシュレディンガー方程式に代入すると

$$ic_n{}'(t) = \omega_n c_n(t) \tag{2.35}$$

この微分方程式を解くとトランズモンの時間発展は

$$|\psi(t)\rangle = \sum_n c_n(0)e^{-i\omega_n t}|n\rangle \tag{2.36}$$

$$= c_0(0)|0\rangle + \sum_{i=1}^{n} c_n(0)e^{-i\omega_{i0}t}|n\rangle \tag{2.37}$$

よって振幅の絶対値は時間変化せず、$|i\rangle\,(i\neq 0)$ に $\omega_{0i}\equiv\omega_i-\omega_0$ に比例した相対シフトが生じるという同じ結果が得られた。なお最後の式では絶対位相 $e^{-i\omega_0 t}$ を除いた。

次にパルスによってトランズモンをドライブする手法について議論する。量子ビットを $\omega_{01}$ に非常に近い振動数のマイクロ波パルスでドライブすると $\omega_{01}+\omega_{\rm pulse} \gg \omega_{01}-\omega_{\rm pulse}$ が成り立つので回転波近似を用いることができ [15]、パルス振幅を $\Omega$、パルス位相を $\phi$ とした時の有効ハミルトニアンは

$$\hat{H}_{\rm eff}/\hbar = -\frac{1}{2}(\omega_{01}-\omega_{\rm pulse})\sigma_z - \Omega(\cos\phi\sigma_x + \sin\phi\sigma_y) \tag{2.38}$$

パルス振幅が時間依存しない定数値と仮定すれば時間非依存のハミルトニアンであるため、時間発展演算子 $\hat{U(t)} = e^{-i\hat{H}(\Delta t)/\hbar}, \Delta t = t - t_0$ で記述できる。

$$\psi(t) = \hat{U(t)}\psi(t_0) = e^{-i\hat{H}\Delta t/\hbar}\psi(t_0) \tag{2.39}$$

と表すことができる。この式から時間発展演算子 $\hat{U(t)}$ が量子ゲートに該当することがわかる。例として $Rx(\theta)$ ゲートを実装したい場合、式 (2.10) より

$$e^{-i\hat{H}\Delta t/\hbar} = e^{-i\sigma_x \frac{\theta}{2}} \tag{2.40}$$

が成り立つようなハミルトニアンは

$$\hat{H} = \frac{\theta}{2\Delta t}\sigma_x \tag{2.41}$$

式 (2.38) よりマイクロ波パルスのパラメータを

$$\omega_{\text{pulse}} = \omega_{01},\ \Omega = \frac{\theta}{2\Delta t},\ \phi = 0 \tag{2.42}$$

と設定すれば良い。式 (2.42) を見ると、x 軸周りの回転角 $\theta$ はパルスの振幅 $\Omega$ とパルス作用時間 $\Delta t$ に比例して増加する。x 軸周りの回転は $|0\rangle$ と $|1\rangle$ 間の振動に該当するためラビ振動でもある。重要な定理として、ハミルトニアンが $\hat{H} = \hat{H}_0 + \hat{H}_1 + ...$ というように複数のハミルトニアンの和として表されるとする。それぞれのハミルトニアン間が可換である時、つまりすべての $j$、$k$ において $[\hat{H}_j, \hat{H}_j] = 0$ が成り立つ時のみ

$$e^{-i\hat{H}t/\hbar} = e^{-i\hat{H}_0 t/\hbar} e^{-i\hat{H}_1 t/\hbar} ... \tag{2.43}$$

と分解することができる。逆に言えばそれぞれのハミルトニアンが非可換である時、このように分解することはできない。その一例として、もしパルスの振動数の設定が甘くハミルトニアンの $\sigma_z$ 成分が残ってしまった場合、$[\sigma_x, \sigma_z] \neq 0$ より非可換であるから $Rz$ ゲートと $Rx$ ゲートに分解できず、補正が非常に難しい [*1]。一般的に表現として、ハミルトニアンに $\sigma_x$、$\sigma_y$、$\sigma_z$ による線形和 $\hat{H} = n_x X + n_y Y + n_z Z$ である場合、ベクトル $(n_x, n_y, n_z)$ を中心軸とした回転になってしまう [4]。なお $Ry$ ゲートの実装は $\phi = \pi/2$ とすれば良い。

トランズモンが量子トリットとして扱われている場合に、先ほどと同じく $\omega_{01}$ のパルスでドライブしてもほぼ同じハミルトニアンになるが、3 基底における AC-Stark shift により複雑な相対位相エラーが生じる [*2]。またトランズモンでは電荷分散により $|2\rangle$ のエネルギー準位が大きく時間変動するため、キャリブレーションした $\omega_{12}$ とリアルタイムの $\omega_{12}$ に誤差が生じる。以上のようなエラーを補正するためには定期的なキャリブレーションか、エラーを自動補正するパルスシークエンスが必要である [*3]。

### 2.2.3　Virtual Z

本節では $Rz$ ゲートの実装手法について解説する。量子ビットにおいて常に生じている相対位相回転を $Rz$ として利用することも可能だが、実験面からはタイミングを測るための高速制御のところでどうしてもズレが生じてしまう。

多くの超伝導量子コンピュータでは、事実上実機の操作を必要としない virtual Z という手法が用いられる [66]。初期状態が $|0\rangle$ であり測定も Z 基底ならば、$|0\rangle$ に作用している $Rz$ と最後の $Rz$ は無視

[*1] 対象の量子ゲートの前後に複数のパラメータを持つ単一量子ビットゲートで挟み、最もゲート忠実度が高くなるようなパラメータ探索を行うこともできる [32]。ただし非可換由来の誤差は必ず残る。

[*2] パルスで量子ビットをドライブした時に生じるエネルギー準位シフト。

[*3] 式 (2.22) で議論したようなベリー位相とは別物である。

できる。$Rz$ を実装する必要があるのは、$Rz$ が x、y 軸周りの回転ゲートによって挟まれている場合である。最も簡単な例として $Rx(\theta_2)Rz(\theta_1)Rx(\theta_0)$ を考える。式 (2.38) において、パルスのパラメータを $(\Omega, \phi) = (-\frac{\theta_2}{2\Delta t}, \phi_1)$ と定めると

$$e^{-i\{-\Omega(\cos\phi\sigma_x+\sin\phi\sigma_y)\}\Delta t} = Rz(-\phi_1)Rx(\theta_2)Rz(\phi_1) \tag{2.44}$$

が成立する。$Rx(\theta_2)Rz(\phi_1)Rx(\theta_0)$ において $Rx(\theta_2)$ を上式のようにパルス位相 $\phi$ を回転させて実装すれば、$Rz(-\phi_1)Rx(\theta_2)Rz(\phi_1)Rx(\theta_0)$ となる。最後に残った $Rz(-\phi_1)$ は次のゲート実装の際に考慮される。

### 2.2.4　CNOT

量子計算には量子ビット間のエンタングルメントが必須である。エンタングルゲートとして最も一般的なものが CNOT ゲートである。CNOT ゲートは式 (2.15) にあるように非常にシンプルであり扱いやすい。しかし実機で実装する時は別のエンタングル相互作用を起こし、単一量子ビットゲートと組み合わせて CNOT を構成する [15]。その中でも交差共鳴 (Cross Resonance: CR) は単一量子ビットゲート実装と同じく、パルスのドライブのみで高精度な $ZX$ 相互作用を生じるため [*4]、IBM Quantum でも採用されている [67, 68, 69]。

CR は標的ビットの振動数を持つパルスを制御ビットにドライブするだけで生じる。有効ハミルトニアンは

$$\hat{H}_{\text{eff}}/\hbar = -\frac{1}{2}\Delta_{12}ZI + \frac{\Omega}{2}(IX - \frac{J}{2\Delta_{12}}ZX) \tag{2.45}$$

と表すことができる [70, 32]。ただし、$\Delta_{12} \equiv \omega_{01}^{\text{ctrl}} - \omega_{01}^{\text{targ}}$ であり、制御ビットの振動数 $\omega_{01}^{\text{ctrl}}$ とマイクロ波パルスの振動数 $\omega_{01}^{\text{targ}}$ 間の大きなズレによって生じる AC-Stark shift 由来の $ZI$ エラーも加えている [71]。$\Omega$ はパルス振幅、$J$ は量子ビット間のカップリング定数である。式 (2.45) にある項のうち不要項である $ZI$ と $IX$ を自動補正する echo シークエンスや crosstalk 由来の $IY$ エラーの抑制 [68]、rotary echo [72] などが開発され、ゲート忠実度の向上が進められている。

CR においてエンタングルメントの役割を担っているのは $ZX$ 項であり、これは

$$ZX(\theta) = e^{-iZX\frac{\theta}{2}} = \begin{bmatrix} \cos\theta/2 & -i\sin\theta/2 & 0 & 0 \\ -i\sin\theta/2 & \cos\theta/2 & 0 & 0 \\ 0 & 0 & \cos\theta/2 & i\sin\theta/2 \\ 0 & 0 & i\sin\theta/2 & \cos\theta/2 \end{bmatrix} \tag{2.46}$$

と表すことができる。二量子ビットが取りうる基底を入力とした時に

$$ZX(\theta)\,|00\rangle = \cos(-\theta/2)\,|00\rangle + i\sin(-\theta/2)\,|01\rangle \tag{2.47}$$
$$ZX(\theta)\,|01\rangle = i\sin(-\theta/2)\,|00\rangle + \cos(-\theta/2)\,|01\rangle \tag{2.48}$$
$$ZX(\theta)\,|10\rangle = \cos\theta/2\,|10\rangle + i\sin\theta/2\,|11\rangle \tag{2.49}$$
$$ZX(\theta)\,|11\rangle = i\sin\theta/2\,|10\rangle + \cos\theta/2\,|11\rangle \tag{2.50}$$

[*4] $ZX$ という表記は $ZX \equiv Z \otimes X$ というようにテンソル積を省略したものである。

となる。つまり $ZX$ は制御ビットが $|0\rangle$ か $|1\rangle$ によって、標的ビットにおいて x 軸周りに逆回転の Rabi 振動を引き起こす。CNOT は制御ビットが $|0\rangle$ の時は何もせず、制御ビットが $|1\rangle$ の時は x 軸まわりに $\pi$ 回転させるゲートなので、$ZX$ によって位相差が図 2.3 のように $\pi$ ズレれば良い。以上より $ZX$ は回

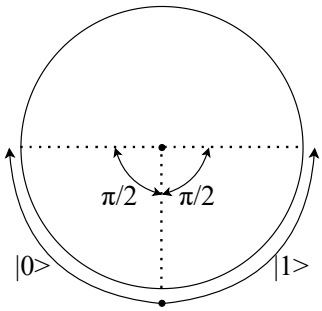



図 2.3　$ZX$ 相互作用による制御ビットが $|0\rangle$、$|1\rangle$ の時の標的ビットにおける Rabi 振動の様子。$ZX$ 項の回転角が $\pi/2$ の時に Rabi 振動の位相差が $\pi$ になっている。

転角が $\pi/2$ の時に最大のエンタングルメントを生み出すことが分かる。しかし $ZX = \pi/2$ のままでは制御ビットが $|0\rangle$ の時も標的ビットが $\pi/2$ 回転するので、$Rx$ ゲートを追加して補正すれば良い。なお $[ZX, IX] = 0$ より可換なので、CR パルスと $IX$ パルスは同時に打つことでゲート時間を短縮できる。このような最小の時間で構成する CNOT を direct CNOT と呼ぶ [69, 73, 74]。完全な CNOT にするためには図 2.4 にあるように相対位相の補正も必要である。

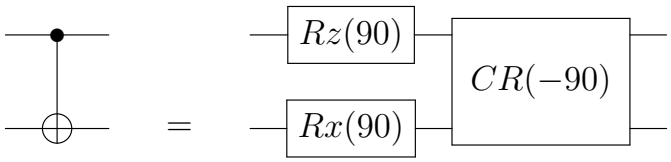



図 2.4　Direct CNOT の構成方法 [67, 15]。

ただ実際には式 (2.45) で見たように、CR を引き起こすと $ZX$ 項だけが存在するわけではない。しかも $ZI$ エラーの振幅が最も大きい [71, 32]。2.2.3 節でみたように、$ZI$ は virtual Z で補正することができるが、CR における $ZI$ エラーは振動数や時間に依存するため、高いゲート忠実度を保つためには頻繁にキャリブレーションを行う必要があり安定性が低い。そこで CR パルスを二つに分割し、制御ビットに対して $X$ ゲートを挟む形である echo CNOT が開発された [68]。Echo CNOT のパルスシークエンスは図 2.5 のようになる。Echo CNOT では途中で制御ビットにおける $X$ ゲートにより基底変換が行われる。

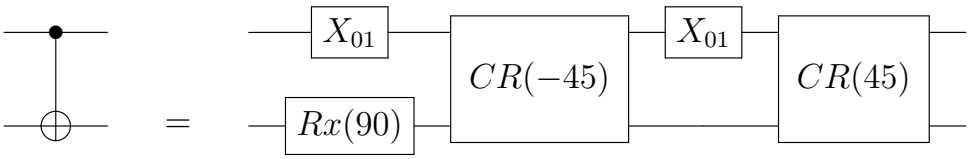



図 2.5　Echo 型の CNOT の構成方法 [68, 33]。

これによって制御ビットが $|0\rangle$、$|1\rangle$ それぞれの時に等しく $ZI$ による相対位相の回転が起きる。等しい相

対位相のズレは絶対位相となるので実質自動でキャンセルされることになる。また $IX$ エラーについても、分割された CR の符号が逆なのでこれも自動でキャンセルされる。

以上の挙動は全て量子ビットに限った場合を考えている。量子トリット上で CR を引き起こすと別のハミルトニアンになることを考慮をすれば、量子ビット上で実装した CNOT をそのまま再利用することはできない。現時点で量子トリットとしてのトランズモンにおいて CR を引き起こした際の解析的な有効ハミルトニアンの導出は行われていない。ただし量子トリット空間で echo CNOT を二つ並べると $C_2X_{01}$ ゲートになるという報告があった [42]。それでもなお CNOT2 個分のコストがかかる上に、位相エラーが自動補正されないことが分かっている。

## 2.3　量子エラー

量子計算機には様々な量子エラーが存在する。量子ビットが完全な基底状態になっていない初期化エラー、ゲートのキャリブレーションエラー、量子ビット間の crosstalk、デコヒーレンス、測定エラーなど様々なものが混合している。

### 2.3.1　量子エラー

理想的には全ての種類の量子エラーを別々に記述する必要があるが、いまだに量子エラーについてわかっていないところが多い。そこで理論的にシンプルに記述できるフリップエラーモデルを考えてみる。フリップエラーでは確率 $p$ で量子ビットが任意の軸周りに $\pi$ 回転するというものである。まず量子ビットの混合状態 $\rho$ を

$$\rho = \frac{1}{2}(I + r_x X + r_y Y + r_z Z) \tag{2.51}$$

と表す。例えば x 軸まわりのフリップエラー $\xi_x$ は Kraus 演算子を用いて、

$$\xi_x(\rho) = (1-p)\rho + pX\rho X \tag{2.52}$$

と表すことができる。同様にして y、z 軸周りのフリップエラーも記述することができ、これらのエラーが等しい確率 $p/4$ でどれかのエラーが起こる場合は、式 (2.51) を用いて

$$\xi_d(\rho) = (1 - \frac{3}{4}p)\rho + \frac{p}{4}(X\rho X + Y\rho Y + Z\rho Z) \tag{2.53}$$

$$= (1-p)\rho + p\frac{I}{2} \tag{2.54}$$

となり、これを脱分極（depolarizing）エラーと呼ぶ。任意の量子ビット数 $n$ に拡張すると

$$\xi_d(\rho) = (1-p)\rho + p\frac{I}{2^n} \tag{2.55}$$

となる。つまりこれは $p$ の確率で量子状態が完全に混合し、該当する量子ビットにおいて取りうる全ビット列が等しい割合で存在するようになるというエラーである。脱分極エラーが正確に量子エラーを表現しているかはともかく、Randomized benchmarking も脱分極エラーが仮定されているように、ゲートエラーによってフィデリティが指数減衰する結果をよく表現する [75]。また IBM Quantum を用いて非常

に長い量子回路を計算すると、多くの場合において全てのビット列が等しい確率で測定されるようになっていく。ただし実際の量子エラーは 2.2.4 節で見たような量子ゲートに残ったエラー項であったり、自然放出 $T_1$ と位相緩和 $T_2$ である。

### 2.3.2　量子トモグラフィ

量子状態や量子ゲートの精度を評価するためには量子状態を推定する必要がある。測定しようとする量子状態に対して何も事前知識がない場合は全ての情報を測定から得ることはできないため、純粋状態ではなく混合状態として推定することしかできない。そこで全ての量子トモグラフィでは混合状態を推定する。また量子状態の情報を得ようとする時に測定という操作を必ずしなければならないが、測定は対象の波動関数量子状態の収縮により量子状態を非可逆的に破壊してしまう。そのため何度も同じ量子状態を用意して様々な測定をする。しかし NISQ は量子状態を用意する過程でも測定する過程でもノイズを含んでおり、完全に同じ量子状態を用意することは出来ない。これは不可避な現象であるので、一般的には量子状態を用意する過程でも測定する過程では全く同じエラーが発生しているという仮定のもと量子状態の推定が行われる。

量子ビットの混合状態は式 (2.51) のように表されるのであった。そこで量子状態トモグラフィ（quantum state tomography）では式 (2.51) におけるパラメータである $\vec{r} = (r_x, r_y, r_z)$ の推定を行う。二つのパウリ演算子 $P_a$ と $P_b$ の積のトレースはクロネッカーのデルタを用いて $\mathrm{Tr}(P_a P_b) = \delta_{ab}$ となるから $r_a = \mathrm{Tr}(P_a \rho)$ が成り立つ。$\rho = \sum_i p_i \left|\psi_i\right\rangle \left\langle\psi_i\right|$ で表される時

$$\mathrm{Tr}(P_a \rho) = \sum_i p_i \left\langle\psi_i\right| P_a \left|\psi_i\right\rangle \tag{2.56}$$

が成り立つ。この式は $r_a$ が混合状態における観測量 $P_a$ の期待値であることを示している。例えば観測量 $Z$ の期待値、つまり $r_z$ を考えてみる。 2.1.3 節で見たように、Z 基底で測定するとそれぞれ $p(0)$ と $p(1) = 1 - p(0)$ の確率で $|0\rangle$ と $|1\rangle$ が得られる。測定で $|0\rangle$ が得られれば Z の観測量が +1 であることと同値であり、$|1\rangle$ が得られれば Z の観測量は-1 であるから

$$r_z = +1 \times p(0) - 1 \times p(1) = 2p(0) - 1 \tag{2.57}$$

というように $r_z$ を測定結果から計算することができる。$r_x$ と $r_y$ の場合は測定基底を変換すれば同様にして得られる。しかしこのやり方では $n$ 量子ビット数を測定する場合、各量子ビットを $X$、$Y$、$Z$ のどの基底で測定するかという組み合わせが $3^n$ 必要であるため、量子状態トモグラフィは指数増加な計算量を要求する。そのため現実的には基本量子ゲートのキャリブレーション等にしか用いられることはなく、量子計算を行う場合は全量子ビットを Z 基底でのみ測定を行う。

量子状態トモグラフィでは量子状態の推定を行ったが、量子ゲートが量子エラーの影響もふくめて最終的にどのような作用をしたかを評価するためには量子プロセストモグラフィ [76] が必要となる。未知の n 量子ビットゲートの作用は未知の Kraus 演算子 $A$ を用いて

$$\xi(\rho) = A \rho A^\dagger \tag{2.58}$$

と表現できる。未知の Kraus 演算子 $A$ は事前に固定した演算子集合 $\{E_m\}$ によって $A = \sum_m a_m E_m$ と分解することができる。ただし $m = 1, 2, ...4^n$。なお演算子集合 $\{E_m\}$ は一般的にパウリ演算子の組み合

わせの集合が用いられ、一量子ビットでは $\{I, X, Y, Z\}$、二量子ビットでは $\{II, IX, ..., ZZ\}$ が用いられる。これを代入すると

$$\xi(\rho) = \sum_{m,n} a_m a_n^* E_m \rho E_n^\dagger \tag{2.59}$$

$$\equiv \sum_{m,n} \chi_{mn} E_m \rho E_n^\dagger \tag{2.60}$$

と表せる。ここで $4^n$ 個の線型独立な混合状態 $\{\rho_j\}$ を用意する。未知の量子ゲートを作用させた後の混合状態を量子状態トモグラフィによって同定すると

$$\xi(\rho_j) = \sum_k c_{jk} \rho_k \tag{2.61}$$

となり、$c_{jk}$ が実験によって測定される。ただし量子計算機で $|0\rangle\langle 1|$ などの混合状態は簡単に用意できないので、$|0\rangle\langle 0|$、$|1\rangle\langle 1|$、$|+\rangle\langle +|$、$|-\rangle\langle -|$ を用意し、これらに未知の量子ゲートを作用させた結果の線形結合によって、$|0\rangle\langle 1|$ に未知の量子ゲートを作用した時の結果を推定する。また量子状態トモグラフィでは $3^n$ の実験回数が必要であったのだから、これを $4^n$ 個の線型独立な混合状態に対して繰り返すと、量子プロセストモグラフィでは $12^n$ の実験回数が必要になってしまうことが分かる。式 (2.60) にも $\rho_j$ を代入し、$E_m \rho_j E_n^\dagger = \sum_k \beta_{mnjk} \rho_k$ というように展開する。$\beta_{mnjk}$ は $\{E_m\}$ と $\{\rho_j\}$ は既にこちら側が指定しているので、簡単な計算で求まる。係数比較より

$$c_{jk} = \sum_{m,n} \chi_{mn} \beta_{mnjk} \tag{2.62}$$

となり、逆行列を用いると次の式が得られる [4]。

$$\chi_{mn} = \sum_{j,k} \beta^{-1}_{mnjk} c_{jk} \tag{2.63}$$

以上より量子エラーを含む未知の量子ゲート $\xi$ を構成するための $\chi_{mn}$ 行列が求まる。

### 2.3.3　忠実度

量子計算機には様々な量子ノイズが存在し、それらによって計算結果が理想の結果からズレてしまう。そのため得られた実験結果がどれだけ理想の結果に近いかを評価する指標が必要である。よく用いられる指標として忠実度（fidelity）と呼ばれるものがある。忠実度 $F$ は二つの混合状態 $\rho$ と $\sigma$ が得られているとして、

$$F = \mathrm{Tr}(\sqrt{\rho^{1/2} \sigma \rho^{1/2}}) \tag{2.64}$$

と表される。$\rho$ と $\sigma$ が近い混合状態であればあるほど 1 に近い値を示し、全く異なった混合状態であるほど 0 に近づく。定義として $F = \left\{\mathrm{Tr}(\sqrt{\rho^{1/2} \sigma \rho^{1/2}})\right\}^2$ とする場合もあるが、距離の近さの指標としては片方に統一すればどちらでも問題ない。なお片方が純粋状態であった時、つまり $\rho = |\psi\rangle\langle\psi|$ である時 $\rho^2 = |\psi\rangle\langle\psi|$ だから

$$F = \sqrt{\langle\psi| \sigma |\psi\rangle} \tag{2.65}$$

が成り立つ。しかし 2.3.2 節で見たように量子状態トモグラフィは指数増加な実験回数を要求するので多くの量子ビットを測定する場合は適していない。よって一般的に Z 基底でしか測定を行わないので最後の相対位相のエラーを考慮する必要がないことを利用して相対位相の情報は無視する古典忠実度（classical fidelity）がある。

$$F = \sum_i \sqrt{p_i^\rho p_i^\sigma} \tag{2.66}$$

ただし、$i$ はビット列の識別番号である。確率分布はどちらも Z 基底で測定されたものである。

特に量子計算機の中でも最も大きいエラー源は量子ゲート、特にエンタングルを担う二量子ビットゲートである。二量子ビットゲートは一量子ビットゲートに比べてマイクロ波パルスとの相互作用時間が 10 倍ほど長いため、エラー率も 1 桁大きくなってしまう。さらに量子誤り訂正も閾値を超えた精度の量子ゲートを繰り返し用いることで任意の精度までエラー訂正を行うという戦略であるため、FTQC 時代であっても量子ゲートエラーは必ず存在し、そもそも原理的に完全に取り除くことはできない。量子ゲートの精度を評価する手法の中でも最も基本的な指標の一つとして量子プロセストモグラフィを利用したゲートフィデリティがある。量子プロセストモグラフィによってノイズを含む未知の量子ゲートを完全に特徴付けられるのだった。入力状態が $|\psi\rangle$ である時に、理想の量子ゲート $U$ が作用した後の純粋状態とノイズを含む未知の量子ゲートが作用した後の混合状態の近さを評価すれば良いから、式 (2.65) より

$$F_g = \sqrt{\langle\psi| U^\dagger \xi(|\psi\rangle\langle\psi|) U |\psi\rangle} \tag{2.67}$$

しかしこのフィデリティは初期状態に依存して値が変わってしまう。そこで任意の初期状態の中で最も小さくなった $F_g$ をゲート忠実度と呼ぶ。実際に多く用いられるのは任意の初期状態における $F_g$ の平均値であり、これを平均ゲート忠実度と呼ぶ [77]。

ゲート忠実度だけを測りたいならば、量子プロセストモグラフィによる方法はデメリットが複数存在する。最も重要なものとして初期化エラーや測定エラー (合わせて SPaM エラー) といった量子ゲートエラーと関係ないエラーの影響を取り除けないことである。もちろんこれらは実験開始前に delay 時間を長く取ったり、得られる結果に対して測定エラー緩和手法を適用するなどして緩和できる。また多量子ビットゲートを評価したい場合は指数増加な実験回数が必要となってしまう。以上のデメリットを克服した手法が Randomized Benchmarking(RB) である [75]。RB ではランダムな量子ゲートを並べた時のフィデリティの指数減衰をフィティングすることで平均ゲート忠実度を測るので、SPaM エラーに依存しない。さらに量子ゲートをランダムに並べることにより様々な入力状態に対するゲートの挙動を多項式実験回数で見ることができる。しかし量子プロセストモグラフィより得られる様々な情報が圧倒的に少ないことと、ランダム性と指数減衰の正確さを担保するために非常に多くのゲートを作用させる必要があるため、エラー率の非常に高い量子ゲートに対しては適していない。

### 2.3.4　量子エラー緩和

量子エラーは大きく測定エラーとゲートエラーに分けることができる。この節ではその中でも測定エラーに対する Qiskit Ignis に実装されている緩和手法 [25]、CNOT ゲートエラーに対する zero-noise extrapolation (ZNE) [24] という緩和手法を紹介する。この 2 つの緩和法は、本研究の回路最適化における 3.2 において用いられている。

Qiskit Ignis では事前に測定エラーによってどのように測定結果がずれるかの指標となるキャリブレーション行列を実験によって構成する。その後実際に得られたビット列の確率分布に対して事前に得られたキャリブレーション行列の逆行列をかけることで、測定エラーが緩和された確率分布を得ることができる。$A$ をキャリブレーション行列、測定される前のエラーのない確率分布のベクトルを $P_b$、実際に測定された確率分布のベクトルを $P_a$ をとすれば、測定で起こっているのは

$$P_a = AP_b \tag{2.68}$$

であるから、実際に実験によって得られるのは $A$ の逆行列と $P_a$ を用いて

$$P_b = A^{-1}P_a \tag{2.69}$$

しかしこのやり方では測定誤差によって $P_b$ の中に負の確率を持つビット列が生じる非物理的な結果が得られることがあるので、

$$\min_{P_b} ||P_a - AP_b|| \tag{2.70}$$

というように最小二乗法によって $P_b$ を得る。

次に ZNE のうち、基本ゲートに分解された量子回路中にある CNOT を全て $n$ 倍に増やし CNOT エラーの影響を大きくした結果と元々の結果を用いて、CNOT の脱分極エラーが 0 であると予想される結果へと外挿する Fixed Identity Insertion Method を議論する。ただし $n$ は奇数にすることで量子回路の等価性を保つ。量子混合状態 $\rho$ に対して、$i$ 番目の CNOT について考える。エラー率を $\epsilon_i$ まず CNOT のエラーが脱分極エラーである時、

$$\mathcal{E}(\rho) = (1-\epsilon_i)CX_i\rho CX_i + \{1-(1-\epsilon_i)\}(\frac{I}{4} \oplus \rho_{\text{other}}) \tag{2.71}$$

と表せる。ただし $\rho_{\text{other}}$ は CNOT と関係ない量子ビットの密度行列である。ここで CNOT を $n$ 倍にすると

$$\mathcal{E}(\rho) = (1-\epsilon_i)^n CX_i\rho CX_i + \{1-(1-\epsilon_i)^n\}(\frac{I}{4} \oplus \rho_{\text{other}}) \tag{2.72}$$

$$\sim (1-n\epsilon_i)CX_i\rho CX_i + n\epsilon_i(\frac{I}{4} \oplus \rho_{\text{other}}) \tag{2.73}$$

ただし二式目は $\epsilon_i \ll 1$ よりテイラー展開を用いた。これを測定すると

$$\langle M\rangle(n) = (1-n\epsilon_i)\langle M\rangle_{ex} + n\epsilon_i\langle M\rangle_{dep_i} + \mathcal{O}(\epsilon_i^2) \tag{2.74}$$

元々の量子回路に CNOT が N 個あるとすれば、これら全ての CNOT も同様に n 倍にすることで

$$\langle M\rangle(n) = (1-n\sum_{i=1}^{N}\epsilon_i)\langle M\rangle_{ex} + n\sum_{i=1}^{N}\epsilon_i\langle M\rangle_{dep_i} + \mathcal{O}(\epsilon_i^2) \tag{2.75}$$

全ての CNOT のエラー率 $\epsilon_i$ を $\epsilon$ に統一すれば

$$\langle M\rangle(n) = (1-nN\epsilon)\langle M\rangle_{ex} + n\epsilon\sum_{i=1}^{N}\langle M\rangle_{dep_i} + \mathcal{O}(\epsilon^2) \tag{2.76}$$

(2.76) を用いて元々の回路 $n = 1$ と CNOT 数を 3 倍にした $n = 3$ を比較すると

$$\langle M\rangle(1) = (1 - N\epsilon)\langle M\rangle_{ex} + \epsilon \sum_{i=1}^{N} \langle M\rangle_{dep_i} + \mathcal{O}(\epsilon^2) \tag{2.77}$$

$$\langle M\rangle(3) = (1 - 3N\epsilon)\langle M\rangle_{ex} + 3\epsilon \sum_{i=1}^{N} \langle M\rangle_{dep_i} + \mathcal{O}(\epsilon^2) \tag{2.78}$$

よって

$$\frac{3}{2}\langle M\rangle(1) - \frac{1}{2}\langle M\rangle(3) = \langle M\rangle_{ex} + \mathcal{O}(\epsilon^2) \tag{2.79}$$

以上より脱分極エラーの一次項がキャンセルされてエラー緩和がされたことがわかる。

# 第 3 章

# 量子回路最適化プロトコル AQCEL

本章で述べる AQCEL (Advancing Quantum Circuit by icEpp and Lbnl) については [57] で公表した。AQCEL は初期状態に依存して不要な制御操作の削除を行う量子回路最適化プロトコルである。さらに量子回路中で繰り返し現れるゲート群（Reccuring Sets of Gates ; RSG）を見つける手法も含んでいる。RSG については付録 A において議論している。AQCEL のフローチャートを図 3.1 に示した。

本章では 3.1 節で不要なビット制御削除の理論的な条件式の導出、 3.2 節では導出した条件式を満たすか判断するために必要なビット列の測定方法、 3.3 節では測定されたビット列から最も少ないビット制御数とビット制御位置を多項式計算量で判断するための方法と AQCEL を簡単な例に適用してみた結果、3.4 節では AQCEL をパートンシャワー量子アルゴリズムに適用してみた結果を紹介する。

このとき任意の初期状態の時と特定の初期状態をみた時に制御ゲートの形がかわる図を載せる。

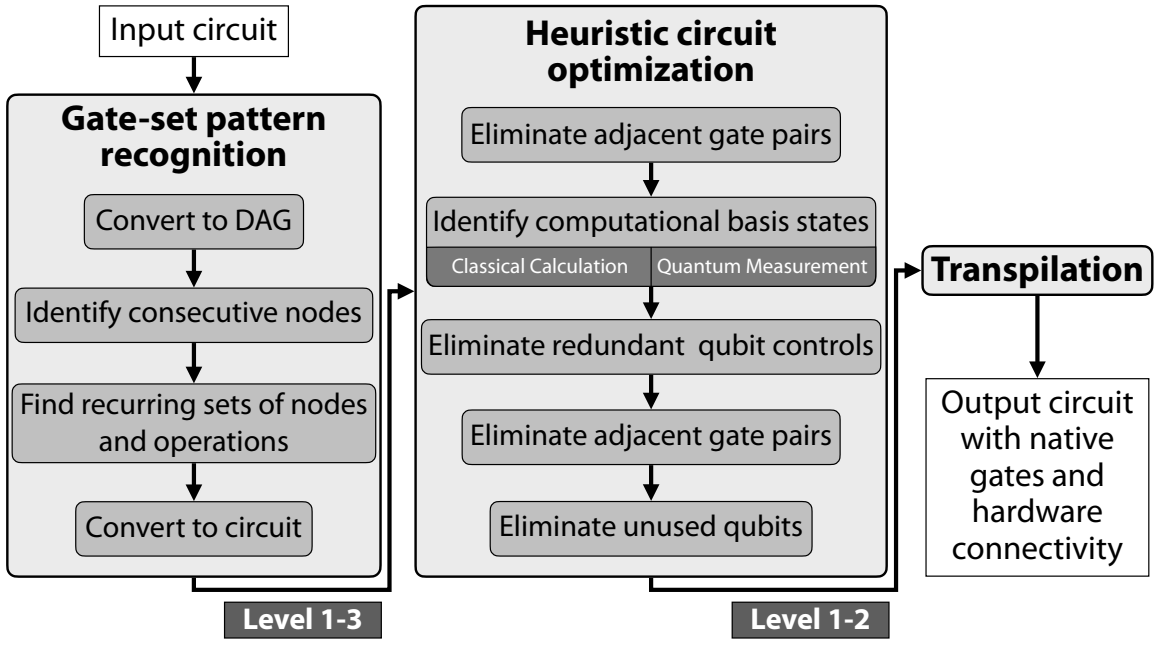



図 3.1　AQCEL のフローチャート [57]。入力された量子回路から繰り返しゲート群（RSG）を識別し、その後回路最適化が行われる。RSG 識別については付録 A を、回路最適化については第 3 章を参考されたい。

## 3.1　不要なビット制御削除

$n$ を量子回路にある全ての量子ビット数、$m$ をビット制御数として、任意の多量子ビット制御ゲートとして $C^m[U]$ を考える。$C^m[U]$ 直前の量子状態を $|\psi\rangle$ とし、次のように表せる。

$$|\psi\rangle = \sum_{j,k} c_{j,k} |j\rangle_{\text{ctl}} \otimes |k\rangle \tag{3.1}$$

$|\cdot\rangle_{\text{ctl}}$ は制御ビットの状態であり、もう片方はそれ以外の量子ビットの状態である。それぞれの状態は $0 \leq j \leq 2^m - 1$ と $0 \leq k \leq 2^{n-m} - 1$ で表現する。この制御ゲートは全ての制御ビットの状態が $|1\rangle$ である時にのみ作用するから、この表記法では

$$|j\rangle_{\text{ctl}} = |11\ldots1\rangle = |2^m - 1\rangle_{\text{ctl}} \tag{3.2}$$

が $C^m[U]$ が作用する基底の条件である。次に $x(< m)$ 個のビット制御数を削除した $C^{m-x}[U]$ を考える。$C^{m-x}[U]$ 直前の量子状態も同じく $|\psi\rangle$ であるが、3 つの添字を用いて次のように表してみる。

$$|\psi\rangle = \sum_{i,l,k} \tilde{c}_{i,l,k} |i\rangle_{\text{ctl}'} \otimes |l\rangle_{\text{free}} \otimes |k\rangle\,, \tag{3.3}$$

$|\cdot\rangle_{\text{ctl}'}$ は制御し続ける $m - x$ 個の量子ビットの状態で、$|\cdot\rangle_{\text{free}}$ はビット制御が解除された $x$ 個の量子ビットの状態を表す。式 (3.1) と比較すれば

$$|i\rangle_{\text{ctl}'} \otimes |l\rangle_{\text{free}} = |2^x i + l\rangle_{\text{ctl}}\,, \tag{3.4}$$

が成り立ち、$0 \leq i \leq 2^{m-x} - 1$ となる。式 (3.2) と同様に

$$\tilde{c}_{i,l,k} = c_{2^x i + l, k}. \tag{3.5}$$

が成り立つ。$C^m[U]$ を $|\psi\rangle$ に作用させると

$$C^m[U]|\psi\rangle = \sum_{j=0}^{2^m-2} \sum_k c_{j,k} |j\rangle |k\rangle + \sum_k c_{2^m-1,k} |2^m - 1\rangle U |k\rangle\,, \tag{3.6}$$

$x$ 個のビット制御を削除した $C^{m-x}[U]$ を $|\psi\rangle$ に作用させると

$$C^{m-x}[U]|\psi\rangle = \sum_{i=0}^{2^{m-x}-2} \sum_{l,k} \tilde{c}_{i,l,k} |i\rangle |l\rangle |k\rangle + \sum_{l,k} \tilde{c}_{2^{m-x}-1,l,k} |2^{m-x} - 1\rangle |l\rangle U |k\rangle\,. \tag{3.7}$$

$x$ 個のビット制御を削除しても良い条件は、式 (3.6) と式 (3.7) の右辺同士が等しいことであるので、

$$\sum_{l=0}^{2^x-2} \sum_k \tilde{c}_{2^{m-x}-1,l,k} |2^{m-x} - 1\rangle |l\rangle U |k\rangle = \sum_{l=0}^{2^x-2} \sum_k c_{2^m - 2^x + l, k} |2^m - 2^x + l\rangle |k\rangle\,. \tag{3.8}$$

$U|k\rangle$ を次のように表記する。

$$U|k\rangle = \sum_{k'} u_{kk'} |k'\rangle \tag{3.9}$$

式 (3.4)、式 (3.5), 式 (3.8) を用い、左辺の $k'$ と $k$ を置き換えると

$$\sum_{l=0}^{2^x-2}\sum_{k,k'}\tilde{c}_{2^{m-x}-1,l,k'}u_{k'k}\left|2^{m-x}-1\right\rangle\left|l\right\rangle\left|k\right\rangle=\sum_{l=0}^{2^x-2}\sum_{k}\tilde{c}_{2^{m-x}-1,l,k}\left|2^{m-x}-1\right\rangle\left|l\right\rangle\left|k\right\rangle. \tag{3.10}$$

以上より

$$\sum_{k'}\tilde{c}_{2^{m-x}-1,l,k'}u_{k'k}=\tilde{c}_{2^{m-x}-1,l,k}\quad\forall l\in\{0,1,...,2^x-2\},k. \tag{3.11}$$

式 (3.11) は列ベクトル $\{\tilde{c}_{2^{m-x}-1,l,k}\}_k$ が $0\leq l\leq 2^x-2$ において行列 $u$ の固有値 1 に対応する固有ベクトルであることを意味する。

しかし列ベクトル $\{\tilde{c}_{2^{m-x}-1,l,k}\}_k$ は制御ゲート直前の量子状態 $|\psi\rangle$ の一部の基底の振幅のリストである。 2.1.3 節でも述べたように、量子状態の振幅を求めることは指数増加な計算量が必要となるため、この条件が成立するかどうかを調べることも指数増加な計算量が必要であることになる。そこで AQCEL ではこの条件の中から多項式計算量で調べる事が可能な十分条件を考える必要があり、そのうちの一つが

$$\tilde{c}_{2^{m-x}-1,l,k}=0\quad\forall l\in\{0,1,...,2^x-2\},k. \tag{3.12}$$

である。式 (3.12) は、$l=2^x-1$ 以外の全ての $l$ において全制御ビットで 1 のビット列の振幅が 0 であることである。言い換えると、$C^m[U]$ は制御しないけれども $C^{m-x}[U]$ は制御してしまう基底がないという意味である。この条件式からわかることは、AQCEL が多量子ビットゲートからビット制御を削除できるような場合は、直前の量子状態において使われていない基底が存在するということである [*1]。つまり AQCEL が有効な量子アルゴリズムは初めに全ての量子ビットにアダマールゲートを作用させて全ての基底を用いるようなグローバー探索 [6] や、量子位相推定 [78, 5] には適用できない。

振幅が 0 かどうかを調べることも古典計算機では制御ゲート直前までの行列計算をする必要があるので指数増加な計算量がかかる。しかし、振幅が 0 ならば量子計算機で十分に繰り返し測定しても測定されないことを意味し、これは多項式計算量で達成可能である。詳細は次節 3.2 節で議論する。また他にも十分条件は多くあり、これらの十分条件はさらなるビット制御削除の可能性を広げる。

同様にして全てのビット制御を削除する ($x=m$) 場合は

$$U\left|\psi\right\rangle=\sum_{j,k}c_{j,k}\left|j\right\rangle U\left|k\right\rangle \tag{3.13}$$

全てのビット制御を削除しても良い条件は、式 (3.6) と式 (3.13) の右辺同士が等しいことであるので、

$$\sum_{j=0}^{2^m-2}\sum_{k}c_{j,k}\left|j\right\rangle U\left|k\right\rangle=\sum_{j=0}^{2^m-2}\sum_{k}c_{j,k}\left|j\right\rangle\left|k\right\rangle \tag{3.14}$$

ここでも式 (3.9) を用い、左辺の $k'$ と $k$ を置き換えると

$$\sum_{j=0}^{2^m-2}\sum_{k}\sum_{k'}c_{j,k'}u_{k'k}\left|j\right\rangle\left|k\right\rangle=\sum_{j=0}^{2^m-2}\sum_{k}c_{j,k}\left|j\right\rangle\left|k\right\rangle \tag{3.15}$$

[*1] ここでは全て Z 基底を想定している。

以上より

$$\sum_{k'} c_{j,k'} u_{k'k} = c_{j,k} \quad \forall j \in \{0, 1, ..., 2^m - 2\}, \forall k. \tag{3.16}$$

式 (3.16) は列ベクトル $\{c_{j,k}\}_k$ が行列 $u$ の固有値 1 に対応する固有ベクトルであることを意味する。ただしこれも式 (3.11) と同様に振幅の情報を必要とするので、多項式計算量で調べる事が可能な十分条件を取り出す必要がある。そのうちの一つが

$$c_{j,k} = 0 \quad \forall j \in \{0, 1, ..., 2^m - 2\}, \forall k. \tag{3.17}$$

この式の意味は式 (3.12) と同じで、$j = 2^m - 1$ 以外の全ての $j$ を持つ基底の振幅が 0 であることである。言い換えると、$C^m[U]$ は制御しないけれども $C^{m-x}[U]$ は制御してしまう基底がないという意味である。

## 3.2　ビット列の測定

3.1 節でも述べたように、$C^m[U]$ 直前の量子状態 $|\psi\rangle$ が式 (3.12) もしくは式 (3.17) を満たすかどうかを調べなければならない。これは制御ビットのみを繰り返し測定すれば良いことを示す。$C^m[U]$ の制御ビットを Z 基底で測定すると $|2^{m-x} - 1\rangle |l\rangle$ に当たるビット列が得られる。全ての制御ビットを測定すると得られるビット列の種類は $2^m$ 個あるが、式 (3.12) では $C^{m-x}[U]$ も制御するビットにおいては $|2^{m-x} - 1\rangle = |11...1\rangle$ であるビット列にのみに条件があり、$|l\rangle = |11...1\rangle$ である場合を除いてそれらの総数は $2^x - 1$ 個である。それぞれが得られる確率が 0 であれば良いから

$$\sum_k |\tilde{c}_{2^{m-x}-1,l,k}|^2 = 0 \quad \forall l \in \{0, 1, ..., 2^x - 2\} \tag{3.18}$$

(3.12) と (3.18) は同値な関係であることから、条件がついている $2^x - 1$ 個のビット列が測定されないことを調べれば良いことがわかる。全ビット制御が削除できるか調べる時も同様で

$$\sum_k |c_{j,k}|^2 = 0 \quad \forall j \in \{0, 1, ..., 2^m - 2\} \tag{3.19}$$

よって条件がついている $2^j - 1$ 個のビット列が測定されない、言い換えれば全ての要素が 1 である $\{11...1\}$ のビット列のみが測定されることを調べれば良いことがわかる。

次に制御ビットにおいて取りうるビット列を得る方法について議論する。古典計算機で制御ビットにおいて取りうるビット列を得ようと思えば $C^m[U]$ 直前まで量子回路のシミュレーションを行う必要がある。行列計算と同様の計算量が必要となる状態ベクトルシミュレータを用いると正確なビット列の組み合わせを得ることができるが、指数増加な計算量が必要となってしまう。この計算量を下げるために振幅は無視し、使われ得る基底の組み合わせのみを追っていくやり方がある。この方法は制御ビットにおいて取りうるビット列の種類を多めに見積もってしまうが、ゲート数を $N$ とした時に計算量を $\mathcal{O}(N2^n)$ まで抑えることができるので AQCEL における古典シミュレーションはこの方法を採用している。

それでもなお指数増加な計算量が必要であることは古典計算機を用いてのビット列測定は非効率であることを示唆している。 1.1.3 でも紹介したように、古典計算機上でグラフィカルに量子回路を表現し量子ビットの情報を得ることも可能であるが、あくまで量子ビットの状態を多項式計算量で追っているため、制御ビットの取りうるビット列を常に判断できるわけではない。

そこで本研究では量子計算機でビット列を測定する手法について議論する。まず $C^m[U]$ 直前までの量子回路を量子計算機で計算し制御ビットを測定する。ただし小さい振幅を持つ基底を正確に判断するために測定回数を十分多くする必要がある。いくら多くの回数実験を繰り返しても全ての基底を測定することは統計誤差により不可能であるが、測定回数を十分大きくすることで任意の精度まで高めることができる。量子回路中の全ての制御ゲートに対して不要なビット制御削除を行いたければ、以上のプロセスを全ての制御ゲートで前から順番に繰り返せば良い。それぞれの制御ゲートにおけるプロセスにおいて測定回数を $M$ とすると、全ての制御ゲートに対して量子計算機による制御ビット測定が行われた場合の全ゲート操作回数と全測定回数は

$$M\{m + (1+m) + (2+m) + \cdots + (N-1+m)\} = \frac{1}{2}MN(N-1) + mMN \tag{3.20}$$

である。よって計算量のオーダーは $\mathcal{O}(MN^2 + mMN)$ となり、多項式計算量である事がわかる。ただし測定回数を $2^m$ 程度に増やしすぎると計算量が指数増加になってしまうので、ある程度の測定回数に留めなければならず、そう言う意味では量子計算機によるビット列測定は近似的な結果を得ることに等しい。つまり量子計算機を用いてビット列測定を行うと多項式計算量でできるが、その結果に基づいたビット制御削除を行った後の量子回路は近似回路になり得るということである。しかし NISQ 時代においてはこの性質が実機上での結果をさらに改善する可能性があることを 3.4 節において議論する。

量子計算機でビット列を測定する時に生じる問題点として、測定エラーやゲートエラーなどによって理想的には 3.20 のように確率が 0 のはずのビット列も測定されてしまうということである。これを軽減する対策は測定エラー緩和とゲートエラー緩和を導入することである。本研究ではビット列測定のプロセスにおいて 2.3.4 で紹介した Qiskit Ignis の測定エラー緩和手法と ZNE による CNOT エラー緩和手法を用いる。しかし量子エラー緩和はノイズの影響を完全にキャンセルできないため、結局のところ確率が 0 のはずのビット列も測定されてしまう現象は避けられない。もし量子エラー訂正により任意の精度まで高めることができるならばこの現象は解消される。しかし NISQ においては不要に測定されてしまうビット列を判断し、切り捨てる必要がある。そこで本研究では閾値を導入する。最も簡単な閾値として一定の閾値 $s^f_\epsilon$ が挙げられる。量子ゲートの数や実機のエラー率を無視し、全ての制御ゲートに対して一定の閾値を与える。この方法はシンプルな反面、量子回路の前半後半ではゲートエラーの積み重なりが異なるという性質を無視したり、どれくらいの大きさの閾値が良いかの指標もないというデメリットがある。$s^f_\epsilon$ のデメリットを解決する方法として、全ての量子ゲートのエラーの積み重なりを定量的に予測した値を閾値として用いる方法がある。しかしゲートエラーの挙動を予測することは非常に難しく、高い精度を出す事が難しい。

基本ゲートの単一量子ゲート ($U_{1,2,3}$) のエラー率を $\epsilon^{(i)}_{\mathrm{U}}$、CNOT のエラー率を $\epsilon^{(i,j)}_{\mathrm{CX}}$ をおく。ただし $i$ と $j$ はゲートが作用している量子ビットを指す。またエラー率は IBM 側で Rondomized Benchmarking によって測定され、随時更新されている値を用いている。厳密には Rondomized Benchmarking によって得られるゲート忠実度は相対位相のズレまで考慮しているが、AQCEL では Z 基底でのみ測定を行い相対位相のズレには影響を受けない。本研究では全ての単一量子ゲート ($U_{1,2,3}$) が正しく作用する確率を $p_{\mathrm{U}}$、

全ての CNOT が正しく作用する確率を $p_{\mathrm{CX}}$ と近似する。

$$p_{\mathrm{U}} = \prod_i \left(1-\epsilon_{\mathrm{U}}^{(i)}\right)^{n_{\mathrm{U}}^{(i)}} \tag{3.21}$$

$$p_{\mathrm{CX}} = \prod_{i\neq j} \left(1-\epsilon_{\mathrm{CX}}^{(i,j)}\right)^{n_{\mathrm{CX}}^{(i,j)}} \tag{3.22}$$

$n_{\mathrm{U}}^{(i)}$ と $n_{\mathrm{CX}}^{(i,j)}$ はそれぞれの index が指す量子ビットに作用する $U_{1,2,3}$ と CNOT のゲート数である。ここで $U_{1,2,3}$ のエラー率と CNOT のエラー率をそれぞれ $\epsilon_{\mathrm{U}}$ と $\epsilon_{\mathrm{CX}}$ に統一し、CNOT のエラー率が $U_{1,2,3}$ のエラー率より充分大きく、かつ CNOT のエラー率が十分 1 より小さいと仮定すれば

$$\epsilon_{\mathrm{U}} \ll \epsilon_{\mathrm{CX}} \ll 1 \tag{3.23}$$

よってここでは $U_{1,2,3}$ のエラー率は無視しても良いとし、$N_{\mathrm{U}}$ を $U_{1,2,3}$ の全ゲート数、$N_{\mathrm{CX}}$ を $CX$ の全ゲート数とすれば全量子ゲートが正しく作用する確率である $p_\epsilon$ は

$$\begin{aligned} p_\epsilon &= 1 - p_{\mathrm{U}} p_{\mathrm{CX}} && (3.24)\\ &\sim 1-(1-\epsilon_{\mathrm{U}})^{N_{\mathrm{U}}}(1-\epsilon_{\mathrm{CX}})^{N_{\mathrm{CX}}} && (3.25)\\ &\sim N_{\mathrm{CX}}\epsilon_{\mathrm{CX}}. && (3.26) \end{aligned}$$

とテイラー展開により、近似をすることができる。ただし本研究では CNOT エラー緩和として ZNE を導入しているから、緩和後に得られるビット列は式 (3.24) が予測するよりも少し良い結果が得られる。ZNE では定式化の簡便化のため近似が用いられているが、精密さが要求される閾値にとっては荒い近似になってしまうので、式 (2.72) において用いたテイラー近似と式 (2.76) において用いた全ての CNOT のエラー率を統一する近似を用いない。そうすると式 (2.76) は

$$\langle M\rangle(n) = \{\prod_i (1-\epsilon_i)^n\}\langle M\rangle_{\mathrm{ex}} + \{1-\prod_i (1-\epsilon_i)^n\}\langle M\rangle_{\mathrm{noise}} \tag{3.27}$$

ここでは正しい測定結果 $\langle M\rangle_{\mathrm{ex}}$ 以外の測定量は全てノイズ項として $\langle M\rangle_{\mathrm{noise}}$ とおいた。よって式 (2.79) は

$$\frac{3}{2}\langle M\rangle(1)-\frac{1}{2}\langle M\rangle(3) = \{\frac{3}{2}\prod_i(1-\epsilon_i)-\frac{1}{2}\prod_i(1-\epsilon_i)^3\}\langle M\rangle_{\mathrm{ex}}+[1-\{\frac{3}{2}\prod_i(1-\epsilon_i)-\frac{1}{2}\prod_i(1-\epsilon_i)^3\}]\langle M\rangle_{\mathrm{noise}} \tag{3.28}$$

となる。正しい観測量 $\langle M\rangle_{\mathrm{ex}}$ が得られる確率を $p_{\mathrm{CX}}^{\mathrm{zne}}$ とおけば、$p_{\mathrm{CX}}$ を用いて

$$p_{\mathrm{CX}}^{\mathrm{zne}} = \frac{3}{2}p_{\mathrm{CX}} - \frac{1}{2}p_{\mathrm{CX}}^3 \tag{3.29}$$

と表せる。式 (3.22) と比較すると

$$\begin{aligned} p_{\mathrm{CX}}^{\mathrm{zne}} - p_{\mathrm{CX}} &= (\frac{3}{2}p_{\mathrm{CX}} - \frac{1}{2}p_{\mathrm{CX}}^3) - p_{\mathrm{CX}} && (3.30)\\ &= \frac{1}{2}p_{\mathrm{CX}}(1-p_{\mathrm{CX}}^2) \geq 0 && (3.31) \end{aligned}$$

が成り立ち、改善されていることがわかる。よって式 (3.24) を更新すると

$$p_\epsilon^{\mathrm{zne}} \sim 1 - p_{\mathrm{CX}}^{\mathrm{zne}} \tag{3.32}$$

$$= 1 - \left(\frac{3}{2}p_{\mathrm{CX}} - \frac{1}{2}p_{\mathrm{CX}}^3\right) \tag{3.33}$$

$$\sim N_{\mathrm{CX}}^2 \epsilon_{\mathrm{CX}}^2 \tag{3.34}$$

式 (3.34) はテイラー展開を用いているが、荒い近似となるので以降は式 (3.33) を用いる。$p_\epsilon^{\mathrm{zne}}$ はゲート数が増えるにつれて大きくなるという性質も満たしている。

以上より、ZNE 適用後の全ての量子ゲートのエラーの積み重なりを定量的に近似した値を求める事ができた。$m$ 個の制御ビットを測定した場合測定されるビット列は最高で $2^m$ 個ある。ゲートエラーの積み重ねが $2^m$ 個のビット列のうちどのビット列に影響を与えるかまで予測していないため、次のように 3 種類の状況を仮定する。まず初めに全てのゲートエラーの影響が一つのビット列に集中したと仮定すると

$$s_\epsilon^{\mathrm{high}} := p_\epsilon^{\mathrm{zne}}. \tag{3.35}$$

次に全てのゲートエラーの影響が全てのビット列に分散したと仮定すると

$$s_\epsilon^{\mathrm{low}} := p_\epsilon^{\mathrm{zne}}/2^m, \tag{3.36}$$

$s_\epsilon^{\mathrm{high}}$ と $s_\epsilon^{\mathrm{low}}$ の中間として

$$s_\epsilon^{\mathrm{mid}} := (s_\epsilon^{\mathrm{low}} + s_\epsilon^{\mathrm{high}})/2. \tag{3.37}$$

とする。$s_\epsilon^{\mathrm{low}}$、$s_\epsilon^{\mathrm{mid}}$、$s_\epsilon^{\mathrm{high}}$ は回路の初めの方では非常に小さな閾値となる。この時に式 (2.70) で緩和しきれなかった測定エラーやゲートエラーの見積もりミスの影響を受けやすいので、常に $s_\epsilon^{\mathrm{low}}, s_\epsilon^{\mathrm{mid}}, s_\epsilon^{\mathrm{high}} \geq 0.05$ という条件を課している。最後に、$s_\epsilon^{\mathrm{low}}$、$s_\epsilon^{\mathrm{mid}}$、$s_\epsilon^{\mathrm{high}}$ は回路の後半では非常に大きな閾値となってしまい、全てのビット列をノイズ由来として削除してしまうので $s_\epsilon^{\mathrm{low}}, s_\epsilon^{\mathrm{mid}}, s_\epsilon^{\mathrm{high}} \leq 0.2$ という制限をつける場合がある。

## 3.3　不要な制御操作の削除

この節では 3.2 で書いたように測定されたビット列が、式 (3.18) を満たすかどうかを調べる方法について議論する。量子計算機を使って $m$ 個の制御ビットを $M$ 回測定したとき、測定されたビット列の種類を $M_{\mathrm{b}}$ とすれば

$$\mathcal{O}(M_{\mathrm{b}}) = \begin{cases} \mathcal{O}(M) & (M \leq 2^m) \\ \mathcal{O}(2^m) & (M \geq 2^m) \end{cases} \tag{3.38}$$

古典計算機によるシミュレーションを用いれば全ての取り得るビット列を得られるので $\mathcal{O}(2^m)$ となる。

ここで $m$ 個あるビット制御のうちいくつかを削除する場合の組み合わせは、一つ一つのビット制御を削除するかしないかの 2 通りが $m$ 回続くので $2 \times 2 \times \ldots \times 2 = 2^m$ 個もの組み合わせがある。ビット列 $\{11\ldots1\}$ のみが測定されれば式 (3.17) を満たし全てのビット制御を削除すればよいが、そうでない場合は $2^m - 1$ もの新しいビット制御の組み合わせのうち、測定されたビット列の組み合わせから式 (3.18) を満たすものはどれかを調べる必要がある。まず $2^m - 1$ もの新しいビット制御の組み合わせのうち一つを取

り出し、これが式 (3.18) を満たすかどうかを調べるための計算量を考える。式 (3.18) は削除するビット制御数を $x$ とした時に $2^x-1$ 個のビット列に対して測定されるべきではないという条件がついている。条件がついているビット列が測定された $M_\mathrm{b}$ 個のビット列にないことを確認する必要がある。この計算量は $\mathcal{O}(M_\mathrm{b}2^x)$。ビット列のサイズは $m$ であること、削除するビット制御数は最大で $m$ であることを考慮すれうば $\mathcal{O}(mM_\mathrm{b}2^m)$。最も多くのビット制御を削除できる場合を見つけるためにこのプロセスを $2^m$ 種類のビット制御の組み合わせで試す必要があるので $\mathcal{O}(mM_\mathrm{b}4^m)$。量子回路中に $N$ 個の制御ゲートがある場合は $\mathcal{O}(mM_\mathrm{b}4^mN)$。

しかしこれでは $m$ に関して指数増加な計算量が必要となってしまう。そこで本研究では任意の制御ゲート $C^m[U]$ を作業ビットを用いて $\mathcal{O}(m)$ 個のトフォリと二量子ビットゲートに分解することで [60, 61, 62]、$m=1,2$ という条件をつけることができる。ただし制御ゲート数は $\mathcal{O}(m)$ 倍となり $\mathcal{O}(mN)$ 個となるから、削除できるビット列の組み合わせ探索に必要な計算量は $\mathcal{O}(mM_\mathrm{b}4^mN)$ から $\mathcal{O}(mN)$ となる。制御ゲート数の増加に伴ってビット列の測定に必要な計算量は

$$M\{k+(1+k)+(2+k)+\cdots+(mN-1+k)\}=\frac{1}{2}MmN(mN-1)+kMmN,\quad k=1,2 \tag{3.39}$$

よって計算量は $\mathcal{O}(m^2MN^2)$ であり多項式計算量である。古典シミュレーションではビット列測定の計算量が $\mathcal{O}(mN2^n)$ となる。以上の結果を表 3.1 にまとめた。この表では量子回路中にビット制御数が $n-1$ 個の制御ゲートが $N$ 個ある場合の計算量をまとめている。作業ビットを用いなくても $\mathcal{O}(m^2)$ 個のトフォリと二量子ビットゲートに分解することができるが、作業ビットに伴う SWAP 数と CNOT 数はトレードオフの関係にある。

表 3.1　AQCEL における計算量一覧

| | 制御ゲートの分解なし | 制御ゲートの分解あり |
|---|---|---|
| ビット列測定（古典, 量子） | $\mathcal{O}(N2^n)$, $\mathcal{O}(MN^2+nMN)$ | $\mathcal{O}(nN2^n)$, $\mathcal{O}(n^2MN^2)$ |
| 不要なビット制御削除 | $\mathcal{O}(nM_\mathrm{b}4^nN)$ | $\mathcal{O}(nN)$ |
| 全体（古典, 量子） | $\mathcal{O}(nM_\mathrm{b}4^nN)$ | $\mathcal{O}(nN2^n)$, $\mathcal{O}(n^2MN^2)$ |

今まではビット制御削除について議論してきたが制御操作自体の削除、つまり制御ゲート自体を恒等演算子として削除できる場合もある。式 (3.6) において $C^m[U]$ が恒等演算子であるためには $C^m[U]\left|\psi\right\rangle=\left|\psi\right\rangle$ であれば良いから

$$\sum_k c_{2^m-1,k}\left|2^m-1\right\rangle U\left|k\right\rangle=\sum_k c_{2^m-1,k}\left|2^m-1\right\rangle\left|k\right\rangle \tag{3.40}$$

ここでも式 (3.9) を用い、左辺の $k'$ と $\leftrightarrow k$ を置き換えると

$$\sum_{k,k'}\tilde{c}_{2^m-1,k'}u_{k'k}\left|2^m-1\right\rangle\left|k\right\rangle=\sum_k\tilde{c}_{2^m-1,k}\left|2^m-1\right\rangle\left|k\right\rangle. \tag{3.41}$$

以上より

$$\sum_{k'}\tilde{c}_{2^m-1,k'}u_{k'k}=\tilde{c}_{2^m-1,k}\quad\forall k. \tag{3.42}$$

式 (3.42) は列ベクトル $\{\tilde{c}_{2^m-1,k}\}_k$ が行列 $u$ の固有値 1 に対応する固有ベクトルであることを意味する。ただしこれも式 (3.11) と同様に振幅の情報を必要とするので、多項式計算量で調べる事が可能な十分条件を取り出す必要がある。そのうちの一つが

$$\tilde{c}_{2^m-1,k} = 0 \quad \forall k. \tag{3.43}$$

この式の意味は非常にシンプルであり、$C^m[U]$ はそもそも制御ビットにおいて全て $|1\rangle$ である基底にしか作用しないから、$C^m[U]$ が作用する基底の振幅が 0 ならば良いということを示している。よって制御ビットを繰り返し測定した時に

$$\sum_k |\tilde{c}_{2^m-1,k}|^2 = 0 \tag{3.44}$$

よって全ての要素が 1 である $\{11...1\}$ のビット列が測定されないことを調べれば良いことがわかる。この場合 $C^m[U]$ は直前の量子状態を $|\psi\rangle$ に対しては恒等演算子であり、削除することができる。

以上より AQCEL の全ての最適化条件を説明したので、ここで CNOT とトフォリに対するビット制御削除の条件式を考えてみる。全てのビット制御を削除するためには CNOT は $\{1\}$、トフォリは $\{11\}$ のビット列のみがそれぞれ測定され、制御ゲート自体を削除するためには CNOT は $\{1\}$、トフォリは $\{11\}$ のビット列がそれぞれ測定されてはいけない。トフォリの 2 つのビット制御のうち 1 つを削除するための条件は式 (3.18) を用いて考えてみる。

$$\sum_k |\tilde{c}_{1,0,k}|^2 = 0 \tag{3.45}$$

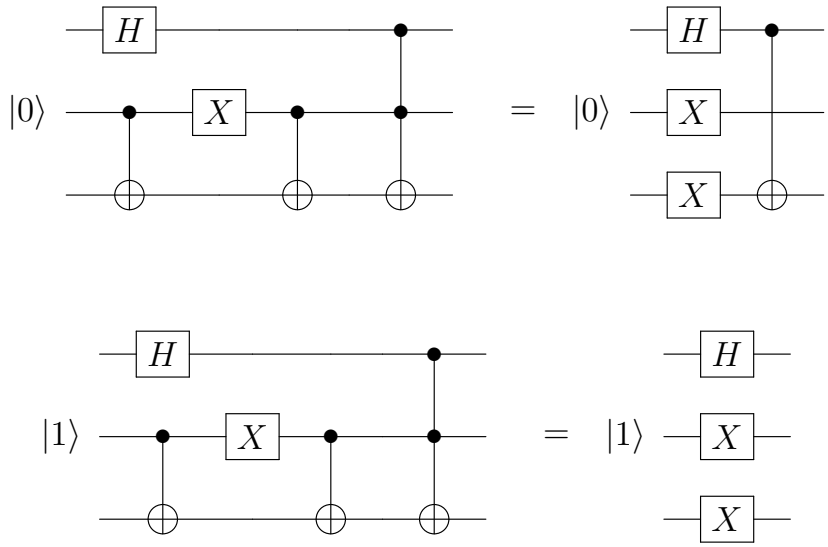
$\tilde{c}_{1,0,k}$ は $|1\rangle_{\text{ctl}'} |0\rangle_{\text{free}} |k\rangle$ の振幅であったことを思い出せば、一つ目のビット制御を削除するためにはビット列 $\{01\}$ が、二つ目のビット制御を削除するためにはビット列 $\{10\}$ が測定されなければよい。AQCEL のビット列測定と不要な制御操作削除プロトコルをまとめた擬似コードを以下の通りである [57]。

**Algorithm 1**: Redundant controlled operations removal

**for all** $C[U]$ or $C^2[X]$ gate $g$ in the circuit **do**
  execute circuit up to, but not including, $g$
  **if** $g$ is a $C[U]$ gate **then**
    measure the control qubit $q$ in the $Z$ basis multiple times
    **if** $\{1\}$ is observed in the measurement results **then**
      **if** $\{0\}$ is not observed in the measurement results **then**
        turn $g$ into a $U$ gate acting on the target qubit
      **end if**
    **else**
      eliminate $g$
    **end if**
  **else**
    measure the control qubits $q_1 q_2$ in the $Z$ basis multiple times

**if** $\{11\}$ is observed in the measurement results **then**
  **if** neither $\{00\}$, $\{01\}$ nor $\{10\}$ is observed in the measurement results **then**
    turn $g$ into an $X$ gate acting on the target qubit
  **else if** $\{01\}$ is not observed in the measurement results **then**
    eliminate the control on $q_1$
  **else if** $\{10\}$ is not observed in the measurement results **then**
    eliminate the control on $q_2$
  **end if**
**else**
  eliminate $g$
**end if**
**end if**
**end for**

この擬似コードに従った最適化の一例を紹介する。




上記の二つの量子回路を見ると、ゲート構成は全く同じでも初期状態に応じて異なる回路に最適化される事がわかる。これは AQCEL では式 (3.11) や式 (3.43) にあるように、制御ゲート直前の量子状態 $|\psi\rangle$ に対する条件のもと成り立つものであることからも明らかである。なおこの例では二つ目の量子ビットに対してのみ特定の初期状態を与え、他の量子ビットは任意の量子状態である。二つ目の量子ビットの初期状態が $|0\rangle$ である時、1 つ目の CNOT は制御ビットにおいてビット列 $\{0\}$ しか得られないので CNOT 自体を削除することができる。二つ目の CNOT は制御ビットにおいてビット列 $\{1\}$ しか得られないのでビット制御を削除することができ、$X$ ゲートに置き換えることができる。トフォリゲートの 2 つの制御ビットでは二つのビット列 $\{01\}$ と $\{11\}$ が得られるので、二つ目のビット制御を削除することができる。二つ目の量子ビットの初期状態が $|1\rangle$ である時、1 つ目の CNOT は制御ビットにおいてビット列 $\{1\}$ ビット制御を削除することができ、$X$ ゲートに置き換えることができる。二つ目の CNOT は制御ビットにお

いてビット列 $\{0\}$ しか得られないので CNOT 自体を削除することができる。トフォリゲートの 2 つの制御ビットでは二つのビット列 $\{00\}$ と $\{10\}$ が得られるので、トフォリ自体を削除することができる。

このように AQCEL は $2^n$ の全基底のうち一部の基底のみが用いられている場合に制御操作を削除することができる最適化プロトコルであることが分かる。さらに $2^n$ の全基底のうちどの基底が用いられているかは初期状態に依存するため、AQCEL は初期状態に依存したプロトコルでもある。全基底のうち一部の基底のみが用いられているケースというのは、物理シミュレーションなどで様々な初期状態に対応した量子回路を設計した場合に生じる。物理シミュレーションなどでは各粒子やイベント毎に基底を割り当てるため、様々な初期状態でも正しく実行される汎用性の高い一般的な量子回路を設計すると、ある特定の初期状態のみを計算したい場合には使われない基底が、特に量子回路の前半で生じる。本研究ではこのような物理シミュレーション量子アルゴリズムのうち、パートンシャワー量子アルゴリズムをベンチマークとして用いた。その結果は 3.4 において議論している。

トフォリも CNOT と単一量子ビットゲートに分解することが可能であるが、これをしないメリットは 3 つ存在する。一つ目はクラウド通信によって実機を利用しているためトフォリは CNOT が 6 つに分解されるため単純にクラウド通信と順番待ちを 6 倍の回数をする必要があり非常に長い時間を要する。二つ目は NISQ に特徴的な実用的な問題であり、さらに実験回数が増えるとエラーの影響による結果の不安定さが増してしまう。特にトフォリに対する不要な制御操作を正しく削除するためには分解先の 6 つの CNOT 全てで正しく不要な制御操作の削除が達成される必要があるが、NISQ の不安定さに加えて、トフォリの制御ビットを直接測定するよりも分解されて増えたゲート分少し深い回路を実行する必要があり、エラーの影響が大きくなってしまう。最後に最も重要な理由として、より少ないビット制御数をもつ制御ゲートに分解すると削除できなくなるビット制御が生じるからである。

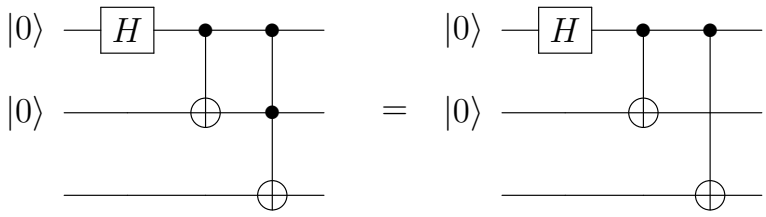
式 (3.12) を思い出すと、ビット制御数が $m+z$ 個から $x$ 個のビット制御を削除する条件は

$$\tilde{c}_{2^{m+z-x}-1,l,k} = 0 \quad \forall l \in \{0,1,...,2^x-2\}, k. \tag{3.46}$$

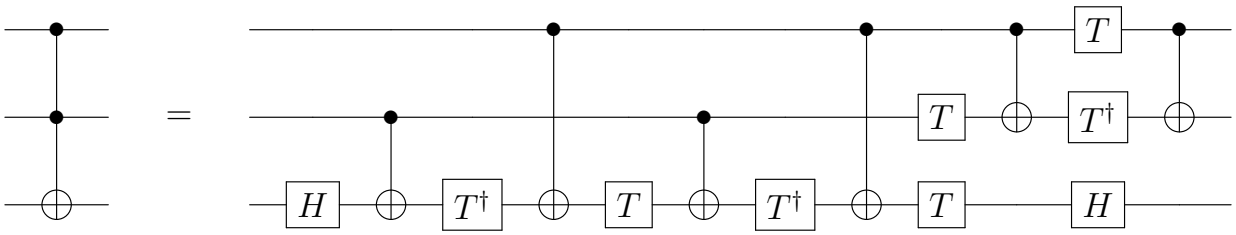
ただし制御ビット数が $z$ 個増えた分、制御ビット以外を除く全ての量子ビット数は $z$ 個減っている。よって $k$ の要素数は $2^{n-m}$ から $2^{n-m-z}$ 個に減少する。$k$ の要素数が減少するということは、振幅が 0 という条件がかかっている基底数も減少したということであり、式 (3.12) と比べて式 (3.46) の方が緩い条件をかけていることになる。つまり制御ゲートを小さな制御ゲートに分解することで指数増加な計算量を多項式計算量に抑えることが可能になるが、ビット制御を削除できる条件がより厳しくなっていくので削除できなくなるビット制御が生じてしまう。実際に CNOT とトフォリの例で考えると

$$c_{1,k} = 0 \quad \forall k. \tag{3.47}$$
$$\tilde{c}_{1,0,k} = 0 \quad \forall k. \tag{3.48}$$

CNOT は $|1\rangle|k\rangle$ の全基底に対して振幅が 0 という条件がかかっており、トフォリは $|10\rangle|k\rangle$ の全基底に対して振幅が 0 という条件がかかっている。制御ビット数が 1 個違う分、CNOT は $2^{n-1}$ 個の基底、トフォリは $2^{n-2}$ 個の基底に条件がかかっている。これの一例として次の量子回路の最適化プロセスを考えてみる。直接トフォリの制御ビットを測定するやり方ならば
というように最適化ができる。しかしトフォリを次のように分解すると、それぞれの CNOT の制御ビットの量子状態は $|0\rangle$ と $|1\rangle$ の重ね合わせ状態になっているので、ビット列 $\{0\}$ と $\{1\}$ の両方が測定され




てしまい、不要な制御操作の削除を行う事ができない。これは非常に面白い性質であり、CNOT は制御ビットが $|0\rangle$ もしくは $|1\rangle$ のどちらかの 1 つの状態である時、言い換えれば他の量子ビットとエンタングルしていない時にしか不要な制御操作ができない反面、トフォリやそれ以上のビット制御数を持つ制御ゲートについては制御ビット同士も含め、他の量子ビットとエンタングルしていてもビット制御を外せる場合があるということである。この性質が ZX-calculus のうち Copy rule [37] と Relaxed Peephole Optimization [39] も考慮していない新規性である。以上より、表 3.1 で見たように多量子ビットゲートは分解しない限り多項式計算量にならないが、分解しすぎると式 (3.12) と比べて式 (3.46) の比較から分かるようにいくつかの不要なビット制御削除の機会を失うというトレードオフの関係にある。




AQCEL ではこの他にも連続するゲートペアや不要な量子ビットの削除も実装している。量子ゲートは全てユニタリ演算子 $U$ であるので、エルミート共役である $U^\dagger$ が連続して並んでいれば $UU^\dagger = I$ となり、恒等演算子となるので削除する事ができる。このプロセスの計算量は $\mathcal{O}(nN)$ である。不要な制御操作削除プロトコルは時間のかかる最適化なので、この計算量を少しでも減らすために初めに連続するゲートペア削除を行う。不要な制御操作削除後も新たに連続するゲートペアが生じるケースがあるので、その後にもう一度連続するゲートペア削除を行う。全ての最適化が行われた後、ゲートや測定など何も操作されていない量子ビットが出る場合がある。最後にこのような不要な量子ビットを削除する。このプロセスの計算量は $\mathcal{O}(nN)$ である。このフローチャートは図 3.1 に記載している。

## 3.4　パートンシャワー量子アルゴリズム (QPS) への応用

1.1.1 節で紹介した物理解析への応用が期待されている量子アルゴリズムのうち、多量子ビットゲートが多くあり、任意の初期状態に対応できるようにあらかじめビット制御数に余裕を持たせる場合がある。このようなビット制御は特定の初期状態のみを計算する時に不要であり、これが AQCEL による改善の余地となる。その一例として QPS [9] を本研究で開発した AQCEL の応用対象とする。

素粒子実験では新物理の寄与により標準模型からの逸脱が期待される実験データと、標準模型に従うシミュレーションを比較することで新物理の発見を目指している。しかしこれを第一原理計算することは非現実であるため近似を用いる必要がある。素粒子反応には factorization という性質があり、異なるエネ

ルギースケール毎に切り離して計算をすることができる。近距離のハードなプロセスと長距離のハドロン化があり、その中間のソフトなプロセスとしてパートンシャワーの 3 つに分類できる。パートンシャワーとはクォークやグルーオン、つまりパートンが様々な分岐、対生成を起こしながら安定な粒子へと時間発展する過程である。この過程が確率的なものであることを利用してモンテカルロ法を用いて数値的に精度良くシミュレーションすることができる [79]。しかしモンテカルロ法は確率分布に従ったランダムなサンプリングをしているだけなので、始状態と終状態が同じである事象間の干渉効果は考慮されない。QPS [9] では、自然に干渉効果が入る量子計算機を用いることで、モンテカルロで失われる干渉効果を多項式計算量で取り入れることに成功している。古典計算機では振幅も含めた行列計算をする必要があるので指数増加計算量が必要となってしまう。

扱う粒子は 2 種類のフェルミオン $f_1$ と $f_2$、1 種類のボソン $\phi$ である。$f_1$ と $f_2$ は確率的に $\phi$ を放出し、$\phi$ は二つのフェルミオン対に分岐する。この際にフェルミオンはフレーバーを変える事ができる。結合定数として $f_1$ と $\phi$ 間の結合定数を $g_1$、$f_1$ と $\phi$ 間の結合定数を $g_2$、$f_1\bar{f}_2$ ($\bar{f}_1 f_2$) と $\phi$ 間の結合定数を $g_{12}$ とする。シャワーの過程は $N_{\text{evol}}$ ステップ数に離散化して表現し、それぞれのステップにおいて 1 粒子のみが反応を起こす。そのためステップ数が増えるとより精密な結果が得られる。図 3.2 は $m$ 番目のステップの量子回路を表している。まず量子回路を簡単にするためにフレーバーの変化が起きないようなフェルミオンの基底 $f_a$、$f_b$ へと変換し、各粒子数を $n_a$、$n_b$、$n_\phi$ のレジスタの保存する。これらの各粒子数を用いてスダコフ因子により放出や分岐が起こる確率を計算し、$e$ のレジスタにおいて放出がおきるならば $|1\rangle$、そうでないなら $|0\rangle$ とする。$|1\rangle$ と $|0\rangle$ のそれぞれの確率はスダコフ因子で計算した確率である。その後どの粒子が放出もしくは分岐をするのかを $h$ のレジスタに保存し、これに従って粒子の状態を変化させる。最後に元の基底に戻す。

なお、QPS 回路で実験をする目的は、先行研究である ZX-calculus や Relaxed peephole optimization に対する優位性を示すためではなく、最適化過程に NISQ による繰り返し測定を取り入れても AQCEL が有効であるということの実証と、閾値による低振幅基底の棄却による効率的な近似回路の生成の実証である。

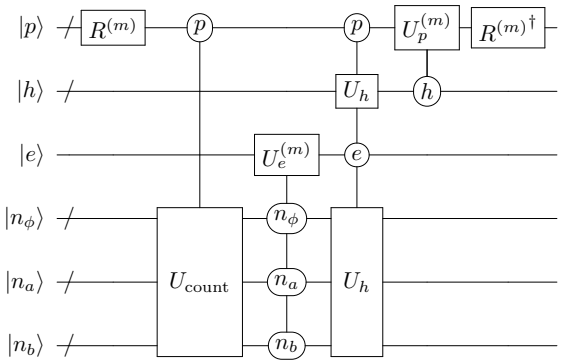



図 3.2　$m$ 番目のステップにおける量子回路 [9]。詳細は参考文献参照。

### 3.4.1　実験のセットアップ

量子回路と AQCEL の実装は全て IBM Qiskit の version 0.21.0 [80] を用いている。Qiskit の中の API のバージョンについては Terra 0.15.2、Aer 0.6.1、Ignis 0.4.0 である。Python 3.8 [81] を用いた。以下量子計算機によるビット列測定以外については全て single 2.4 GHz Intel core i5 processor の環境で行った。

また AQCEL の比較対象として t|ket⟩ (pytket 0.6.1) を利用した。t|ket⟩ の使い方として 10 個のパスのリスト [*2]のうち一つずつ適用して最もゲートを削減したパスを選び、これを QPS 回路に適用する。このように選ばれたパスの適用後の回路にも同様のプロセスを行う。この際一度使われたパスももう一度選ぶ事ができる。ゲート数が減らなくなるまでこれを繰り返された回路を t|ket⟩ によって最適化された回路として扱う。なお t|ket⟩ は transpilation 前に用いている。その理由は後述する。

ビット列測定と最終的な結果の測定のために用いた量子計算機は IBM Quantum Falcon Processors の一つである 27-qubit IBM's *ibmq_sydney* device である。Statevector simulator は Qiskit Aer のものを用いた。*ibmq_sydney* で実験する際には level 3 pass manager を用いたトランスパイルを 11 回繰り返し、最もゲート数が少なかったトランスパイル後の量子回路を実機で回した。

### 3.4.2　$N_{\mathrm{evol}} = 2$ ステップ

この節では QPS の 2 ステップシミュレーション回路に対する AQCEL による回路最適化について議論する。初期状態は $|f_1\rangle$、結合定数は $g_1 = 2$、$g_2 = g_{12} = 1$ とした。よってこの場合は $f \to f'\phi$ だけでなく $\phi \to f\bar{f}$ の反応も起きる汎用的な量子回路となる。

QPS の 2 ステップシミュレーションについては非常に長い回路であるため、ビット列の測定は古典シミュレーションのみで行い、理想的な回路最適化の結果を調べた。最適化後は量子ビット間の全結合。つまり SWAP が 0 で済む理想的な状況を仮定し、単一量子ビットゲート $(U_1, U_2, U_3)$ と CNOT に分解した。図 3.3 では単一量子ビットゲートと CNOT それぞれのゲート数と、全ゲート数、depth をヒストグラムで比較している。ヒストグラムでは Original circuit、t|ket⟩ のみで最適化、AQCEL のみで最適化、t|ket⟩ と AQCEL の順番で 2 回最適化した結果を載せている。ゲート数はゲートエラーの積み重なりの指標となる。Depth とは量子回路中で最も長くゲートが作用している量子ビットにおけるゲート数であり、脱励起などの指標となる。t|ket⟩ は CNOT はほとんど減らせていないが、単一量子ビットゲートは多く減らせていることが分かる。これは t|ket⟩ がこの時点では初期状態に依存した最適化が実装されていなかった事が原因である。それに対して AQCEL は単一量子ビットゲートだけでなく非常に多くの CNOT も減らせている事がわかる。これは t|ket⟩ は削除できなかった不要なビット制御を多く削除できたからである。t|ket⟩ には前述した ZX-calculus の一部も実装されている。よって将来的には初期状態に依存して CNOT のビット制御を削除できる copy rule も実装されたとしても、(3.47) で示したようにトフォリから直接ビット制御を削除しない限り、AQCEL と同等の不要な制御操作の削除は難しい。t|ket⟩ と AQCEL

[*2] 10 個のパスは *EulerAngleReduction(OpType.Rz,OpType.Rx)*, *RemoveRedundancies*, *GuidedPauliSimp*, *SquashHQS*, *FlattenRegisters*, *OptimisePhaseGadgets*, *KAKDecomposition*, *USquashIBM*, *CliffordSimp*, *FullPeepholeOptimise*。*RebaseIBM*, *CommuteThroughMultis* の 2 つのパスについては最初に適用している。`https://cqcl.github.io/pytket/build/html/passes.html` を参照。

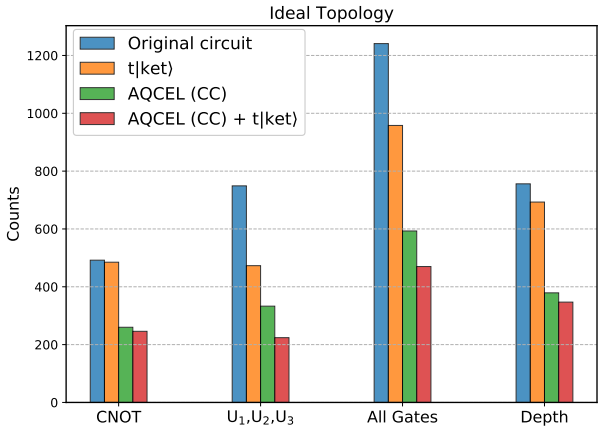



図 3.3　2 ステップの QPS 回路の最適化後と元々の回路を基本ゲートに分解した時の $(U_{1,2,3})$ と CNOT のそれぞれのゲート数とそれらの総和である全ゲート数、depth。ビット列の測定は古典シミュレーション、量子ビットは全結合を仮定している。

を組み合わせた場合はお互いを相補する形になるので最もゲート数を減らすことに成功している。しかし t|ket⟩ は全結合を仮定した最適化を行ってしまうので、超伝導量子コンピュータのような量子ビット同士の結合に制限がある場合は今回の全結合を仮定した理想的な結果と異なる結果が現れることを 3.4.3 節で議論する。

AQCEL では不要な制御操作削除と連続するゲートペアの削除という 2 つのゲート削除プロトコルが実装されているが、最もゲート削減に寄与したのは不要な制御操作である。ここでのゲート数は最適化後に基本ゲートに分解した後のゲート数とする。最初は (バリアと測定を除いて)1241 ゲートあった Original circuit は 1 回目の連続するゲートペアの削除で 132 ゲート、不要な制御操作削除で 510、2 回目の連続するゲートペアの削除で 6 ゲートの削減の内訳である。計算時間は連続する 2 回のゲートペアの削除が全体の 98% を占めており、不要な制御操作は 1% であった。ただし計算量のスケールとして、量子ビット数が大きくなるほど不要な制御操作削除が占める割合が大きくなっていく。

量子ビット数は AQCEL によって 24 から 21 まで減少した。削除された量子ビットは $n_a$、$n_b$、$n_\phi$ の 3 つのレジスタから各一つずつである。これらの量子ビットは本来 $N_{\text{evol}} \geq 3$ ステップにのみ必要であるが、インプットとして与えられた回路には用いられていたため、削除された。

### 3.4.3　$N_{\text{evol}} = 1$ ステップ

この節では QPS の 1 ステップシミュレーション回路に対する AQCEL による回路最適化について議論する。初期状態は $|f_1\rangle$、結合定数は $g_1 = 2$、$g_2 = g_{12} = 1$ とした。よってこの場合は $f \to f'\phi$ だけでなく $\phi \to f\bar{f}$ の反応も起きる汎用的な量子回路となる。QPS の 1 ステップシミュレーションについては、ビット列の測定を古典シミュレーションと *ibmq_sydney* の両方で行い、それぞれの回路最適化の結果を調べた。QPS のパラメータは全て 3.4.2 と同じである。

まず 2 ステップの時と同様にビット列を古典シミュレーションで測定し最適化したところ、最初は 472 ゲートあった Original circuit が 1 回目の連続するゲートペアの削除で 10 ゲート、不要な制御操作削除

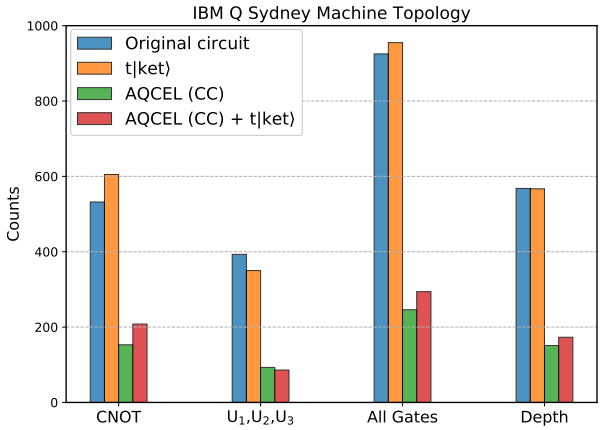



図 3.4　1 ステップの QPS 回路の最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の ($U_{1,2,3}$) と CNOT のそれぞれのゲート数とそれらの総和である全ゲート数、depth [57]。ビット列の測定は古典シミュレーションである。

で 346、2 回目の連続するゲートペアの削除で 2 ゲートの削減の内訳である。計算時間は連続するゲートペアの削除プロセスで 97% を示していたが、不要な制御操作削除プロセスは 2 ステップの時に比べて 1/3 の時間しかかかっておらず、 3.3 節で議論した計算量と同じような挙動を示している。量子ビット数も 15 から 13 に減少した。一つは多量子ビットゲートを分解するための 4 つの作業ビットのうちの 1 つであったが 1 ステップでは 3 つで十分であったので削除された。もう片方の $n_\phi$ は初期状態が $|\phi\rangle$ である場合のみに必要であったので削除された。このように最適化された回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルを行った結果を図 3.4 で示している。

AQCEL が 2 ステップの時に比べて 1 ステップの時の方がよりゲート削減割合が高いことがわかる。これは浅い回路の方が深い回路よりも、制御ゲート直前の量子状態において振幅が 0 である基底の割合が高いからである。これは回路の前半の方が後半よりも振幅が 0 である基底の割合が高いため不要な制御操作削除が多く行われるということでもある。

t|ket⟩ に関しては図 3.3 では Original circuit よりもゲート数を減らせているのに対して、図 3.4 では Original circuit よりも CNOT 数が増加してしまっていることが分かる。 [*3] これは t|ket⟩ が量子ビット間の全結合を仮定して最適化を行うからである。全結合を仮定した最適化手法は色々な量子ビット間における CNOT を多く使ってしまう。超伝導量子コンピュータのように量子ビット間の結合が限られている場合は、色々な量子ビット間における CNOT を再現するために量子ビットの入れ替えに相当する SWAP が多く必要となる [29]。 [*4]swap は CNOT を 3 つ必要とするため、t|ket⟩ を適用した後に実機のトポロジーを考慮したトランスパイルを行うと CNOT 数が増えてしまう。 [*5]それに対して AQCEL では元々の制御ゲートから不要な制御操作を削除するだけなので、必要な量子ビット間の結合が減ることはあっても

[*3] Original circuit と t|ket⟩ によって最適化された量子回路はどちらも 3.4.1 節にあるように level 3 pass manager を用いて *ibmq_sydney* のトポロジーを考慮してトランスパイルを行っている。t|ket⟩ が提供するトランスパイルは今回の実験では使用せず、最適化のみを適用している。

[*4] イオントラップ型量子計算機などの全結合型量子計算機ではこのような問題は生じない可能性がある。

[*5] 当然ながらトランスパイルした後に t|ket⟩ を適用しても実機では使えない量子ビット間の CNOT が必要になり再度トランスパイルする必要がある。

決して増えることはないので SWAP の増加はないため、そういう意味でも全結合型の最適化手法に対して優位性がある。

ここからはビット列を *ibmq_sydney* で測定することを考える。 3.2 節で議論したように、ビット列を量子計算機で測定すると多項式計算量で最適化が可能になる代わりに、ノイズの影響を受けてしまって、本来観測されないはずのビット列も観測されてしまう。よって 2.3.4 で紹介した測定エラー緩和と CNOT エラー緩和を導入している。AQCEL では量子エラー緩和のための実験において 1 回路あたり 8192 回の実験を繰り返して確率分布を求めている。ZNE においては CNOT の数を 3 倍にして外挿を行う。しかし量子エラー緩和を行っても完全にノイズの影響をキャンセルすることは難しいため、少ないカウント数であろうノイズ由来のビット列を棄却するための閾値を設定する必要があり、本研究では一定の閾値である $s_\epsilon^f$ と制御ゲート直前までの総ゲートエラーを予測した閾値 $s_\epsilon^{\mathrm{low}}$、$s_\epsilon^{\mathrm{mid}}$、$s_\epsilon^{\mathrm{high}}$ の計 4 種類の閾値を設定したのだった。AQCEL では測定されたビット列の全体に対する割合が閾値を超えていなければ、本来は測定されないはずだがノイズによって測定されてしまったノイズ列として棄却する。この棄却が正しければ正しい制御操作削除が行われ、ノイズ列を棄却できなければ何も行われず、まちがって本物のビット列を棄却してしまうと、ゲート数は減るけれども回路の等価性を失うことになる。例えば $s_\epsilon^{\mathrm{high}}$ は最も高い閾値であるため、回路の後半ではほぼ全ての制御ゲートを削除してしまう。これは回路の後半では積み重なったゲートエラーによってほぼ完全に意味のない計算が行われるよりは意味のある計算ができる範囲まで制御ゲートを削除してしまうと解釈することができるが、回路の等価性を保つ量子回路最適化という意味では正しい挙動とは言えない。そのため、量子回路の等価性と実機上での結果の正しさ間におけるトレードオフを考慮する必要がある。また回路の後半で全ての制御ゲートを削除してしまうことを防ぐために、閾値がある一定の値以上になった場合はそこで閾値の上昇を打ち切る手法も適用している。

図 3.5 を見ると $s_\epsilon^{\mathrm{low}}$ から $s_\epsilon^{\mathrm{high}}$ へと閾値が高くなるにつれてゲート数が減少している。特に $s_\epsilon^{\mathrm{mid}}$ と $s_\epsilon^{\mathrm{high}}$ では殆どのゲートが削除されていることが分かる。閾値に対して 0.2 以上になった場合そこで閾値の上昇を打ち切り、0.2 で固定する $s_\epsilon^{\mathrm{mid}} \leq 0.2$ と $s_\epsilon^{\mathrm{high}} \leq 0.2$ では大幅なゲート数の減少が抑えられている。図 3.6 を見るとここでも同様に閾値が高くなるにつれてゲート数が減少している。図には記載されていないが $s_\epsilon^f = 0.3$ においてはほぼ全てのゲートが削除された。

量子ビット数は $s_\epsilon^{\mathrm{low}}$、$s_\epsilon^{\mathrm{mid}}$、$s_\epsilon^{\mathrm{high}}$ とそれぞれに対して 0.2 の上限をかけた場合の 6 種類閾値全てにおいて 15 から 13 に減少した。$0.05 \leq s_\epsilon^f \leq 0.2$ では 15 から 13 減少したが、$s_\epsilon^f = 0.3$ では 8 まで減少した。これは非常に多くの制御ゲートが削除されたため、量子ゲートが作用しない量子ビットが生じたからである。

今まではゲート数の減少を評価してきたが、量子回路の等価性や実機での結果の改善を議論しなければならない。本来最適化前後の量子回路間の等価性を正確に評価するためには、量子回路全体を一つのゲートとみなして、量子プロセストモグラフィによりゲートフィデリティを求めなければならない。しかし AQCEL は初期状態に依存する最適化手法だから、初期状態 $|00...0\rangle$ に最適化前後の量子回路が計算された後の量子状態間のフィデリティを求めれば十分である。これは量子状態トモグラフィで達成できる。ただし殆どの量子アルゴリズムが最終的な位相の状態は無視しているので、実際は Z 基底での測定結果のみを比較すれば良い。これは QPS でも同様である。よって最適化前後の量子回路への Z 基底の測定による

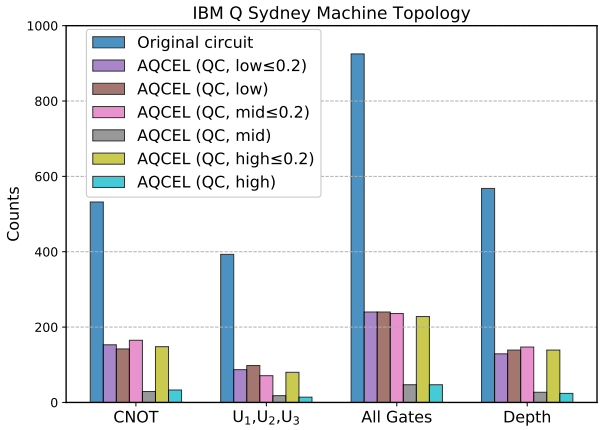



図 3.5　1 ステップの QPS 回路の最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の $(U_{1,2,3})$ と CNOT のそれぞれのゲート数とそれらの総和である全ゲート数、depth [57]。ビット列の測定は *ibmq_sydney* で行われ、閾値は $s_\epsilon^{\mathrm{low}}$、$s_\epsilon^{\mathrm{mid}}$、$s_\epsilon^{\mathrm{high}}$ とそれぞれに対して 0.2 の上限をかけた場合の 6 種類である。

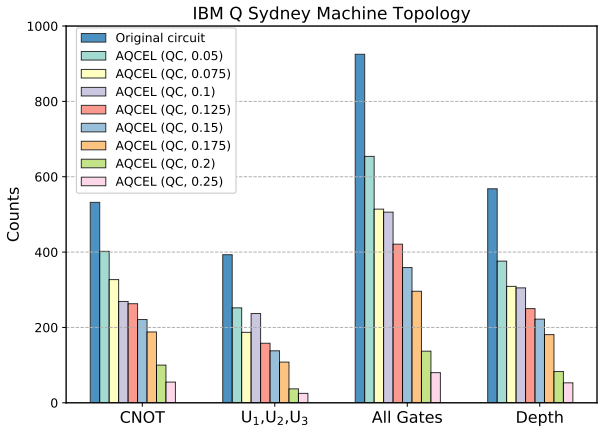



図 3.6　1 ステップの QPS 回路の最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の $(U_{1,2,3})$ と CNOT のそれぞれのゲート数とそれらの総和である全ゲート数、depth [57]。ビット列の測定は *ibmq_sydney* で行われ、閾値は $s_\epsilon^f$ である。

古典的な確率分布間の古典忠実度 $F$ の比較で定量的に評価することができる。

$$F = \sum_k \sqrt{p_k^{\mathrm{orig}} p_k^{\mathrm{opt}}}, \tag{3.49}$$

$p^{\mathrm{orig}}$ と $p^{\mathrm{opt}}$ はそれぞれ最適化前後の量子回路の Z 基底による測定された確率分布である。$p_k^{\mathrm{orig}}$ と $p_k^{\mathrm{opt}}$ はビット列 $k$ が測定された確率である。なお、$p^{\mathrm{orig}}$ に関しては理想的な確率分布として $p^{\mathrm{opt}}$ と比較するためのものなので、最適化前の量子回路に対して state vector シミュレーションを用いて得られた正確な確率分布を用いる。$p^{\mathrm{opt}}$ に関しては後述のように用途に応じて古典シミュレーションまたは量子計算機で計算する。$F = 1$ は最適化前後の量子回路が位相を無視すれば等価であることを示す。逆に 1 から最小値 0 に近づくほど等価性を失っていることを示す。以降この節では古典フィデリティのことを単にフィデリ

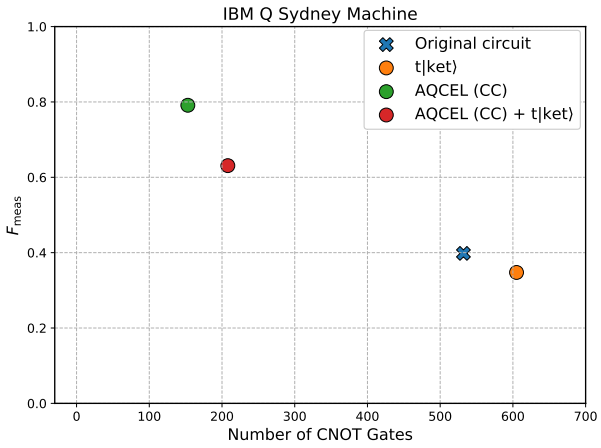



図 3.7　$F_{\mathrm{meas}}$ と 1 ステップの QPS 回路の最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の CNOT 数の散布図 [57]。ビット列の測定は古典シミュレーションで行なった。$F_{\mathrm{meas}}$ は *ibmq_sydney* を用いて測定された。

ティと呼ぶ。

本研究ではフィデリティを用いて二つの量を定義する。まず最適化後の量子回路を古典計算機による state vector シミュレーションで得られた理想的な $p^{\mathrm{opt}}$ を用いて $F_{\mathrm{sim}}$ を計算する。これによってノイズの影響を受けず量子回路の等価性を評価することが出来る。次に実機での結果の改善を測るために、最適化後の量子回路を量子計算機で計算して得られた $p^{\mathrm{opt}}$ を用いて $F_{\mathrm{meas}}$ を計算する。これは量子回路自体の等価性を評価するというよりも、AQCEL によるゲート削減によってどれだけ実機上での結果がよくなったかを測るための指標である。

$F_{\mathrm{meas}}$ については例え最適化前後の回路が等価であったとしてもノイズの影響によって $F_{\mathrm{meas}} < 1$ となる。特にエラー率の高い CNOT 数が多いと $F_{\mathrm{meas}}$ はより小さくなる。よって基本的には閾値を高くして多くの制御操作を削除して CNOT 数を抑えることができれば、ノイズの影響が小さくなり $F_{\mathrm{meas}}$ は大きくなる。しかし、閾値が高すぎて低い振幅をもつ本物の基底まで棄却し制御ゲートを削除し過ぎてしまうと、回路の等価性を大きく失ってしまい $F_{\mathrm{meas}}$ が小さくなってしまう。したがってこのトレードオフを考慮しながら最適な閾値を求める必要がある。

図 3.7 はビット列の測定を古典シミュレーションで行なった場合の最適化前後の回路の $F_{\mathrm{meas}}$ と CNOT 数の散布図である [*6]。CNOT 数が減少するにつれて、予想通り $F_{\mathrm{meas}}$ が改善されていることが分かる。Original circuit よりも t|ket⟩ によって最適化された時の方が $F_{\mathrm{meas}}$ の値が低い理由は、前述の通り全結合を仮定したことによる SWAP の増加である。$F_{\mathrm{sim}}$ については図に示されていないが、古典シミュレーションでビット列を測定しているので回路の等価性は保証され、$F_{\mathrm{sim}} = 1$ である。$F_{\mathrm{meas}}$ を得るため、1 プロット当たり 81920 回実験を繰り返した。なお $F_{\mathrm{meas}}$ の測定の際には量子エラー緩和は用いられていない。

図 3.8 では最適化におけるビット列を textitibmq_sydney で測定し、閾値は $s_\epsilon^f$ を適用した時の結果が示されている。左図を見ると CNOT 数が減少するにつれて途中までは $F_{\mathrm{meas}}$ が改善されているが、

[*6] $F_{\mathrm{meas}}$ と $U_{1,2,3}$ を含む全ゲートとの散布図もほぼ同じ傾向を示したが、最もエラー率の高い CNOT の寄与が殆どである。

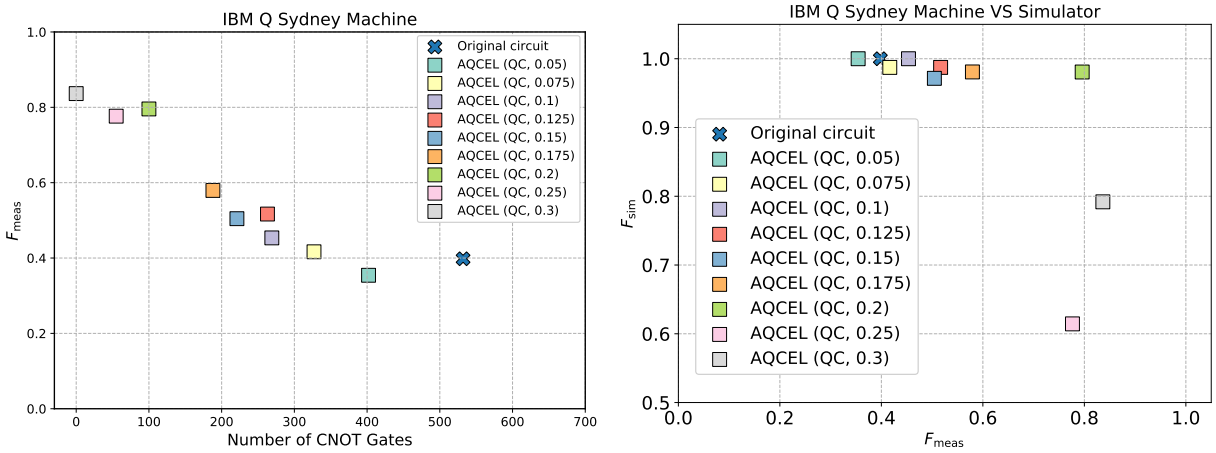



図 3.8　左図は $F_{\mathrm{meas}}$ と 1 ステップの QPS 回路の最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の CNOT 数の散布図 [57]。右図は $F_{\mathrm{meas}}$ と $F_{\mathrm{sim}}$ の散布図。最適化におけるビット列は textitibmq_sydney を用いて測定され、閾値は $s_\epsilon^f$ が適用された。$F_{\mathrm{meas}}$ は *ibmq_sydney* を用いて測定された。

$s_\epsilon^f = 0.2$ 付近でそれ以上改善されにくくなっていることが分かる。右図を見ると $F_{\mathrm{sim}}$ は $s_\epsilon^f = 0.2$ まではほぼ 1 を保っているが、$s_\epsilon^f > 0.25$ から小さくなり始めている。これは高い閾値により低い振幅を持つ本物の基底がノイズ由来として棄却されて回路の等価性を失っていることを意味する。等価性の減少が $F_{\mathrm{meas}}$ を下げることに寄与し、CNOT エラーの抑制と回路の等価性のトレードオフにより $s_\epsilon^f = 0.2$ 付近で $F_{\mathrm{meas}}$ が改善されてにくくなることが分かる。以上より、回路の等価性を保ちつつ実機での結果を改善するための量子回路最適化という意味では $s_\epsilon^f$ の中では $s_\epsilon^f \sim 0.2$ が最適である。

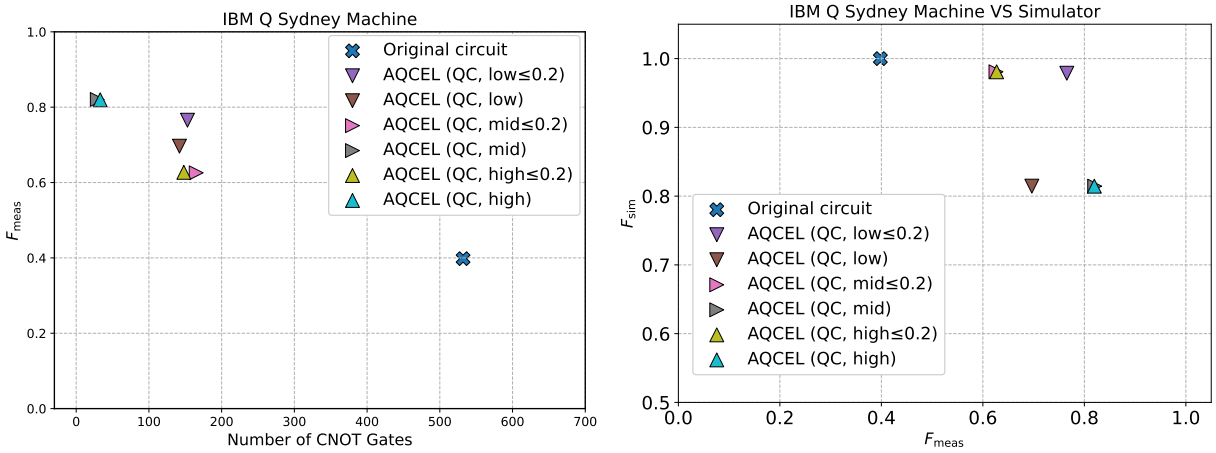



図 3.9　左図は $F_{\mathrm{meas}}$ と 1 ステップの QPS 回路の最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の CNOT 数の散布図 [57]。右図は $F_{\mathrm{meas}}$ と $F_{\mathrm{sim}}$ の散布図。最適化におけるビット列は textitibmq_sydney を用いて測定され、閾値は $s_\epsilon^{\mathrm{low}}$, $s_\epsilon^{\mathrm{mid}}$ and $s_\epsilon^{\mathrm{high}}$ とそれぞれに上限 0.2 を与えた時の 6 種類を適用した。$F_{\mathrm{meas}}$ は *ibmq_sydney* を用いて測定された。

図 3.9 最適化におけるビット列を textitibmq_sydney で測定し、閾値は $s_\epsilon^{\mathrm{high}}$、$s_\epsilon^{\mathrm{mid}}$、$s_\epsilon^{\mathrm{low}}$ とそれぞれに

上限 0.2 を与えた時の結果が示されている。右図を見ると $F_{\text{meas}}$ は閾値が高くなるほど改善される傾向があるが、左図を見ると逆に $F_{\text{sim}}$ が低くなる事が分かる。そこで上限 0.2 を与えると $F_{\text{meas}}$ は下がっているが、回路の後半における制御ゲートの大幅な削除を抑制し、$F_{\text{sim}}$ を高く保つことを可能にしている。面白いケースとして $s_\epsilon^{\text{low}}$ と $s_\epsilon^{\text{low}} \leq 0.2$ を比較すると、CNOT 数は $s_\epsilon^{\text{low}} \leq 0.2$ が増えているが、$F_{\text{sim}}$ と $F_{\text{meas}}$ の両方で上回っている。これは CNOT 数の減少よりも回路の等価性の寄与が大きくなったからであると推測できる。

今まで見てきた様々な最適化の結果を図 3.10 と 3.11 にまとめている。まず AQCEL による最適化は Orginal circuit と t|ket⟩ による最適化と比べて、ほぼ全てのケースで $F_{\text{meas}}$ を改善した [*7]。これは Qiskit や今回用いたバージョンの t|ket⟩ では削除できない不要な制御操作の削除によるものである。また最適化の中でビット列を量子計算機で測定する場合は、低い振幅を持つ基底を棄却することで制御操作が余分に削除され一定の等価性を失うケースが多かった。しかし失う回路の等価性が大きくなければ、CNOT 数を減らすことの寄与の方が大きく、$F_{\text{meas}}$ の改善に寄与する場合もあった。これが AQCEL による低振幅基底の棄却によって有効な近似回路が生成される一例である。

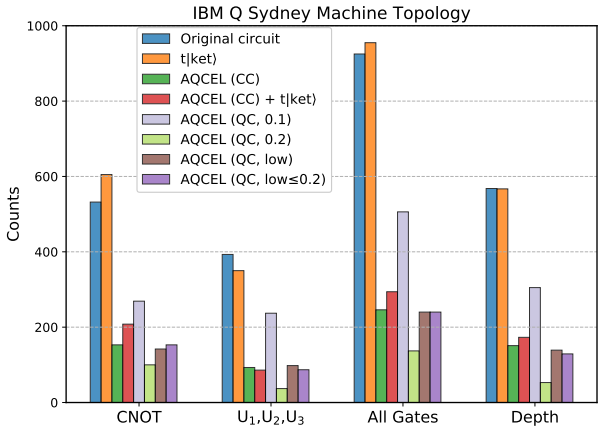



図 3.10　1 ステップの QPS 回路の様々な最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の ($U_{1,2,3}$) と CNOT のそれぞれのゲート数とそれらの総和である全ゲート数、depth [57]。

この面白い性質が与えた NISQ ならではの結果として、*ibmq_sydney* を用いてビット列を測定したノイズの影響のある AQCEL (QC,0.2) や AQCEL (QC,0.3) が、古典シミュレーションを用いてビット列を測定した理想的な最適化結果である AQCEL (CC) に対して実機上での結果 $F_{\text{meas}}$ では上回っていることである。この結果は現状の NISQ では CNOT エラーの影響が大きいため、回路の等価性を少々失っても CNOT 数を減らすことが実機上での結果を改善するという面白い性質を示唆している。例えばグローバーの探索アルゴリズム [6] や量子機械学習 [28] ではパートンシャワー量子アルゴリズムと違って、初期状態 $|0\rangle^{\otimes n}$ に対してアダマールゲート $H^{\otimes n}$ をかけるため Z 基底上では全ての基底の振幅が 0 より大きいことになり、現状の AQCEL は何も最適化することができないが、一部の基底の振幅が非常に小さけれ

[*7] 前述した通り、Orginal circuit も含めた全ての量子回路は、様々な最適化オプション含まれている Qiskit の level 3 pass manager を用いてトランスパイルを行っている。

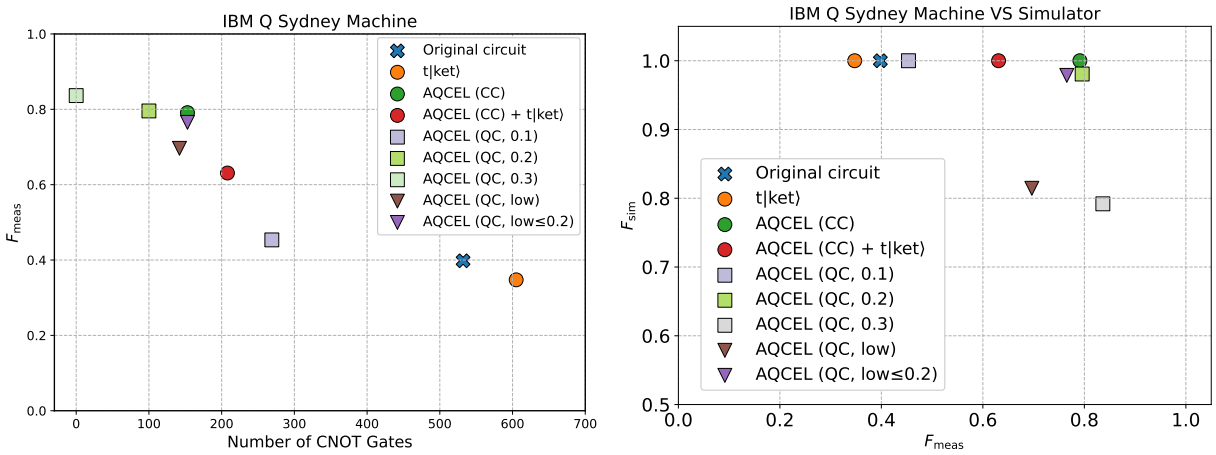



図 3.11　左図は $F_{\mathrm{meas}}$ と 1 ステップの QPS 回路の様々な最適化後と元々の回路を *ibmq_sydney* のトポロジーを考慮してトランスパイルした後の CNOT 数の散布図 [57]。右図は $F_{\mathrm{meas}}$ と $F_{\mathrm{sim}}$ の散布図。$F_{\mathrm{meas}}$ は *ibmq_sydney* を用いて測定された。

ばその影響は小さいので棄却して有効な近似回路を生成し、ビット列を大幅に削って CNOT のエラーを抑えることで実機上での結果 $F_{\mathrm{meas}}$ を改善する可能性があることを示唆する。なおこの性質は一部の計算基底が用いられるような量子アルゴリズムに限定されず、任意の量子アルゴリズムに対して有効である。

将来ゲートエラーが低くなり誤り訂正も出来るようになればこの性質は失われてしまうが、AQCEL の本来の目的は回路の等価性を保ちながら不要な制御操作を減らすことであった。例えば AQCEL (QC,low$\leq$ 0.2) などが最も理想的な結果 AQCEL (CC) に近づいていくことができる。つまり量子誤り訂正が出来るようになっても、ある量子回路を繰り返し使いたい場合などでは、一度 AQCEL (QC) で多項式計算量で理想的な不要な制御操作を行い、それ以降は最適化された回路を繰り返し用いることで総合的なコストを下げることができる。

# 第 4 章

# 量子トリットにおける制御ゲート実装

本章では多量子ビットゲートを量子トリットゲートを用いた実装手法を紹介する。 4.1 節でトフォリゲートの量子トリット上での様々な分解方法、 4.2 節では制御ビットの状態が量子トリットの取り得る任意の状態 $|0\rangle$、$|1\rangle$、$|2\rangle$ のいずれであっても正しく作用する二量子トリットゲート実装の戦略、 4.3 節ではそのような二量子トリットゲートのパルスレベルにおける実装手法を紹介する。

## 4.1 Toffoli の分解方法

量子ビットにおいてトフォリを分解しようとすると少なくとも 6 個の CNOT が必要である [4]。3 個の CNOT で構成可能だが一部の相対位相が $\pi$ ズレるトフォリを相対トフォリと呼ぶ。多量子ビットゲートをトフォリで分解する際は、作業ビットをリセットするために同じトフォリを 2 回作用させるという特徴があり、分解のために相対トフォリを用いても相対位相のズレが $2\pi$ になってキャンセルされるため、CNOT のコストを約半分にすることが可能である [62]。制御ビット数が $n$ 個である多量子ビットゲートを、$n-2$ 個の作業ビットと相対トフォリを用いて最も少ない CNOT 数で量子ビット上で分解する場合でも、$6n-6$ 個の CNOT と 1 個の CU を必要とする。

それに対して量子トリット上でトフォリゲートを分解すると相対位相のズレなしに 3 つの CNOT のみで分解することが可能である [40, 41, 42, 43, 44, 45, 46]。また多量子ビットゲートをツリー型に分解するならば [41, 43]、$|2\rangle$ を作業ビットの代わりに用いるため作業ビットは不要であり、ゲート数も $2n-2$ 個の CNOT と 1 個の CU を必要とし 3 分の 1 のコストカットになるため、ゲート数と depth のみの観点から見れば完全な上位互換となる。また通常のトフォリは 3 つの量子ビット結合を必要とし SWAP が生じるが、量子トリットを用いたトフォリでは 2 種類の量子ビット結合しか要求しないため実質 SWAP は生じない。

このような量子トリット上のトフォリを実際の量子計算機で実装しようとする場合は、高いゲート忠実度の量子トリットゲートの実装が不可欠である。そのような候補は限られており [33, 42, 53]、本研究では実現可能性の高い基本ゲートのみを用いた様々な分解手法を発案した [43]。図 4.1 における右上にある二つの基底間における $X$ ゲートを用いている。これらはパルスの振動数をそれぞれのエネルギー準位間の振動数に設定し、x 軸周りに $\pi$ 回転するような振幅掃引によって実現する。二量子トリットゲートは CR 相互作用により実現可能する。特に $X_{01}$ もしくは $X_{12}$ のみから構成される B1、B3、C2 のトフォリ

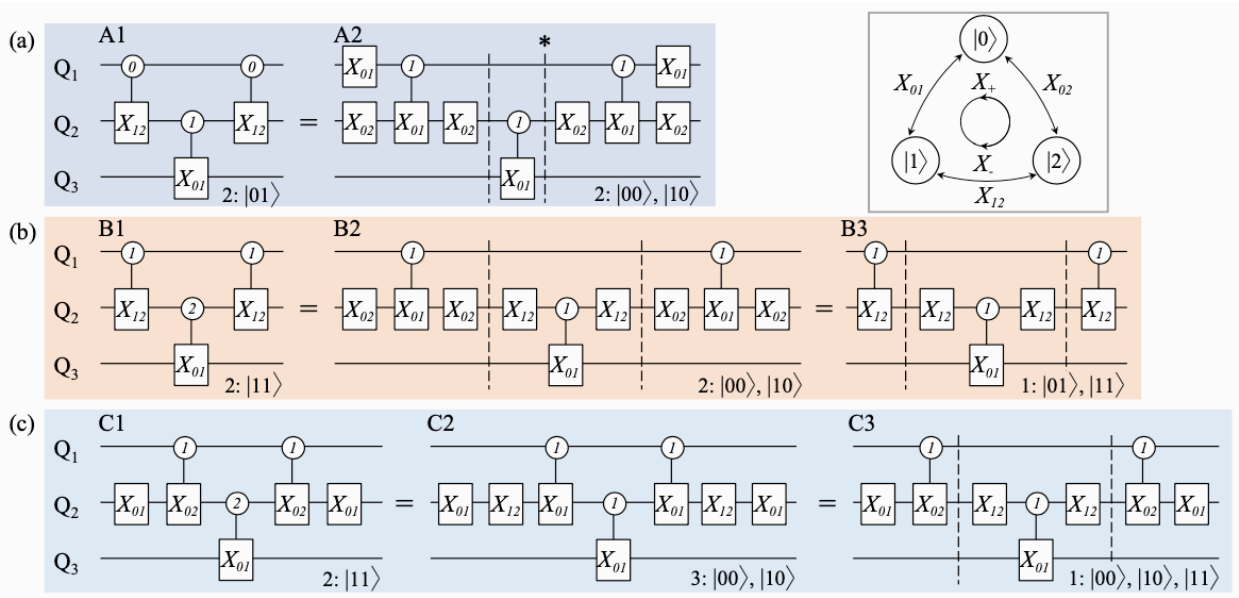



図 4.1　量子トリットにおけるトフォリゲートの様々な分解方法。IBM Quantum で実現できる可能性が高い基本ゲートのみを用いて分解している [43]。

ゲートの実現を目標としている。

量子トリットで $|2\rangle$ として扱われている第二励起状態は第一励起状態 $|1\rangle$ に比べて charge dispersion が数十倍ほど大きい [54, 55]。$T_2$ は charge dispersion に反比例するため $|2\rangle$ における相対位相はすぐに緩和してしまう。よって $|2\rangle$ をもつ基底の振幅を出来るだけ小さくすること、量子トリットとして扱っている時間を短くすることが重要である。それを満たすのは B1 と B3 トフォリであり、将来的にはこれらが最も効率の良いトフォリになる可能性がある。ただし本研究ではまず、新たに実装する必要があるゲート数が少ない C2 トフォリの実装を行うことで、今後の研究への足がかりとする。

## 4.2　二量子トリットゲート

まず図 4.1 にある全ての分解手法において、それぞれの中央にある CNOT では制御ビットの状態が $|0\rangle$、$|1\rangle$、$|2\rangle$ の重ね合わせ状態になってしまう。そのため量子ビットを仮定した従来の CNOT では望まない挙動を示す [53, 42, 54]。

二量子トリットゲートは、制御ビットが $|0\rangle$、$|1\rangle$、$|2\rangle$、標的ビットの振動が $|0\rangle - |1\rangle$ 間、$|1\rangle - |2\rangle$ 間、$|0\rangle - |2\rangle$ 間が考えられるので全部で 9 通り存在する。9 通りのうち一つでも実装ができれば、図 2.2 で見たように、制御ビットや標的ビットの前後に単一量子トリットゲートをおけば任意の二量子トリットゲートを実現できるのだった。実際に量子ビット空間においても実装されているのは $C_1X_{01}$ のみであり、$C_0X_{01}$ は制御ビットを $X_{01}$ ゲートで挟んで実現される。今回は、制御ビットが $|0\rangle$、$|2\rangle$ の時に同じ Rabi 振動をするような二量子トリットゲートとしての $C_1X_{01}$ の実装を目指す。二量子トリットゲートの実装手法としては大きく二つにあり、CR パルスを分割せず最小ゲート時間で実装可能な direct CNOT と、CR を 3 つに分割することで AC-Stark shift 由来の $ZI$ エラーを自動補正する echo CNOT である。二量

子トリットゲートの echo シークエンスを考案したのは本研究が初めてである。

量子トリット空間における CR 相互作用の詳細は未だはっきりと判明していないが、パルス振幅を大きくしていくと制御ビットが $|0\rangle$、$|2\rangle$ の時に同じ Rabi 振動をする場合が存在することが経験的に知られている [54]。これは $ZX$ 項 [*1]が単純な機構ではないことを示唆しているが、これを利用して Z 基底上では正しい CNOT を実装することができる。相対位相まで正しい操作をする二量子トリットゲートを実装するためには、ハミルトニアンの詳細を計算する必要があり今後の課題である。本研究では制御ビットが $|0\rangle$ と $|2\rangle$ の時に同じ挙動を示すパルス振幅を前提にして進める。

### 4.2.1　Direct CNOT

制御ビットが $|0\rangle$ と $|2\rangle$ の時に同じ挙動を示すパルス振幅を前提にしているので、自然に実装できるのは $C_1X_{01}$ である。さらに量子トリット空間でも CR パルスと $IX$ パルスを同時ドライブし、direct CNOT を構成する。Direct CNOT のメリットは最小のゲート時間にある。CR パルスを単一量子ビット

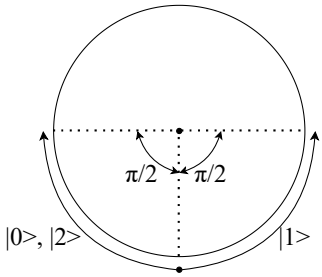



図 4.2　制御ビットが $|0\rangle$、$|2\rangle$ の時に同じ Rabi 振動をする場合。制御ビットが $|0\rangle$、$|1\rangle$、$|2\rangle$ の時の標的ビットにおける Rabi 振動の様子。回転角が $\pi/2$ の時に制御ビットが $|0\rangle$ または $|1\rangle$ の時と $|2\rangle$ の時の Rabi 振動の位相差が $\pi$ になっている。

ゲートで挟みながら複数に分割する echo CNOT よりも、direct CNOT は単一量子ビットゲート分の時間を節約できる。よってキャリブレーションした直後であれば direct CNOT の方が echo CNOT よりもエラー率が低くなる場合がある [69]。しかし実際には 2.2.4 節で見たように、時間依存性のある $ZI$ エラーが echo CNOT では自動補正されるのに対して、direct CNOT はキャリブレーションから時間が経つにつれてエラー率が高くなってしまう。量子トリットにおける direct CNOT のパルスシークエンスは、量子ビットにおける direct CNOT と同じく図 2.4 のように表される。

### 4.2.2　Echo CNOT

量子ビットにおける echo CNOT の考え方は制御ビットが $|0\rangle$、$|1\rangle$ の両方の場合に等しく $ZI$ エラーを載せることで相対位相を絶対位相にしてしまい、無視できるようにする仕組みであった。量子トリットにおける echo CNOT もこの戦略を応用する。注意しなければならないのは量子ビット空間における echo CNOT のように CR パルスを二つに分割して、制御ビットが $|0\rangle$、$|1\rangle$ に等しく $Z_{01}I$ エラーを載せても、$|2\rangle$ における相対位相の存在によって、絶対位相としてキャンセルすることが出来ないことである。例え

[*1] 量子トリット空間における CR のエンタングル項が $Z_{01}X_{01}$ だけでは実験事実を説明できない。

ば量子ビット空間において $|0\rangle$、$|1\rangle$ に等しく相対位相エラー $e^{i\phi_x}$ がかかると

$$|\psi\rangle = e^{i\phi_x}\alpha|0\rangle + e^{i\phi_x}e^{i\phi_1}\beta|1\rangle \tag{4.1}$$
$$= e^{i\phi_x}\left\{\alpha|0\rangle + e^{i\phi_1}\beta|1\rangle\right\} \tag{4.2}$$

となり、$e^{i\phi_x}$ は絶対位相としてキャンセルされるが、量子トリット空間では

$$|\psi\rangle = e^{i\phi_x}\alpha|0\rangle + e^{i\phi_x}e^{i\phi_1}\beta|1\rangle + e^{i\phi_2}\gamma|2\rangle \tag{4.3}$$
$$= e^{i\phi_x}\left\{\alpha|0\rangle + e^{i\phi_1}\beta|1\rangle\, e^{i(\phi_2-\phi_x)}\gamma|2\rangle\right\} \tag{4.4}$$

となり、$e^{i\phi_x}$ が絶対位相としてキャンセルされないことが分かる。

よって量子トリットで echo CNOT を構成するためには $|0\rangle$、$|1\rangle$、$|2\rangle$ に等しく $ZI$ エラーをかけるため、少なくとも CR パルスを 3 つに分割する必要がある。また $ZI$ エラーが等しく作用するためには、分割された 3 つの CR パルスそれぞれにおいて制御ビットの基底変換を行い、$|0\rangle$、$|1\rangle$、$|2\rangle$ の状態にある時間を等しくする。これらを満たす基底変換は図 4.1 の右上にある $X_+$ もしくは $X_-$ である。これらは

$$X_+ = X_{01}X_{12} \tag{4.5}$$
$$X_- = X_{12}X_{01} \tag{4.6}$$

と定義される。これを用いた量子トリットにおける echo CNOT のパルスシークエンスは図 4.3 の通りである。なお、図 4.3 に示したパルスシークエンスが $C_1X_{01}$ になるという挙動を図 4.4 において説明した。

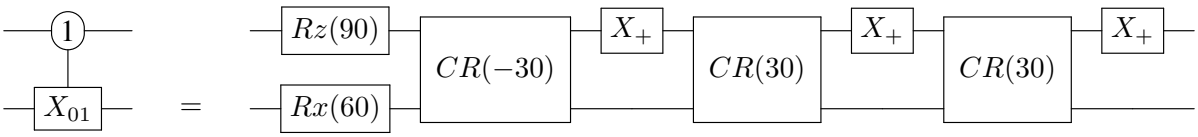



図 4.3　量子トリットにおける echo CNOT の構成方法。

この echo CNOT は CR パルスで生じる AC-Stark Shift 由来の $Z_{01}I$ と $Z_{12}I$ エラーを自動で補正すると

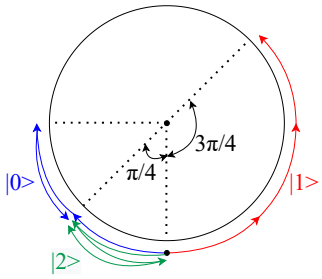



図 4.4　図 4.3 に示したパルスシークエンスにおける、制御ビットが $|0\rangle$、$|1\rangle$、$|2\rangle$ の時の標的ビットにおける Rabi 振動の様子。

いう非常に大きなメリットが存在する [*2]。これは $X_+$ による 3 回の基底変換によって、CR において生じる $ZI$ エラーが $|0\rangle$、$|1\rangle$、$|2\rangle$ に等しく作用し、これらを絶対位相としてキャンセルできるからである。

[*2] これはあくまで非可換由来の誤差を無視した場合である。非可換由来の誤差の詳細な影響はこれから調べる必要がある。

## 4.3　パルスレベルでの実装

本節では図 4.1 中の C2 トフォリの実装を試みる。そこで C2 トフォリを構成する各ゲートの、Qiskit Pulse を用いたキャリブレーション手法と一部の結果について解説する。そして最後に全てのゲートを合わせて作った新しい C2 トフォリゲートの結果と特徴について議論する。二量子トリットゲートについては direct $C_1X_{01}$ の実験結果を解説する。今回は相対位相を考慮せず、つまり Z 基底による測定結果を古典忠実度により評価した。そのため相対位相補正と量子プロセストモグラフィによるゲート忠実度の測定を今後の課題とする。また active crosstalk mitigation や rotaty echo なども導入していない。用いた量子計算機は IBM Quantum Falcon Processors の一つである 27-qubit IBM's *ibmq_toronto* device である。トフォリの制御ビットとして 14 と 11 番目の量子ビットを、標的ビットとして 8 番目の量子ビットを用いた。なお、制御ビットのうち量子トリットとして扱ったのは 11 番目の量子ビットである。

### 4.3.1　$X_{01}$ と $X_{12}$

まず Qiskit Textbook [82] に紹介されている Qiskit Pulse を用いた量子トリット空間における $X_{01}$ ゲートの実装手法について説明する。$X_{01}$ ゲートのためのパルス形はガウス関数

$$f(x) = A_{\text{amp}} e^{-\frac{(x-\text{duration})^2}{2\sigma^2}} \tag{4.7}$$

で定義している。なお duration はパルス時間幅を表していて、単一量子ビットゲートのパルス時間幅は 36 ns に固定されている。式 (2.42) にあるように、量子ゲートの回転角はパルス時間幅と振幅の面積で決まり、パルス時間幅は固定されているので、ガウシアン内の面積が $\pi$ 回転するように振幅の実数成分のキャリブレーションを行う。また量子ビットの振動数 $\omega_{01}$ のキャリブレーションも行う。これらは定期的に IBM 側でキャリブレーションされた値が公開されているので、その値周辺で再度探索しなおす。新しくキャリブレーションした量子ビットの振動数をパルスに振動数に設定した後は、パルスの振幅の掃引を行う。図 4.5 は $X_{01}$ ゲートのための量子ビットの振動数 $\omega_{01}$ 掃引の結果を示している。全ての実験において初期状態 $|0\rangle$ を準備し、適当な振幅を持つパルスの振動数を少しずつ変化させながら各振動数毎に 8192 回実験を繰り返して、それらの測定された IQ 信号の虚数の実数成分と虚数部分それぞれの平均値をプロットしている [*3]。パルスの振動数が量子ビットの振動数に近いほど $Z$ の項が小さくなり $X$ の項が大きくなって、Rabi 振動による $|1\rangle$ への遷移を起こすようになっている様子が分かる。また図 4.5 を見れば虚数成分よりも実数成分の方が $|0\rangle$ と $|1\rangle$ の区別に敏感であることがわかるので、$X_{01}$ ゲートのキャリブレーションでは実数成分を用いる。そこで次式に表されるコーシー（ローレンツ）分布

$$\frac{A}{\pi}\frac{B}{(x-\omega_{\text{freq}})^2 + B^2} + C \tag{4.8}$$

により、測定された実数成分とパルスの振動数のプロット図をフィットして最も $|1\rangle$ への遷移率が高かったピークの位置を求めると、5.116 GHz が得られる。

[*3] 量子ビットの状態に応じて異なる共振器の周波数シフトを起こす。よって共振器の周波数を持つの反射係数に位相のズレが生じ、これ測定して IQ 信号に変換している [15, 52]。

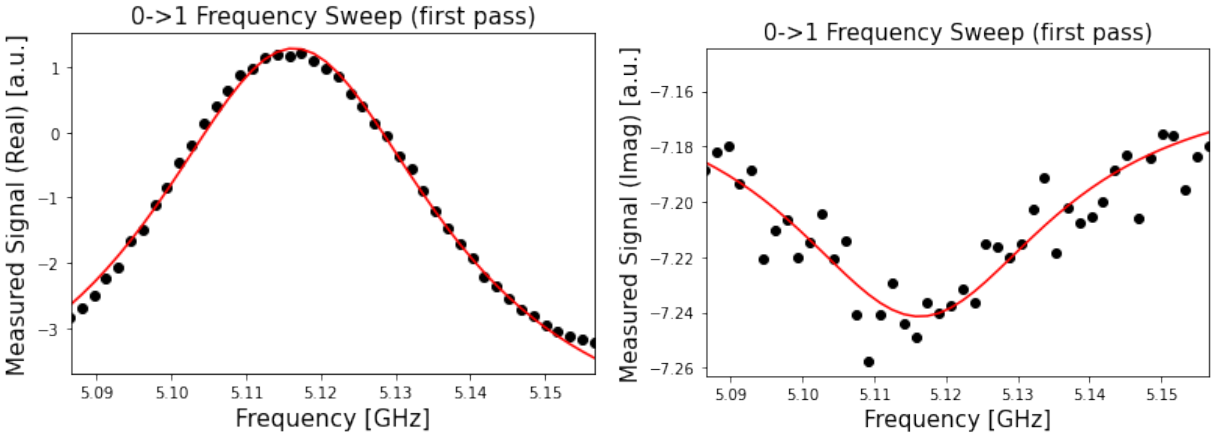



図 4.5　$X_{01}$ ゲートのための量子ビットの振動数 $\omega_{01}$ 掃引の結果。左図と右図はそれぞれ振動数を掃引した時に測定された IQ 信号のうち虚数の実数成分と虚数部分の値を示している。なおパルスの振幅は 0.1 a.u. に設定している。

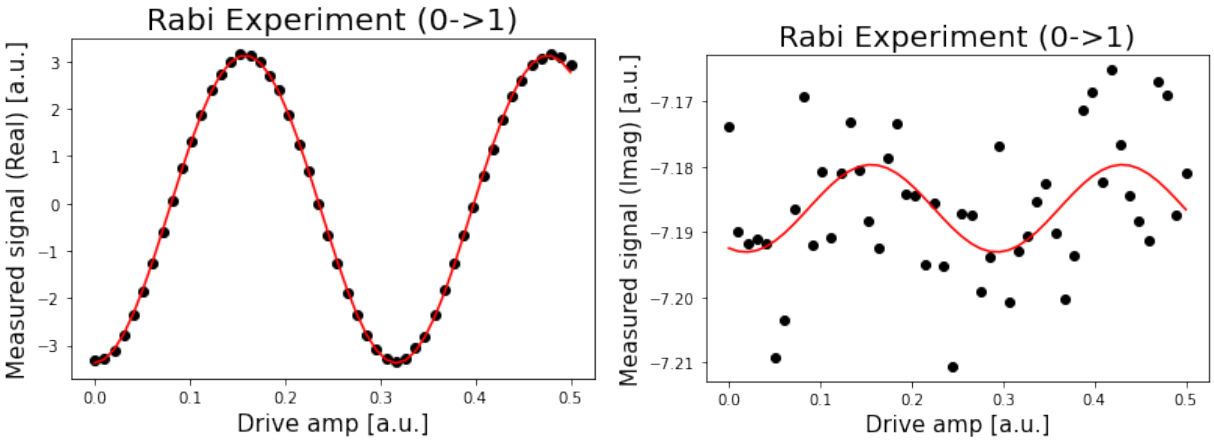



図 4.6　$X_{01}$ ゲートのためのパルスの振幅掃引の結果。左図と右図はそれぞれ振幅を掃引した時に測定された虚数の実数成分と虚数部分の値を示している。なおパルスの振動数は先にキャリブレーションして得られた 5.116 GHz に設定している。

フィットによって得られた $\omega_{01}$ を設定し、次はパルスの振幅の掃引を行う。実験のプロセスは振動数掃引と同様である。ここでは三角関数

$$A\cos\left(2\pi\frac{x}{B_{\mathrm{amp}}}\right)+C \tag{4.9}$$

によってフィットする。これも測定された実数成分のフィットするが、得られる振幅はブロッホ球を一周するような $2\pi$ 回転するものなので、これを 2 で割った結果が 0.159 a.u. である。

以上より得られたパルスの振動数と振幅を用いて $X_{01}$ ゲートを構成し、初期状態 $|0\rangle$ に対して何もせずそのまま測定した場合と $X_{01}$ ゲートを作用してから測定した結果を図 4.7 において比較している。理想的にはそれぞれ $|0\rangle$ と $|1\rangle$ が得られるはずである。初期状態 $|0\rangle$ に対して、キャリブレーションした $X_{01}$ ゲートが $|1\rangle$ に励起していることがわかる。一部図の右上の方にいくつかの測定点が見られるが、これは $|2\rangle$ であり、少し漏れが含まれていることがわかる。これは今後 DRAG パルスを用いることによって軽減

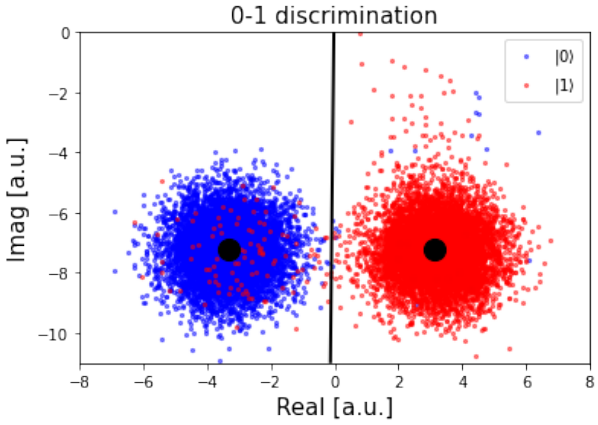



図 4.7　初期状態 $|0\rangle$ をそのまま測定した場合（青）と、キャリブレーションによって得られた $X_{01}$ ゲートを作用してから測定した結果（赤）の IQ 散布図。$|0\rangle$ と $|1\rangle$ を区別する黒線は Linear Discriminant Analysis (LDA) [83] と呼ばれる機械学習のテクニックによって得られたものである。

できる [84, 85]。

$X_{12}$ ゲートの実装手法は基本的に $X_{01}$ ゲートの場合と同様である [86]。パルスの振動数が $\omega_{12}$ であるガウシアンパルスを構成したい。例えばパルス自体の振動数は $\omega_{01}$ のままにして、パルスの形をガウシアンに $\omega_{12} - \omega_{01}$ の周波数を持つ sin 関数を乗算することで周波数変調を起こす手法があり、本研究でもこれを採用している [33, 86, 52] [*4]。図 4.8 を見れば、$|1\rangle$ と $|2\rangle$ の区別では虚数成分の方が敏感であること

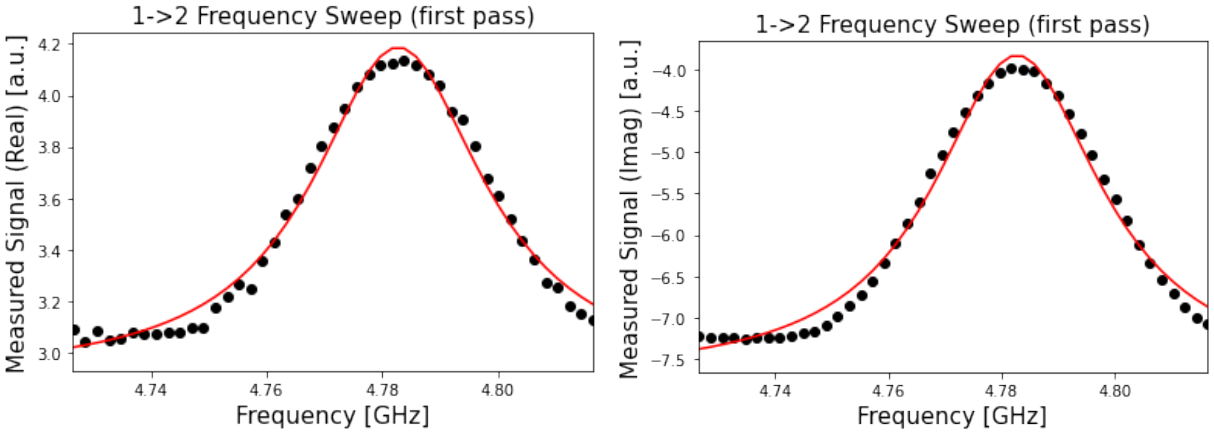



図 4.8　$X_{12}$ ゲートのための量子ビットの振動数 $\omega_{12}$ 掃引の結果。左図と右図はそれぞれ振動数を掃除した時に測定された IQ 信号のうち虚数の実数成分と虚数部分の値を示している。なおパルスの振幅は 0.2 a.u. に設定している。

がわかるので、$X_{12}$ ゲートのキャリブレーションでは虚数成分を用いる。コーシー（ローレンツ）分布により、測定された虚数成分とパルスの振動数のプロット図をフィットすると 4.783 GHz が得られる。こ

[*4] なおこのような周波数変調はパルスを生成する local oscilltor(LO) と Arbitrary waveform generator(AWG) 同士の混合の際に引き起こすことも可能であり [15]、パルスの生成前に周波数変調を起こすか、生成後に周波数変調を起こすかの違いである。

れは $\omega_{01}$ よりも小さい値であり、これは式 (2.31) と式 (2.32) から導いた $\delta < 0$ より $\omega_{12} < \omega_{01}$ という予想と一致している。振幅も図 4.8 にあるように $X_{01}$ と同様にフィットして 0.251 a.u. である。

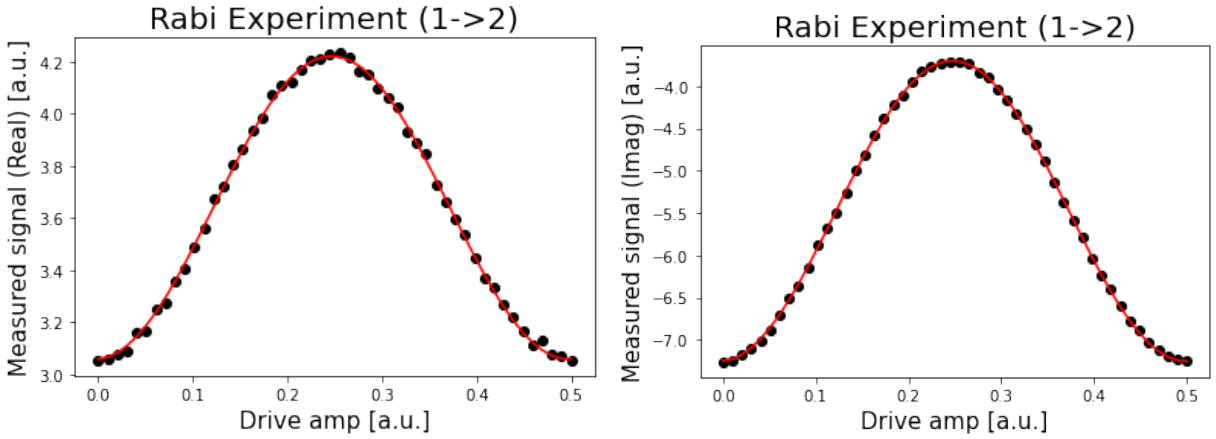



図 4.9　$X_{12}$ ゲートのためのパルスの振幅掃引の結果。左図と右図はそれぞれ振幅を掃引した時に測定された虚数の実数成分と虚数部分の値を示している。なおパルスの振動数は先にキャリブレーションして得られた 4.783 GHz に設定している。

図 4.7 と同じ実験に、初期状態 $|0\rangle$ に $X_{01}$、$X_{12}$ の順番に二つのゲートを作用させてから測定した結果も加えたものが図 4.10 であり、理想的にはそれぞれ $|0\rangle$ と $|1\rangle$、$|2\rangle$ が得られるはずである。

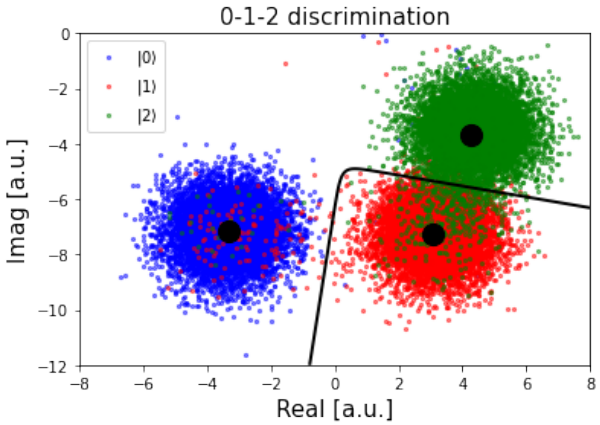



図 4.10　初期状態 $|0\rangle$ をそのまま測定した結果（青）、キャリブレーションによって得られた $X_{01}$ ゲートを作用してから測定した結果（赤）、キャリブレーションによって得られた $X_{01}$ ゲートと $X_{12}$ ゲートを順に作用してから測定した結果（緑）の IQ 散布図。$|0\rangle$ と $|1\rangle$、$|2\rangle$ を区別する黒線は LDA によって得られたものである。

### 4.3.2　二量子トリットゲート

ここでは二量子トリットゲートとしての direct $C_1X_{01}$ ゲートのキャリブレーション手法について議論する。 2.2.4 節で見たように、CR 相互作用を起こすために制御ビットを標的ビットの振動数でドライブ

する。そのパルスの振幅を掃引し、制御ビットが $|0\rangle$、$|2\rangle$ の時に同じ Rabi 振動をするような振幅を見つける。そのために標的ビットの初期状態は常に $|0\rangle$ とする。パルス振幅を掃引しながら、制御ビットがそれぞれの状態の時の標的ビットにおける Rabi 振動の周期を測定した。その結果が図 4.11 であり、フィットの結果の交点から、振幅が 0.602 a.u. の時に Rabi 振動の周期が 1347 ns で一致すると予想できる。これは元々量子ビット空間用に IBM で自動キャリブレーションされた CR パルスの振幅の 1.94 倍であり、大きなパルス振幅を要することがわかる。

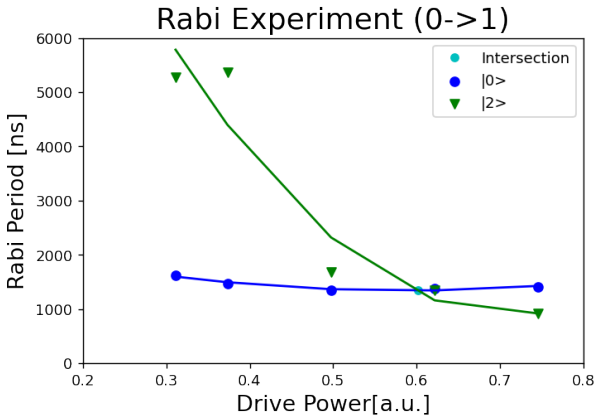



図 4.11　パルスの振幅を掃引をした時に、制御ビットが $|0\rangle$、$|2\rangle$ である場合の標的ビットにおける Rabi 振動の周期。

これで $ZX(\pi/2)$ の回転分のパルス時間幅を与えれば、図 4.2 で見たように制御ビットが $|0\rangle$、$|2\rangle$ の時と $|1\rangle$ の時の標的ビットにおける Rabi 振動の位相差が $\pi$ ずれる。しかし 4.2.1 節でも言及したように、CNOT として扱うには制御ビットが $|0\rangle$、$|2\rangle$ の時には標的ビットの状態は変化してほしくないので、事前に $Rx$ ゲートを標的ビットに作用させて、CR による Rabi 振動により制御ビットが $|0\rangle$、$|2\rangle$ の時には標的ビットがちょうど $|0\rangle$ に戻り、制御ビットが $|1\rangle$ の時の時には標的ビットがちょうど $|1\rangle$ まで移動するようにする。つまりこの場合のパルスシークエンスは図 4.12 となる。

$Rx$ ゲートの角度キャリブレーションでは制御ビットが $|0\rangle$ の場合のみを実験している。掃引した結果が図 4.13 である。フィットの結果の交点から、$Rx(1.2)$ かつ CR パルス時間幅が 206 ns の時に CNOT になると予想できる。

以上よりキャリブレーションで得られた、相対位相を無視した direct $C_1X_{01}$ を用いてベル状態を作り、これの精度を古典忠実度で評価したところ自作の direct $C_1X_{01}$ は 0.9997、IBM の default echo CNOT は 0.9492 であった。図 4.14 からエンタングルゲートとしての動作が確認できる。相対位相を無視した古典忠実度だけを見れば高い精度で CNOT の挙動を示していることがわかる。しかしこの実験だけではその他の初期状態の場合の挙動や相対位相の精度が不明なため、今後さらなる実験が必要である。

### 4.3.3　Toffoli

これまでに作成した全ゲートを用いてトフォリゲートを構成した時のパルスシークエンスが図 4.15 に示されている。量子トリットを用いた新しい C2 トフォリゲートでは総ゲート時間が 1.3 µs であり、量子

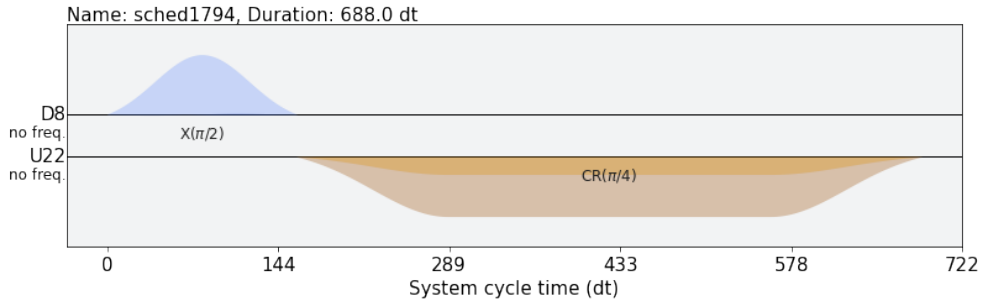



図 4.12　Direct $C_1X_{01}$ のパルスシークエンス。二つ目のパルスは CR パルスで制御ビットにドライブする。制御ビットが $|0\rangle$、$|2\rangle$ の時と $|1\rangle$ の時の標的ビットにおける Rabi 振動の位相差が $\pi$ ずれるような、CR パルスの振幅と時間幅を決定する。一つ目のパルスは $Rx$ パルスであり、CR によって制御ビットが $|0\rangle$、$|2\rangle$ の時には標的ビットがちょうど $|0\rangle$ に戻り、制御ビットが $|1\rangle$ の時の時には標的ビットがちょうど $|1\rangle$ まで移動するような角度に調整する。なお CR パルスと $Rx$ パルスは同時にドライブすることもできる。

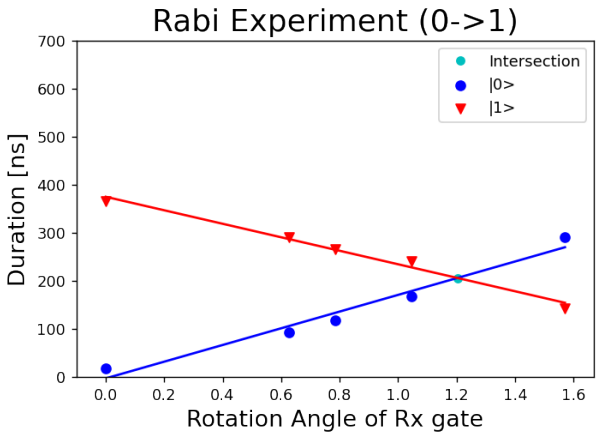



図 4.13　標的ビットに対する $Rx$ ゲートの角度を掃引をした時に、制御ビットが $|0\rangle$ 時は標的ビットが $|0\rangle$ になるまでの時間、制御ビットが $|1\rangle$ 時は標的ビットが $|1\rangle$ になるまでの時間をプロットしている。

ビットのみを用いた従来のトフォリゲートの総ゲート時間が 4.4 μs であった。量子トリットを用いることで約 3 倍のコストカットに成功していることがわかる。これは CNOT 数半減と SWAP の有無によるものであり、 4.1 節での予想と一致する。

次に初期状態として $|000\rangle$ と $\frac{1}{2}(|000\rangle + |010\rangle + |100\rangle + |110\rangle)$ を準備し、これらのトフォリを作用させた結果を Z 基底で測定し、古典忠実度により相対位相を無視した古典忠実度で評価した。なお測定においては Qiskit の measurement level 1 によって自動で $|0\rangle$ か $|1\rangle$ かを判別させたものであるため、$|2\rangle$ への漏れが含まれる場合は $|1\rangle$ として測定される [*5]。よって Qiskit の measurement level 1 による測定結果から求めた古典忠実度は $|2\rangle$ への漏れがないという前提である。

初期状態が $|000\rangle$ の場合は、量子トリットを用いた C2 トフォリでは 0.939、IBM の従来のトフォリ

[*5] 図 4.10 における $|2\rangle$ は、図 4.7 における判別線において $|1\rangle$ と判別されることがわかる。

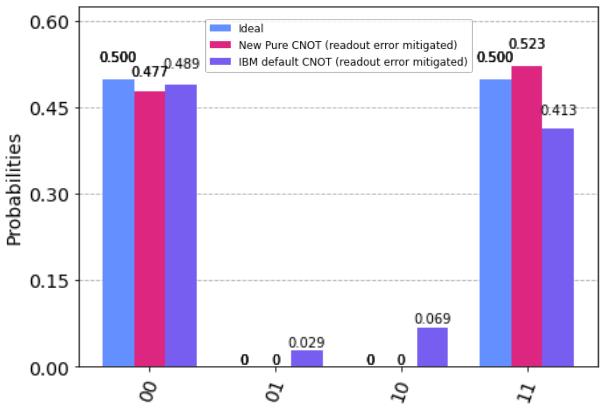



図 4.14　相対位相を無視した direct $C_1X_{01}$ とデフォルトの IBM の CNOT、それぞれを用いたベル状態を Z 基底で測定した時の古典忠実度。

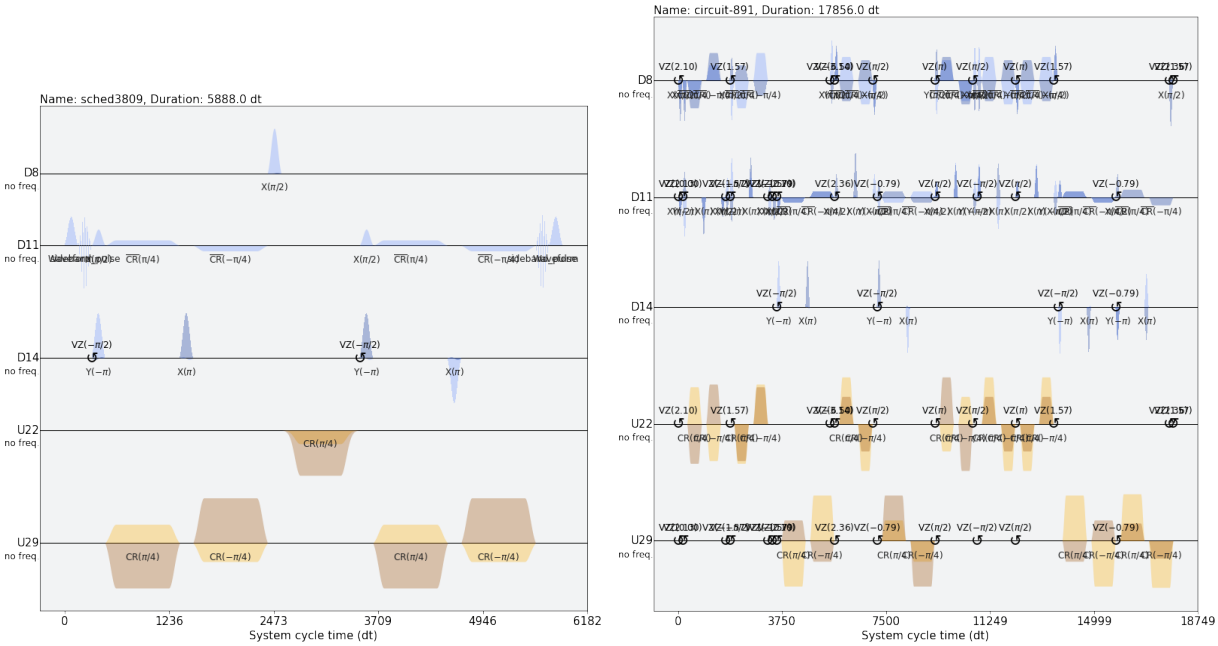



図 4.15　量子トリットを用いた新しい C2 トフォリゲートと量子ビットのみを用いた従来のトフォリゲートのパルスシークエンス。従来のトフォリゲートは Optimization level3 の Qiskit の transpiler を用いてコンパイルしたが、より多くの量子ビット結合を要求するため SWAP も生じている。

では 0.930 となった。なおここで重要であるのは初期状態が $|000\rangle$ であると、図 4.1 中の C2 トフォリにおいて量子トリットとして扱っている真ん中の量子ビットが $X_{12}$ ゲートによって $|2\rangle$ へと励起される。その後自作した direct $C_1X_{01}$ が作用するが、図 4.16 から direct $C_1X_{01}$ の標的ビットは x 軸周りにほぼ回転していないことが確認できる。よって direct $C_1X_{01}$ が、制御ビットが $|2\rangle$ の時に標的ビットにおける Rabi 振動を起こさない CNOT としての役割を果たしていることが分かる。また初期状態が $\frac{1}{2}(|000\rangle + |010\rangle + |100\rangle + |110\rangle$ の場合の古典忠実度は、量子トリットを用いた C2 トフォリでは 0.977、IBM の従来のトフォリでは 0.971 となった。制御ビットの状態が $|11\rangle$ の時にのみ標的ビットの反転を起

こしており、目的通りの挙動を示していることがわかる。

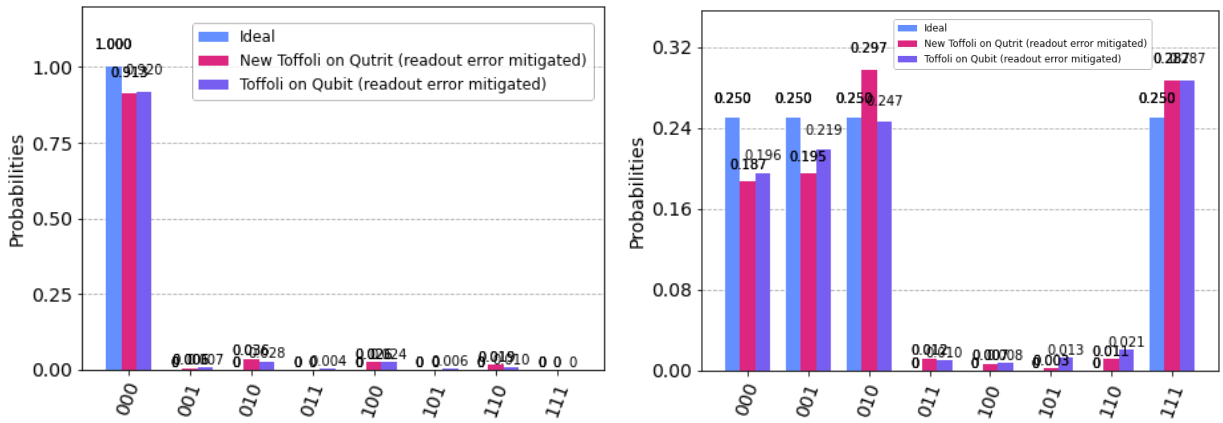



図 4.16　初期状態 $|000\rangle$ と $\frac{1}{2}(|000\rangle + |010\rangle + |100\rangle + |110\rangle)$ に対して、相対位相を無視した C2 トフォリと従来のトフォリをそれぞれ作用させた結果を Z 基底で測定した時の古典忠実度。

最後にそれぞれのトフォリを 1,5,...96 個並べた時の古典忠実度の指数減衰の様子を確認した。なお 3 量子ビットの場合は脱分極エラーモデルにより完全に量子状態が混合した場合の古典忠実度が 0.7071 となるので、この値が最低ライン基準となる。図 4.17 のうち左図を見ると従来のトフォリでは 9 個のトフォリが作用した時点で最小値となっており量子状態が完全に混合してしまっていると予想できる。それに対して C2 トフォリでは 36 個のトフォリを作用させるまで古典忠実度の面では優位性を保っていることがわかる。しかし C2 トフォリは 36 個以降 0.7071 をさらに下回っている。図 4.17 のうち、右図では横軸をトフォリ数ではなく総ゲート時間をした。この図を見ると 20 μs、つまり約 14 個の C2 トフォリが作用するまでは従来のトフォリとほぼ同じ減衰をしていることから脱分極モデルで説明できると予測できるが、20 μs 以降は従来のトフォリと異なる減衰が起きている。ただし、この図から C2 トフォリは CNOT 数減少によるゲート時間減少が従来のトフォリに対する優位性の最も大きな要因の一つであることを示唆している。

この挙動の原因を調べるため、C2 トフォリを初期状態 $\frac{1}{2}(|000\rangle + |010\rangle + |100\rangle + |110\rangle$ に対して 99 個作用させた後、測定を行った。図 4.18 のうち、Qiskit の measurement level 1 による左図では量子トリットとして扱っている真ん中の量子ビットが、ほぼ $|1\rangle$ の状態であると判定されている。そこで Qiskit の measurement level 0 により真ん中の量子ビットの IQ プロットを確認したところ、$|1\rangle$ のみではなく $|2\rangle$ への漏れも多く含まれていたことが判明した。考えられる原因は各ゲートにおける漏れやキャリブレーションエラー、電荷分散による振動数ズレなどが考えられる。また量子トリットでは脱励起 $T_1$ や位相緩和 $T_2$ が量子ビットに比べて早いのでその影響の可能性もある。これらについては今後さらなる研究が必要である。

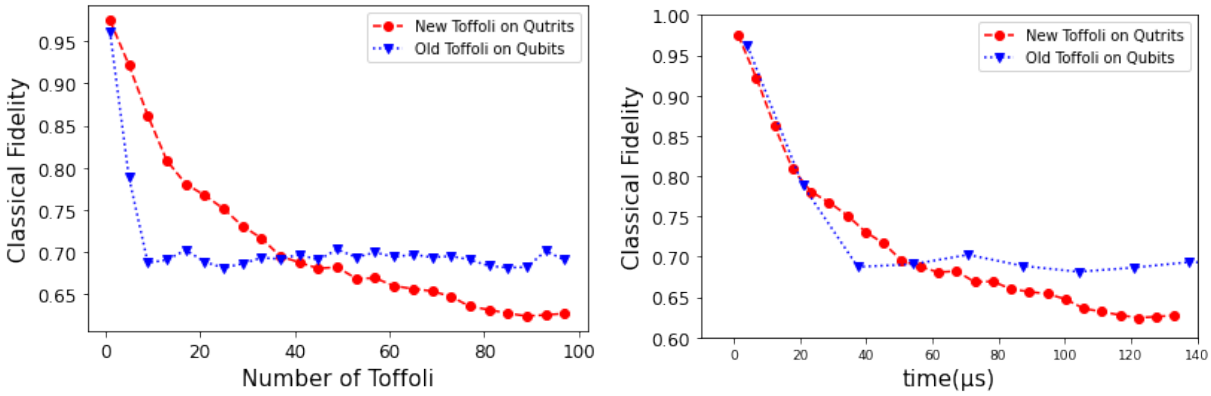



図 4.17　C2 トフォリと従来のトフォリを 1,5,...96 個並べた時の古典忠実度減衰の様子。従来のトフォリでは 9 個のトフォリが作用した時点で量子状態が完全に混合してしまっていると予想できる。それに対して C2 トフォリでは 36 個のトフォリを作用させるまで古典忠実度の面では優位性を保っている。

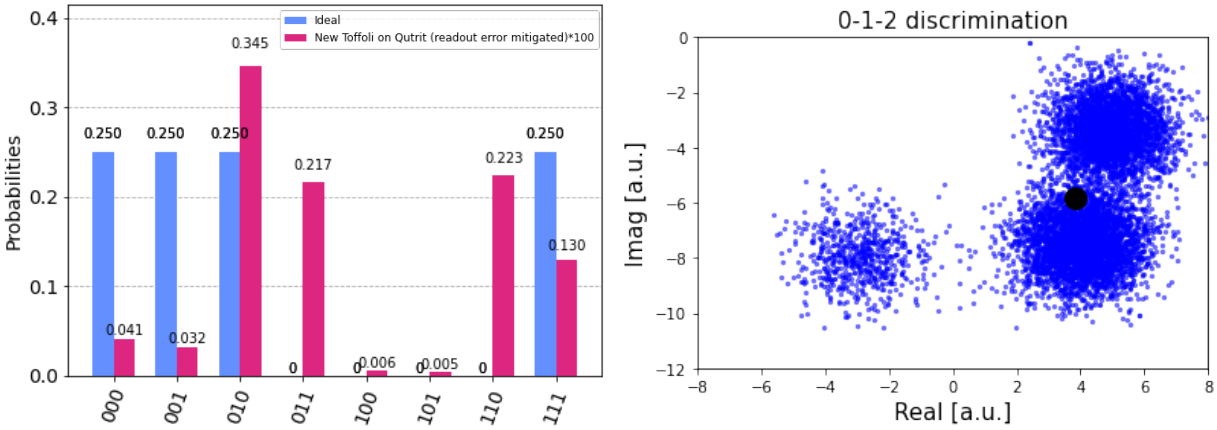



図 4.18　C2 トフォリを初期状態 $\frac{1}{2}(|000\rangle + |010\rangle + |100\rangle + |110\rangle$ に対して 99 個作用させた後、測定した結果。左図が Qiskit の measurement level 1 によって自動で $|0\rangle$ か $|1\rangle$ かを判別させたものであり、右図が Qiskit の measurement level 0 により量子トリットとして扱っている制御ビットの IQ プロットを確認したものである。

# 第 5 章

# 結論と展望

本研究では超伝導量子コンピュータにおいて最も大きいエラー源である CNOT を多く消費する多量子ビットゲートを、より少ない CNOT 数で実装するために、初期状態に依存した量子回路最適化手法の開発と量子トリットを用いたパルスレベルでの効率的なトフォリゲート実装を試みた。量子回路最適化では、回路の前半で一部の計算基底のみが用いられるような量子アルゴリズムにおいて先行研究よりも多くのビット制御の削除を可能とし、任意の量子アルゴリズムに対してビット列の測定過程における閾値の設定によって近似回路生成が可能な最適化プロトコルの開発に成功した。量子トリットの研究においては、二量子トリットゲートのパルスシークエンスの考案と簡単な実証実験によって、将来的な量子トリットの実用化の道を開いた。以上の成果より、NISQ 時代における多量子ビットゲートの効率的な実装を加速させた。また量子トリットを用いた量子誤り訂正の実用化に向けた足がかりとなった。

## 5.1 量子回路最適化

本研究では量子回路最適化の中でも初期状態に依存した不要な制御操作削除に注目し、任意のビット制御数を持つ 1 つの制御ゲートのみに注目した時の理論的な条件式を導いた。そしてこの条件式から先行研究を拡張した十分条件を取り出し、エンタングルしている制御ビットからもビット制御を削除できる新たな最適化手法を開発した。そのために必要な制御ビットにおけるビット列の情報を量子計算機を用いて繰り返し測定することで、多項式計算量で得ることに成功した。またノイズ由来のビット列を切り捨てるための閾値を導入した。任意のビット制御数をもつ制御ゲートもトフォリ以下に分解することで、多項式計算量でエンタングルしている制御ビットから一部の不要な制御操作を削除することができることを導いた。同時に制御ゲートを分解することで削除できなくなる不要なビット制御が生じる可能性があることも導いた。AQCEL を実際に *ibmq_sydney* を用いてパートンシャワー量子アルゴリズムに適用し、AQCEL を適用しない場合は CNOT 数が 532 で $F_{\mathrm{meas}}$ が 0.398 だったのに対して、実機で測定し適切な閾値を設定した AQCEL (QC,low$\leq$0.2) の時に CNOT 数が 153 で $F_{\mathrm{meas}}$ が 0.765 と、有効であることを実証した。この結果は古典計算機で最適化した AQCEL (CC) の結果である CNOT 数が 153 で $F_{\mathrm{meas}}$ が 0.791 と非常に近かった。さらに実機で測定し高い閾値を設定した AQCEL (QC,0.2) では量子ノイズ由来でない低振幅のビット列も多く棄却することで、CNOT 数が 100 で $F_{\mathrm{meas}}$ が 0.796 と AQCEL (CC) よりも高い $F_{\mathrm{meas}}$ となっていることから、効率の良い近似回路生成も行うことができると実証した。

AQCEL の改善点としてはより高度な量子エラー緩和の導入と、閾値の計算を脱分極エラーモデルに基づくなどしてより高精度にすることである。AQCEL の発展の方向性として現在の制限を拡張することが挙げられる。つまり今の十分条件を拡張する。ZX-calculus のようなグラフィカルの表現では一部の振幅の情報を多項式計算量で取り出すことができ、標的ビットの情報も含めて新たな最適化が可能となる。複数の制御ゲートを一つのグループとして捉えたり、作業ビットも導入すると新たな最適化が可能となる。

## 5.2　量子トリット

本研究では超伝導量子コンピュータが量子トリットとしても扱えることや量子トリット上で多量子ビットゲートを効率よく分解できることに注目し、量子トリットにおける制御ゲート実装手法についての研究を行った。まず IBM Quantum でも実装できる可能性が高い量子トリットゲートのみを用いた現実的な分解方法を多く提案した。また量子トリットにおいても正しく作用する二量子トリットゲートの実装手法として direct CNOT と echo CNOT の両方のパルスシークエンスを提案した。Direct CNOT は最もゲート時間が短く、echo CNOT は AC-Stark shift 由来の $ZI$ エラーを自動補正できる。最後に $X_{12}$ ゲートと Direct $C_1X_{01}$ を組み合わせて図 4.1 中の C2 トフォリを実装し、量子ビット空間における相対位相を無視した古典忠実度を測定した。その結果 $|2\rangle$ への漏れを無視すれば 36 個のトフォリゲートを並べるまでは従来のトフォリに対して優位性があることが判明した。量子トリットでは相対位相エラーが非常に重要なので、相対位相まで考慮すると優位性が下がる可能性が高いが、これは本研究で考案した echo CNOT や新たな手法開発等によって十分に優位性を見出せるほどまで改善できると期待している。

なお修士論文提出後にも研究が続き、量子回路最適化の成果は [87] で、量子トリットの成果は [58] でも公表されました。

# 謝辞

本研究は多くの方のご協力のもと成り立っています。指導教員である澤田龍准教授には研究に関する指導はもちろんのこと、色々な場面で迷った時には適切な方向性を示して頂き、充実した修士生活を送ることができました。大変お世話になりました。寺師弘二准教授には研究テーマの相談から始まり、幾度も密接な議論をして頂いたおかげで論文を出すまでに至ることが出来ました。さらに学会発表や共同研究の様々な機会も与えて頂き、大きく成長することが出来ました。本当にありがとうございました。田中純一教授には研究発表へのアドバイスを頂いたり、輪講で多くの質問を頂き理解を深めることができました。飯山悠太郎助教には最適化の定式化など、研究で詰まっていた所ではいつも的確なアドバイスをして頂きました。稲田聡明特任助教には何度も個人的に相談に乗って頂き、パルス制御の研究を形になるまで進めることができました。齊藤真彦特任助教にはコードを一緒に開発したり計算量に関して有益な議論をさせて頂きました。永野廉人特任研究員にはミーティングで何度も質問やアドバイスを頂き参考にさせて頂きました。改めて ICEPP の量子コンピューティンググループの皆さんに感謝申し上げます。

本研究の回路最適化においては Lawrence Berkeley National Laboratory の方々にもご協力頂きました。Christian W. Bauer さんは本研究の計算量を打破するアイデアやエラー軽減について貴重な提案をして頂きました。Benjamin Nachman さんにはコードの相談から論文の作成まで大変ご協力頂きました。量子トリットの研究においては IBM 東京基礎研究所の金澤直輝研究員に有意義な情報を多く共有して頂き、様々なアイデアの考案のきっかけとなりました。増渕達也助教には物理解析の基礎の指導から研究テーマの相談まで乗って頂きました。Sanmay Ganguly 特任助教にも物理解析における機械学習に関して情報共有をして頂きました。

一緒に量子計算の研究を頑張った同期の大久保さんにはコードを参考にさせてもらったり、何度も鋭い質問や議論、コメントをしてくれて大変勉強になりました。水原さんはパルスに関して色々議論できてとても楽しかったです。並木さんと Lu さんとは研究室に行く度に雑談をしてとても楽しい時間を過ごすことができました。同期の皆さんありがとうございました。最後に、素晴らしい研究環境を提供して頂いた浅井祥仁教授、研究生活を支えて頂いた ICEPP 事務室の皆様、家族に感謝申し上げます。

# 引用文献


[1] The ATLAS Experiment at the CERN Large Hadron Collider. *JINST*, 3:S08003. 437 p, 2008. doi: 10.1088/1748-0221/3/08/S08003. URL https://cds.cern.ch/record/1129811. Also published by CERN Geneva in 2010.

[2] Apollinari G., Béjar Alonso I., Brüning O., Fessia P., Lamont M., Rossi L., and Tavian L. *High-Luminosity Large Hadron Collider (HL-LHC): Technical Design Report V. 0.1*. CERN Yellow Reports: Monographs. CERN, Geneva, 2017. doi: 10.23731/CYRM-2017-004. URL https://cds.cern.ch/record/2284929.

[3] P Calafiura, J Catmore, D Costanzo, and A Di Girolamo. ATLAS HL-LHC Computing Conceptual Design Report. Technical report, CERN, Geneva, Sep 2020. URL https://cds.cern.ch/record/2729668.

[4] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, USA, 10th edition, 2011. ISBN 1107002176.

[5] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.*, 41(2):303–332, 1999.

[6] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 212–219, New York, NY, USA, 1996. Association for Computing Machinery. ISBN 0897917855. doi: 10.1145/237814.237866. URL https://doi.org/10.1145/237814.237866.

[7] Tüysüz, Cenk, Carminati, Federico, Demirköz, Bilge, Dobos, Daniel, Fracas, Fabio, Novotny, Kristiane, Potamianos, Karolos, Vallecorsa, Sofia, and Vlimant, Jean-Roch. Particle track reconstruction with quantum algorithms. *EPJ Web Conf.*, 245:09013, 2020. doi: 10.1051/epjconf/202024509013. URL https://doi.org/10.1051/epjconf/202024509013.

[8] Cenk Tüysüz, Carla Rieger, Kristiane Novotny, Bilge Demirkoz, Daniel Dobos, Karolos Potamianos, Sofia Vallecorsa, Jean-Roch Vlimant, and Richard Forster. Hybrid quantum classical graph neural networks for particle track reconstruction. *Quantum Machine Intelligence*, 3, 12 2021. doi: 10.1007/s42484-021-00055-9.

[9] Benjamin Nachman, Davide Provasoli, Wibe A. de Jong, and Christian W. Bauer. Quantum algorithm for high energy physics simulations. *Physical Review Letters*, 126(6), Feb 2021. doi: 10.1103/physrevlett.126.062001.

[10] Davide Provasoli, Benjamin Nachman, Christian Bauer, and Wibe A de Jong. A quantum algorithm to efficiently sample from interfering binary trees. *Quantum Sci. Technol.*, 5:035004, 2020. doi: 10.1088/2058-9565/ab8359.

[11] Koji Terashi, Michiru Kaneda, Tomoe Kishimoto, Masahiko Saito, Ryu Sawada, and Junichi Tanaka. Event Classification with Quantum Machine Learning in High-Energy Physics. *Comput. Softw. Big Sci.*, 5:2, 2021. doi: 10.1007/s41781-020-00047-7.

[12] Belis, Vasilis, González-Castillo, Samuel, Reissel, Christina, Vallecorsa, Sofia, Combarro, Elías F., Dissertori, Günther, and Reiter, Florentin. Higgs analysis with quantum classifiers. *EPJ Web Conf.*, 251:03070, 2021. doi: 10.1051/epjconf/202125103070. URL https://doi.org/10.1051/epjconf/202125103070.

[13] Sau Lan Wu, Jay Chan, Wen Guan, Shaojun Sun, Alex Wang, Chen Zhou, Miron Livny, Federico Carminati, Alberto Di Meglio, Andy C. Y. Li, Joseph D Lykken, Panagiotis Spentzouris, Samuel Yen-Chi Chen, Shinjae Yoo, and Tzu-Chieh Wei. Application of quantum machine learning using the quantum variational classifier method to high energy physics analysis at the lhc on ibm quantum computer simulator and hardware with 10 qubits. *Journal of Physics G: Nuclear and Particle Physics*, 2021. URL http://iopscience.iop.org/article/10.1088/1361-6471/ac1391.

[14] Wen Guan, Gabriel Perdue, Arthur Pesah, Maria Schuld, Koji Terashi, Sofia Vallecorsa, and Jean-Roch Vlimant. Quantum machine learning in high energy physics. *Mach. Learn.: Sci. Technol.*, 2: 011003, 2021. doi: 10.1088/2632-2153/abc17d.

[15] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver. A quantum engineer's guide to superconducting qubits. *Applied Physics Reviews*, 6(2):021318, 2019. doi: 10.1063/1.5089550. URL https://doi.org/10.1063/1.5089550.

[16] Sergei Slussarenko and Geoff J. Pryde. Photonic quantum information processing: A concise review. *Applied Physics Reviews*, 6(4):041303, 2019. doi: 10.1063/1.5115814. URL https://doi.org/10.1063/1.5115814.

[17] Colin D. Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M. Sage. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews*, 6(2):021314, 2019. doi: 10.1063/1.5088164. URL https://doi.org/10.1063/1.5088164.

[18] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018. doi: 10.22331/q-2018-08-06-79.

[19] D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, page 176–188, New York, NY, USA, 1997. Association for Computing Machinery. ISBN 0897918886. doi: 10.1145/258533.258579. URL https://doi.org/10.1145/258533.258579.

[20] Kosuke Fukui, Akihisa Tomita, Atsushi Okamoto, and Keisuke Fujii. High-threshold fault-tolerant quantum computation with analog quantum error correction. *Phys. Rev. X*, 8:021054, May 2018. doi: 10.1103/PhysRevX.8.021054. URL https://link.aps.org/doi/10.1103/PhysRevX.8.021054.

[21] M. D. Reed, L. DiCarlo, S. E. Nigg, L. Sun, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf. Realization of three-qubit quantum error correction with superconducting circuits. *Nature*, 482 (7385):382–385, Feb 2012. ISSN 1476-4687. doi: 10.1038/nature10786. URL http://dx.doi.org/10.1038/nature10786.

[22] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, and et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, Oct 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-1666-5. URL http://dx.doi.org/10.1038/s41586-019-1666-5.

[23]

[24] Andre He, Benjamin Nachman, Wibe A. de Jong, and Christian W. Bauer. Zero-noise extrapolation for quantum-gate error mitigation with identity insertions. *Phys. Rev. A*, 102:012426, Jul 2020. doi: 10.1103/PhysRevA.102.012426. URL https://link.aps.org/doi/10.1103/PhysRevA.102.012426.

[25] MD SAJID ANIS et al. Qiskit: An open-source framework for quantum computing, 2021.

[26] Cirq Developers. Cirq, August 2021. URL https://doi.org/10.5281/zenodo.5182845. See full list of authors on Github: https://github .com/quantumlib/Cirq/graphs/contributors.

[27] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. t|ket⟩: a retargetable compiler for nisq devices. *Quantum Science and Technology*, 6(1): 014003, Nov 2020. ISSN 2058-9565. doi: 10.1088/2058-9565/ab8e92. URL http://dx.doi.org/10.1088/2058-9565/ab8e92.

[28] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. Quantum circuit learning. *Physical Review A*, 98(3), Sep 2018. ISSN 2469-9934. doi: 10.1103/physreva.98.032309. URL http://dx.doi.org/10.1103/PhysRevA.98.032309.

[29] Marc Grau Davis, Ethan Smith, Ana Tudor, Koushik Sen, Irfan Siddiqi, and Costin Iancu. Heuristics for quantum compiling with a continuous gate set, 2019.

[30] Matthew Otten, Cristian L. Cortes, and Stephen K. Gray. Noise-resilient quantum dynamics using symmetry-preserving ansatzes, 2019.

[31] Sumeet Khatri, Ryan LaRose, Alexander Poremba, Lukasz Cincio, Andrew T. Sornborger, and Patrick J. Coles. Quantum-assisted quantum compiling. *Quantum*, 3:140, May 2019. ISSN 2521-327X. doi: 10.22331/q-2019-05-13-140. URL http://dx.doi.org/10.22331/q-2019-05-13-140.

[32] Thomas Alexander, Naoki Kanazawa, Daniel J Egger, Lauren Capelluto, Christopher J Wood, Ali Javadi-Abhari, and David C McKay. Qiskit pulse: programming quantum computers through the cloud with pulses. *Quantum Science and Technology*, 5(4):044006, Aug 2020. ISSN 2058-9565. doi: 10.1088/2058-9565/aba404. URL http://dx.doi.org/10.1088/2058-9565/aba404.

[33] Pranav Gokhale, Ali Javadi-Abhari, Nathan Earnest, Yunong Shi, and Frederic T. Chong. Optimized quantum compilation for near-term algorithms with openpulse. *2020 53rd Annual IEEE/ACM*

*International Symposium on Microarchitecture (MICRO)*, pages 186–200, 2020.

[34] Shelly Garion, Naoki Kanazawa, Haggai Landa, David C. McKay, Sarah Sheldon, Andrew W. Cross, and Christopher J. Wood. Experimental implementation of non-clifford interleaved randomized benchmarking with a controlled-s gate. *Physical Review Research*, 3(1), Mar 2021. ISSN 2643-1564. doi: 10.1103/physrevresearch.3.013204. URL http://dx.doi.org/10.1103/PhysRevResearch.3.013204.

[35] Nathan Earnest, Caroline Tornow, and Daniel J. Egger. Pulse-efficient circuit transpilation for quantum applications on cross-resonance-based hardware. *Phys. Rev. Research*, 3:043088, Oct 2021. doi: 10.1103/PhysRevResearch.3.043088. URL https://link.aps.org/doi/10.1103/PhysRevResearch.3.043088.

[36] Ibm quantum, 2021. URL https://quantum-computing.ibm.com.

[37] Ross Duncan, Aleks Kissinger, Simon Perdrix, and John van de Wetering. Graph-theoretic simplification of quantum circuits with the zx-calculus. *Quantum*, 4:279, Jun 2020. ISSN 2521-327X. doi: 10.22331/q-2020-06-04-279. URL http://dx.doi.org/10.22331/q-2020-06-04-279.

[38] John van de Wetering. Zx-calculus for the working quantum computer scientist, 2020.

[39] Ji Liu, Luciano Bello, and Huiyang Zhou. Relaxed peephole optimization: A novel compiler optimization for quantum circuits. In *Proceedings of the 2021 IEEE/ACM International Symposium on Code Generation and Optimization*, CGO '21, page 301–314. IEEE Press, 2021. ISBN 9781728186139. doi: 10.1109/CGO51591.2021.9370310. URL https://doi.org/10.1109/CGO51591.2021.9370310.

[40] Pranav Gokhale, Jonathan M. Baker, Casey Duckering, Natalie C. Brown, Kenneth R. Brown, and Frederic T. Chong. Asymptotic improvements to quantum circuits via qutrits. *Proceedings of the 46th International Symposium on Computer Architecture*, Jun 2019. doi: 10.1145/3307650.3322253. URL http://dx.doi.org/10.1145/3307650.3322253.

[41] Yushi Wang and Marek Perkowski. Improved complexity of quantum oracles for ternary grover algorithm for graph coloring. In *2011 41st IEEE International Symposium on Multiple-Valued Logic*, pages 294–301, 2011. doi: 10.1109/ISMVL.2011.42.

[42] Alexey Galda, Michael Cubeddu, Naoki Kanazawa, Prineha Narang, and Nathan Earnest-Noble. Implementing a ternary decomposition of the toffoli gate on fixed-frequencytransmon qutrits, 2021.

[43] Toshiaki Inada, Wonho Jang, Yutaro Iiyama, Koji Terashi, Ryu Sawada, Junichi Tanaka, and Shoji Asai. Measurement-free ultrafast quantum error correction by using multi-controlled gates in higher-dimensional state space, 2021.

[44] Yuchen Wang, Zixuan Hu, Barry C. Sanders, and Sabre Kais. Qudits and high-dimensional quantum computing. *Frontiers in Physics*, 8, Nov 2020. ISSN 2296-424X. doi: 10.3389/fphy.2020.589504. URL http://dx.doi.org/10.3389/fphy.2020.589504.

[45] T. C. Ralph, K. J. Resch, and A. Gilchrist. Efficient toffoli gates using qudits. *Physical Review A*, 75(2), Feb 2007. ISSN 1094-1622. doi: 10.1103/physreva.75.022313. URL http://dx.doi.org/10.1103/PhysRevA.75.022313.

[46] E. O. Kiktenko, A. S. Nikolaeva, Peng Xu, G. V. Shlyapnikov, and A. K. Fedorov. Scalable quantum computing with qudits on a graph. *Physical Review A*, 101(2), Feb 2020. ISSN 2469-9934. doi: 10.1103/physreva.101.022304. URL http://dx.doi.org/10.1103/PhysRevA.101.022304.

[47] Amit Saha, Debasri Saha, and Amlan Chakrabarti. Moving quantum states without swap via intermediate higher-dimensional qudits. *Phys. Rev. A*, 106:012429, Jul 2022. doi: 10.1103/PhysRevA.106.012429. URL https://link.aps.org/doi/10.1103/PhysRevA.106.012429.

[48] Mehdi Saeedi and Igor L. Markov. Synthesis and optimization of reversible circuits—a survey. *ACM Comput. Surv.*, 45(2), mar 2013. ISSN 0360-0300. doi: 10.1145/2431211.2431220. URL https://doi.org/10.1145/2431211.2431220.

[49] Jing Zhong. A new fault model and its application in synthesizing toffoli networks. 2008.

[50] Jing Zhong and Jon C. Muzio. Using crosspoint faults in simplifying toffoli networks. *2006 IEEE North-East Workshop on Circuits and Systems*, pages 129–132, 2006.

[51] D. Michael Miller, Robert Wille, and Rolf Drechsler. Reducing reversible circuit cost by adding lines. In *2010 40th IEEE International Symposium on Multiple-Valued Logic*, pages 217–222, 2010. doi: 10.1109/ISMVL.2010.48.

[52] David C. McKay, Thomas Alexander, Luciano Bello, Michael J. Biercuk, Lev Bishop, Jiayin Chen, Jerry M. Chow, Antonio D. Córcoles, Daniel Egger, Stefan Filipp, Juan Gomez, Michael Hush, Ali Javadi-Abhari, Diego Moreda, Paul Nation, Brent Paulovicks, Erick Winston, Christopher J. Wood, James Wootton, and Jay M. Gambetta. Qiskit backend specifications for openqasm and openpulse experiments, 2018.

[53] Alba Cervera-Lierta, Mario Krenn, Alán Aspuru-Guzik, and Alexey Galda. Experimental high-dimensional greenberger-horne-zeilinger entanglement with superconducting transmon qutrits. *Phys. Rev. Applied*, 17:024062, Feb 2022. doi: 10.1103/PhysRevApplied.17.024062. URL https://link.aps.org/doi/10.1103/PhysRevApplied.17.024062.

[54] M. S. Blok, V. V. Ramasesh, T. Schuster, K. O'Brien, J. M. Kreikebaum, D. Dahlen, A. Morvan, B. Yoshida, N. Y. Yao, and I. Siddiqi. Quantum information scrambling on a superconducting qutrit processor. *Phys. Rev. X*, 11:021010, Apr 2021. doi: 10.1103/PhysRevX.11.021010. URL https://link.aps.org/doi/10.1103/PhysRevX.11.021010.

[55] Jens Koch, Terri M. Yu, Jay Gambetta, A. A. Houck, D. I. Schuster, J. Majer, Alexandre Blais, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf. Charge-insensitive qubit design derived from the cooper pair box. *Physical Review A*, 76(4), Oct 2007. ISSN 1094-1622. doi: 10.1103/physreva.76.042319. URL http://dx.doi.org/10.1103/PhysRevA.76.042319.

[56] 野口 篤史 長田 有登, 山崎 歴舟. 量子技術序論, March 2021. URL https://www.sqei.c.u-tokyo.ac.jp/qed/QEd_textbook.pdf.

[57] Wonho Jang, Koji Terashi, Masahiko Saito, Christian W. Bauer, Benjamin Nachman, Yutaro Iiyama, Tomoe Kishimoto, Ryunosuke Okubo, Ryu Sawada, and Junichi Tanaka. Quantum gate pattern recognition and circuit optimization for scientific applications. *EPJ Web of Conferences*, 251:03023, 2021. ISSN 2100-014X. doi: 10.1051/epjconf/202125103023. URL http://dx.doi.org/10.

1051/epjconf/202125103023.

[58] Wonho Jang, Yutaro Iiyama, Naoki Kanazawa, Toshiaki Inada, Ryu Sawada, Tamiya Onodera, and Koji Terahsi. Stable toffoli gate on fixed-frequency superconducting qutrits. IEEE Quantum Week 2022.

[59] Andrei B Klimov, Luis L Sánchez-Soto, Hubert de Guise, and Gunnar Björk. Quantum phases of a qutrit. *Journal of Physics A: Mathematical and General*, 37(13):4097–4106, Mar 2004. ISSN 1361-6447. doi: 10.1088/0305-4470/37/13/012. URL http://dx.doi.org/10.1088/0305-4470/37/13/012.

[60] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52(5):3457–3467, Nov 1995. ISSN 1094-1622. doi: 10.1103/physreva.52.3457. URL http://dx.doi.org/10.1103/PhysRevA.52.3457.

[61] Yang Liu, Gui Lu Long, and Yang Sun. Analytic constructions of general n-qubit controlled gates, 2007.

[62] Dmitri Maslov. Advantages of using relative-phase toffoli gates with an application to multiple control toffoli optimization. *Physical Review A*, 93(2), Feb 2016. ISSN 2469-9934. doi: 10.1103/physreva.93.022311. URL http://dx.doi.org/10.1103/PhysRevA.93.022311.

[63] A. Morvan, V. V. Ramasesh, M. S. Blok, J. M. Kreikebaum, K. O'Brien, L. Chen, B. K. Mitchell, R. K. Naik, D. I. Santiago, and I. Siddiqi. Qutrit randomized benchmarking. *Physical Review Letters*, 126(21), May 2021. ISSN 1079-7114. doi: 10.1103/physrevlett.126.210504. URL http://dx.doi.org/10.1103/PhysRevLett.126.210504.

[64] Yao-Min Di and Hai-Rui Wei. Elementary gates for ternary quantum logic circuit, 2012.

[65] M. A. Yurtalan, J. Shi, M. Kononenko, A. Lupascu, and S. Ashhab. Implementation of a walsh-hadamard gate in a superconducting qutrit. *Physical Review Letters*, 125(18), Oct 2020. ISSN 1079-7114. doi: 10.1103/physrevlett.125.180504. URL http://dx.doi.org/10.1103/PhysRevLett.125.180504.

[66] David C. McKay, Christopher J. Wood, Sarah Sheldon, Jerry M. Chow, and Jay M. Gambetta. Efficient z gates for quantum computing. *Physical Review A*, 96(2), Aug 2017. ISSN 2469-9934. doi: 10.1103/physreva.96.022330. URL http://dx.doi.org/10.1103/PhysRevA.96.022330.

[67] Jerry M. Chow, A. D. Córcoles, Jay M. Gambetta, Chad Rigetti, B. R. Johnson, John A. Smolin, J. R. Rozen, George A. Keefe, Mary B. Rothwell, Mark B. Ketchen, and et al. Simple all-microwave entangling gate for fixed-frequency superconducting qubits. *Physical Review Letters*, 107(8), Aug 2011. ISSN 1079-7114. doi: 10.1103/physrevlett.107.080502. URL http://dx.doi.org/10.1103/PhysRevLett.107.080502.

[68] Sarah Sheldon, Easwar Magesan, Jerry M. Chow, and Jay M. Gambetta. Procedure for systematically tuning up cross-talk in the cross-resonance gate. *Physical Review A*, 93(6), Jun 2016. ISSN 2469-9934. doi: 10.1103/physreva.93.060302. URL http://dx.doi.org/10.1103/PhysRevA.93.060302.

[69] Petar Jurcevic, Ali Javadi-Abhari, Lev Bishop, I. Lauer, Daniela F. Bogorin, Markus Brink, Lauren Capelluto, Oktay Günlük, Toshinari Itoko, Naoki Kanazawa, Abhinav Kandala, George A Keefe, Kevin Krsulich, William Landers, Eric P Lewandowski, Douglas T. McClure, Giacomo Nannicini, Adinath Narasgond, Hasan Nayfeh, Emily J. Pritchett, Mary Beth Rothwell, Srikanth Srinivasan, Neereja M. Sundaresan, Cindy Wang, Ken Xuan Wei, Christopher J. Wood, Jeng-Bang Yau, Eric J Zhang, Oliver E Dial, Jerry M. Chow, and Jay M. Gambetta. Demonstration of quantum volume 64 on a superconducting quantum computing system. *Quantum Science & Technology*, 6, 2020.

[70] Easwar Magesan and Jay M. Gambetta. Effective hamiltonian models of the cross-resonance gate. *Physical Review A*, 101(5), May 2020. ISSN 2469-9934. doi: 10.1103/physreva.101.052308. URL http://dx.doi.org/10.1103/PhysRevA.101.052308.

[71] D. I. Schuster, A. Wallraff, A. Blais, L. Frunzio, R.-S. Huang, J. Majer, S. M. Girvin, and R. J. Schoelkopf. ac stark shift and dephasing of a superconducting qubit strongly coupled to a cavity field. *Physical Review Letters*, 94(12), Mar 2005. ISSN 1079-7114. doi: 10.1103/physrevlett.94.123602. URL http://dx.doi.org/10.1103/PhysRevLett.94.123602.

[72] Neereja Sundaresan, Isaac Lauer, Emily Pritchett, Easwar Magesan, Petar Jurcevic, and Jay M. Gambetta. Reducing unitary and spectator errors in cross resonance with optimized rotary echoes. *PRX Quantum*, 1(2), Dec 2020. ISSN 2691-3399. doi: 10.1103/prxquantum.1.020318. URL http://dx.doi.org/10.1103/PRXQuantum.1.020318.

[73] A. Kandala, K. X. Wei, S. Srinivasan, E. Magesan, S. Carnevale, G. A. Keefe, D. Klaus, O. Dial, and D. C. McKay. Demonstration of a high-fidelity cnot gate for fixed-frequency transmons with engineered zz suppression. *Physical Review Letters*, 127(13), Sep 2021. ISSN 1079-7114. doi: 10.1103/physrevlett.127.130501. URL http://dx.doi.org/10.1103/PhysRevLett.127.130501.

[74] K. X. Wei, E. Magesan, I. Lauer, S. Srinivasan, D. F. Bogorin, S. Carnevale, G. A. Keefe, Y. Kim, D. Klaus, W. Landers, N. Sundaresan, C. Wang, E. J. Zhang, M. Steffen, O. E. Dial, D. C. McKay, and A. Kandala. Quantum crosstalk cancellation for fast entangling gates and improved multi-qubit performance, 2021.

[75] Easwar Magesan, J. M. Gambetta, and Joseph Emerson. Scalable and robust randomized benchmarking of quantum processes. *Physical Review Letters*, 106(18), May 2011. ISSN 1079-7114. doi: 10.1103/physrevlett.106.180504. URL http://dx.doi.org/10.1103/PhysRevLett.106.180504.

[76] Isaac L. Chuang and M. A. Nielsen. Prescription for experimental determination of the dynamics of a quantum black box. *Journal of Modern Optics*, 44(11-12):2455–2467, Nov 1997. ISSN 1362-3044. doi: 10.1080/09500349708231894. URL http://dx.doi.org/10.1080/09500349708231894.

[77] Michael A Nielsen. A simple formula for the average gate fidelity of a quantum dynamical operation. *Physics Letters A*, 303(4):249–252, Oct 2002. ISSN 0375-9601. doi: 10.1016/s0375-9601(02)01272-0. URL http://dx.doi.org/10.1016/S0375-9601(02)01272-0.

[78] A. Yu. Kitaev. Quantum measurements and the abelian stabilizer problem, 1995.

[79] Andy Buckley, Jonathan Butterworth, Stefan Gieseke, David Grellscheid, Stefan Höche, Hendrik Hoeth, Frank Krauss, Leif Lönnblad, Emily Nurse, Peter Richardson, and et al. General-purpose event generators for lhc physics. *Physics Reports*, 504(5):145–233, Jul 2011. ISSN 0370-1573. doi: 10.1016/j.physrep.2011.03.005. URL http://dx.doi.org/10.1016/j.physrep.2011.03.005.

[80] Gadi Aleksandrowicz et al. Qiskit: An Open-source Framework for Quantum Computing, 2019. URL https://doi.org/10.5281/zenodo.2562111.

[81] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[82] Qiskit Textbook. Calibrating qubits with qiskit pulse, October 2021. URL https://qiskit.org/textbook/ch-quantum-hardware/calibrating-qubits-pulse.html.

[83] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011. URL http://jmlr.org/papers/v12/pedregosa11a.html.

[84] F. Motzoi, J. M. Gambetta, P. Rebentrost, and F. K. Wilhelm. Simple pulses for elimination of 漏れ in weakly nonlinear qubits. *Physical Review Letters*, 103(11), Sep 2009. ISSN 1079-7114. doi: 10.1103/physrevlett.103.110501. URL http://dx.doi.org/10.1103/PhysRevLett.103.110501.

[85] J. M. Gambetta, F. Motzoi, S. T. Merkel, and F. K. Wilhelm. Analytic control methods for high-fidelity unitary operations in a weakly nonlinear oscillator. *Physical Review A*, 83(1), Jan 2011. ISSN 1094-1622. doi: 10.1103/physreva.83.012308. URL http://dx.doi.org/10.1103/PhysRevA.83.012308.

[86] Qiskit Textbook. Accessing higher energy states, October 2020. URL https://qiskit.org/textbook/ch-quantum-hardware/accessing_higher_energy_states.html.

[87] Wonho Jang, Koji Terashi, Masahiko Saito, Christian W. Bauer, Benjamin Nachman, Yutaro Iiyama, Ryunosuke Okubo, and Ryu Sawada. Initial-State Dependent Optimization of Controlled Gate Operations with Quantum Computer. *Quantum*, 6:798, September 2022. ISSN 2521-327X. doi: 10.22331/q-2022-09-08-798. URL https://doi.org/10.22331/q-2022-09-08-798.

[88] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In *Proceedings of the 7th Python in Science Conference*, 2008. URL https://www.osti.gov/biblio/960616.

[89] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-Lehman Graph Kernels. *J. Mach. Learn. Res.*, 12:2539–2561, 2011. doi: 10.5555/1953048.2078187.

# 付録 A

# 繰り返しゲート群 (RSG) の識別

ここでは AQCEL におけるもう一つの戦略である RSG の識別について議論する。RSG とは量子回路中で多用されるゲート群のことであり、その量子回路の中では最も核となり得る。よって RSG を量子回路最適化、トランスパイル、パルス制御レベルでの最適化における重要なターゲットとして期待することができる。また RSG を効率よく実行することができるトポロジー探索にも応用することが可能であり、特定の量子アルゴリズム専用の実機開発や既存の実機の中で最適な実機の自動選択などにもつながる。

## A.1　DAG での表現

量子回路では量子ビットが量子ゲートが作用し異なる状態へと変化する。また量子回路における左側から右側へと時間発展していくため一方向性である。このような形式は量子ビットをノード、量子ゲートをエッジとして有向非巡回グラフ（directed acyclic graph；DAG) として表現することができる。DAG の性質を用いることで RSG をより簡単に探索することが可能である。

図 A.1 に示されているようにトフォリゲートを DAG で表現した場合は、トフォリゲートを'ccx' というノードで表し、二つの制御ビットのエッジと一つの標的ビットのエッジを合わせた三つのエッジが入力、そして出力されている。

## A.2　RSG の識別

RSG 候補の選び方として、まずスタートなるノードを選択し、そこから後続のノードの組み合わせを考える。このやり方だと単純な全探索であるので探索計算量は $O(N_{\mathrm{nodes}}!)$ となってしまう。[*1]殆どの場合に $N_{\mathrm{nodes}} = n_{\mathrm{gates}} \geq n_{\mathrm{qubits}}$ であるから、$O(N_{\mathrm{nodes}}!) \geq O(2^{n_{\mathrm{qubits}}})$ が成り立ってしまう。そこで本研究ではノード数に RSG 中の一番長い列のゲート数、最小ノード数と最大ノード数に制限を与えている。[*2]最大ノード数を $N_{\mathrm{thr}}$ というように制限がかかることで計算量は $O(N_{\mathrm{nodes}}^{N_{\mathrm{thr}}})$ まで減少する。[*3]なお最小ノード数の制限は非常に小さな RSG を除くためにある。ただしこのアルゴリズムでは図 A.2 の右図のように、

---

[*1] $i$ 番目のノードは最大で $N_{\mathrm{nodes}} - i$ 個の後続ノードを持つ。

[*2] 簡便のため一番長い列のゲート数と最大ノード数は一致させている。

[*3] $N_{\mathrm{nodes}}$ 個のノードの中から $N_{\mathrm{thr}}$ 個の組み合わせ探索であるので計算量は ${}_{N_{\mathrm{nodes}}}C_{N_{\mathrm{thr}}}$ となる。

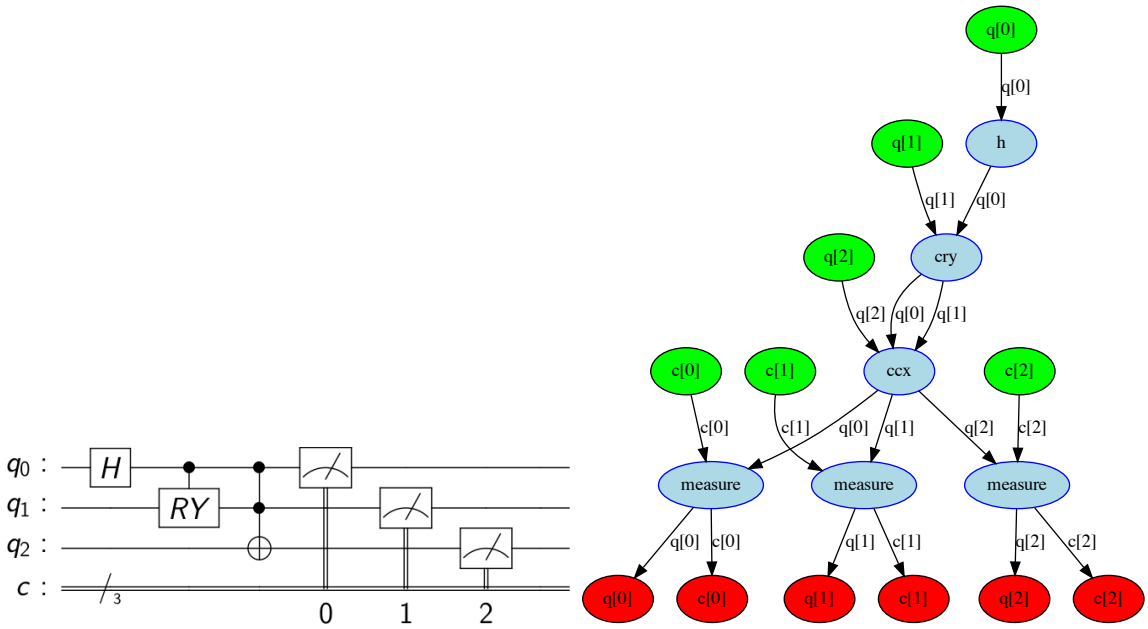



図 A.1　トフォリゲートを含む量子回路の一例 (上図) と DAG による表現 (下図)。

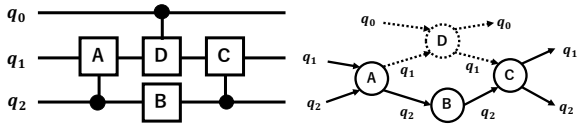



図 A.2　正しい RSG の一例 (左図) とこれの部分グラフ ($G^{'} = \{A, B, C\}$)。$D$ と独立していないので RSG ではない (右図)。

独立したゲート群として見なせない RSG は除去している。これは RSG 中のいずれかのノードにとっての親ノードでもあり子ノードでもある RSG 外部のノードが存在する場合、この RSG を除去することで達成できる ($\exists g_i, g_j \subseteq G^{'}, \{g_k | g_i \to g_k, g_k \to g_j\} \nsubseteq G^{'}$)。

このようにして得られた RSG 候補を全て探索した後、それぞれの繰り返し数を計算する。そのために二つの RSG 候補が同じであるかというグラフ同型問題を解く。NetworkX library [88] に実装されている Weisfeiler Lehman graph hash [89] によってグラフをハッシュ化し同型なグラフかを判断する。 [*4]

量子回路中で RSG は様々な量子ビットへの作用の仕方がある場合があるので、二つの RSG 候補が一致すると判断する条件として次のような 3 つのレベルを設定している。

**Level 1**：　ゲートの種類の並びが一致
**Level 2**：　ゲートの種類の並びとそれぞれのゲートが作用している量子ビットの役割が一致 [*5]
**Level 3**：　ゲートの種類の並びとそれぞれのゲートが作用している量子ビットの役割とインデックスが一致

[*4] ハッシュ化とはグラフをハッシュ関数に入力し、出力としてハッシュ値を得ること。ハッシュ値は固定長の数値であり同じグラフならば同じハッシュ値が得られて、少しでも異なるグラフならば異なるハッシュ値が得られる。
[*5] 制御ゲートの制御ビットと標的ビットのこと。

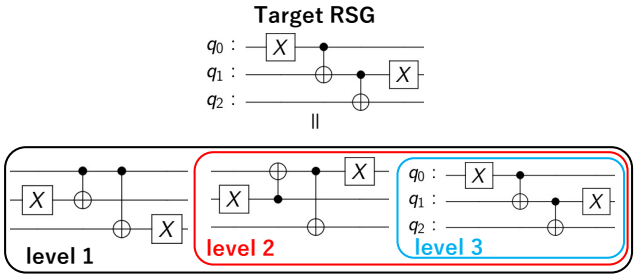



図 A.3　Possible RSG patterns for a given target RSG corresponding to the three levels of matching criteria.

三つのそれぞれの探索レベルが同一だと判断する例を図 A.3 に示している。

このように探索された RSG の様々な候補は RSG 中のゲート数と繰り返し回数の積を点数としてランク付けされる。そして上位 $k$ 個の RSG を取り出す。例えば RSG 中のゲート数をエラー率が最も高い CNOT 数にすれば、エラー率という面で重要な RSG を特定することが可能である。以上より RSG 探索アルゴリズムのスードコードは以下の通りとなる [57]。

---

**Algorithm 2**: Gate set pattern recognition with DAG

---

**for all** quantum gate (node) ($g_i$) in the circuit ($G$) **do**
  **for all** subset ($G^{'}$) beginning with the target node ($g_i$) **do**
    **if** the longest path is longer than the threshold **then**
      continue
    **end if**
    **if** number of elements in subset is out of thresholds **then**
      continue
    **end if**
    **if** $\exists g_i, g_j \subseteq G^{'}, \{g_k | g_i \to g_k, g_k \to g_j\} \nsubseteq G^{'}$ **then**
      continue
    **end if**
    $G^{'}$ is a RSG candidate.
  **end for**
**end for**
Make a set of RSGs ($S(h) = \{G^{'} | \mathrm{hash}(G^{'}) = h\}$)
Select top-$k$ sets of RSGs (S(h)) ordering by the frequency $|S(h)|$ times RSG size ($|G^{'}|$)

---

RSG 探索の応用例として、QPS の 2 ステップシミュレーション回路に対する AQCEL による RSG 探索について議論する。初期状態は $|f_1\rangle$、結合定数は $g_1 = 2$、$g_2 = g_{12} = 1$ とした。よってこの場合は $f \to f'\phi$ だけでなく $\phi \to f\bar{f}$ の反応も起きる汎用的な量子回路となる。

初めに QPS の 2 ステップシミュレーション回路に対する AQCEL による Level 2 の RSG 探索の結果を

図 A.4 に示す。各 RSG のノード数は 5 から 7、量子回路中での繰り返し数は 4 以上という条件をつけている。探索レベルを上げると探索条件が厳しくなるので、基本的にはノード数と繰り返し数が少ない RSG が探索される。

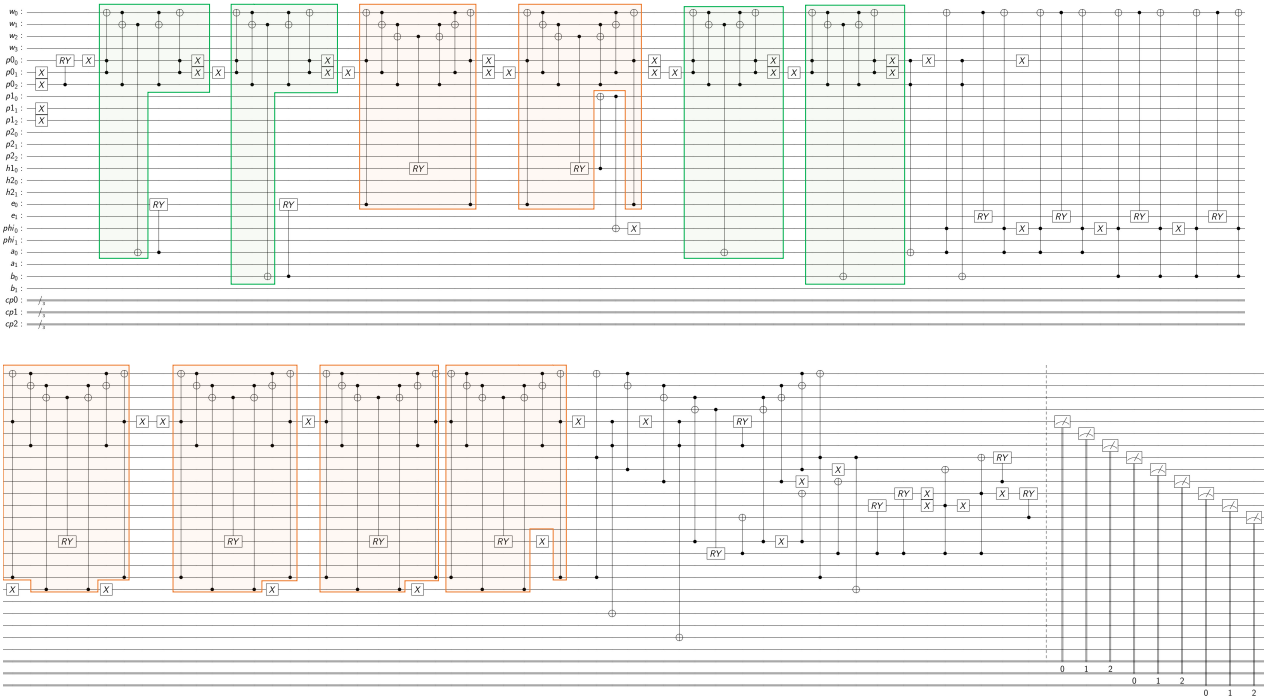


図 A.4　QPS の 2 ステップシミュレーション回路において探索された 2 つの RSG（緑色と橙色の背景がかかっているゲート群）。

図 A.4 を見ると二つとも制御ゲートがトフォリゲートと二量子ビットゲートに分解された形が探索されている。これらの RSG は非常に多くの CNOT と量子ビット間の結合を必要としているため、非常に重要な RSG であることがわかる。よってもし 2 つの RSG に特化した量子回路最適化、トランスパイル、パルス制御レベルでの最適化を行うことが出来れば大きな改善が見込まれる。

## A.3　RSG を用いたトランスパイル

RSG を用いた新しいトランスパイルを行う場合には、トランスパイルする前の量子回路に対して RSG 探索をする必要がある。トランスパイル後に RSG 探索をした時の改善の可能性についての議論は **??**で行う。他にも量子回路最適化の前後どちらで RSG 探索を行うかによっても異なる。量子回路最適化前に RSG 探索を行うと殆どの量子アルゴリズムに含まれるオラクルのような RSG が見つかりやすい。しかし AQCEL のように初期状態に依存した最適化後に RSG 探索を行うと、回路の前半においては不要な制御操作が削除されゲート群が変形されるため RSG 判定を受けない。もちろん最適化による CNOT 数の削減の恩恵を受けるが、RSG として扱われないため RSG に特化したトランスパイルの恩恵（量子ビットのマッピングの改善による swap の減少など）を得ることができない。このトレードオフの関係は今後調べる必要がある。

よって AQCEL による最適化が RSG 探索の前後かによって次の 2 つのレベルの最適化を考える。

**Level 1**： 回路全体を最適化
**Level 2**： RSG に対しては連続するゲートペアの削除のみ、他全体に対しては最適化

これを用いてトランスパイルにも以下のような 3 つのレベルを考えることができる。

**Level 1**： 最適化レベル 1→ 既存のトランスパイル
**Level 2**： RSG 探索 → 最適化レベル 1→ RSG を用いた新しいトランスパイル
**Level 3**： 最適化レベル 1→ RSG 探索 → RSG を用いた新しいトランスパイル