Vol.2022-QS-6 No.18
2022/7/8

誤り耐性量子計算におけるSELECT回路の並列化と高速化

鈴木 泰成[1,2,a)]　冬鏡 澪[3,1,b)]

概要：初期の誤り耐性量子計算機で計算速度での量子優位性を示すには、利用する量子アルゴリズムや回路への分解が計算機の設計や状況に最適化されていることが望ましい。特に、SELECT と呼ばれる操作は初期に応用が見込まれている Qubitization に基づく位相推定でボトルネックとなる構造であり、最適化する需要が大きい。しかし、SELECT 操作を計算機で利用できるリソースの状況に応じて調整可能な形で実現する回路のデザインは知られていなかった。本研究では並列化により計算機の状況に合わせて速度とリソース消費の速度を変更可能な DistSELECT 回路を提案する。この原稿では DistSELECT を解析的/数値的に性能を見積もり、隣接相互するスピン系のハミルトニアンを少ない命令のストールでエンコードする方法を議論する。さらに、複数のチップに分散された計算における論理ビット割り当ての最適化についても議論する。こうした解析を通し、本手法は論理量子ビットの数をハミルトニアンの項の数に対して増やすことにより、誤り耐性量子計算の実行速度を特徴づけるいくつかのパラメータを反比例的に小さくすることができることを明らかにする。

Parallelization and optimization of SELECT operations in fault-tolerant quantum computing

Abstract: Adaptive optimization of quantum circuit decomposition according to the status of a fault-tolerant quantum computer is essential for demonstrating quantum computational advantage in an early stage. In particular, SELECT operations, which are a core part of the state-of-the-art construction of quantum phase estimation, are known as the bottleneck of a wide range of applications, and there is strong demand to optimize them as an efficient and tunable form. In this paper, we propose a tunable, efficient, and easy-to-distribute design of SELECT operation, which we call DistSELECT. We analyze and numerically evaluate the performance of DistSELECT with several metrics. We also discuss their efficient implementation for interacting two-dimensional spin systems in the cases of a single node and distributed systems. We reveal that the proposed construction reduces reaction times and non-local operations by modest increase of the ancillary logical qubits.

1. 導入

ノイズの無い量子計算はいくつかのタスクで従来の計算機に比べ指数関数的な高速化を実現すると期待されている [1], [2], [3]。重ね合わせ状態がノイズに脆弱であることに起因して現実の量子計算機の誤り率は通常の計算機に比べ非常に大きいものの、個々の誤り率が一定以下であれば量子誤り訂正符号による符号化で論理量子ビットの誤り率を任意の値まで小さくすることができる。表面符号の上で行うことができる操作は符号による制約を受けるが、魔法状態蒸留と格子手術を用いることで任意の局所量子操作を符号上で効率的に近似できる [4], [5], [6], [7]。量子誤り訂正はオーバーヘッドの大きい技術であるため、初期の誤り耐性量子計算機はサイズや利用可能な誤り率に制約が生じると予想される。このため、量子アルゴリズムやその回路実装は誤り耐性量子計算機のリソース制約などに最適化されて設計されるべきである。比較的小規模な量子計算機で実行できて、実用的で、かつ計算量的な優位性が示せると期待されている量子アルゴリズムに、ハミルトニアンの基底状態のエネルギー推定を求める量子位相推定がある [8], [9], [10]。Qubitization という手法に基づく量子位相推定は後述する SELECT と PREPARE と呼ばれる要素の繰り返しで主に構成されている。誤り耐性量子計算機においては non-Clifford gate が特にオーバーヘッドの大きな操作

[1] NTT コンピュータ&データサイエンス研究所
[2] JST さきがけ
[3] 東京大学 工学系研究科 物理工学専攻
[a)] yasunari.suzuki.gz@hco.ntt.co.jp
[b)] tokami@qi.t.u-tokyo.ac.jp

© 2022 Information Processing Society of Japan

であることが知られているため、SELECT と PREPARE の操作を出来るだけ少ない non-Clifford gate で実現する方法が探求されてきた。その成果として、ハミルトニアンの項の数や求めるエネルギーの桁数に対して線形な数で実現する方法が提案されている [3], [11], [12]。しかし、誤り耐性量子計算の実行効率は non-Clifford gate 以外を含む様々な要素から定まるため、実際の計算では non-Clifford gate の数を最小化する構成が計算時間の観点で最適であるとは限らない。従って、より効率的な誤り耐性量子計算の活用のためには、誤り耐性量子計算における回路の実行で高速化の障害となるボトルネックの整理と、計算機のリソース状況に応じバランスの取れた SELECT や PREPARE 回路の設計が重要となる。

本研究では特に量子計算機の初期活用でボトルネックになると見込まれる SELECT 操作に焦点を絞り、この処理を低負荷に並列化することで回路の深さを大幅に削減する DistSELECT 回路の設計を提案する。提案する手法を用いると、例えば 1024 スピンの係数が均質なハイゼンベルグ模型において、論理量子ビットの数を 1.7 倍にし、全体のゲート数や non-Clifford gate の数をそれぞれ 2.4%、0.37% 増加する代わりに、回路の深さを 28 倍、non-Clifford gate の深さを 32 倍改善することができる。また、この SELECT 回路を複数の量子通信路で繋がれたチップ間で実装した際にも分散処理に必要な量子もつれの数はエネルギー推定を行うハミルトニアンが 2 次元的なスピン系の隣接相互作用によるものであるなら、スピン数に対して二乗根に抑えることができ、素朴な実装に比べて多項式的に改善できる。また、我々の提案する回路は追加で利用できる論理量子ビット数に応じて並列化の度合いを調整可能なため、複数の誤り耐性量子計算のノード間でコンパイル時に実機に適した回路合成が可能になるだけでなく、単一のノードを複数のユーザがシェアしたり量子もつれや魔法状態の供給が不安定である場合に、リソース状況に応じて動的に計算速度を調整するのに使うこともできる。

本予稿では DistSELECT の背景となる SELECT 操作の概要と、DistSELECT 操作の設計について解説する。また、誤り耐性量子計算における計算モデルを定義し、そのモデルにおいて計算機のパイプラインが破綻する要因を列挙し、これらの観点について解析や数値計算を通して性能評価を行う。

2. 背景

2.1 表面符号を用いた誤り耐性量子計算

表面符号 [4], [5] は 2 次元平面に埋め込み可能な量子誤り訂正符号であり、論理量子ビットは図 1 のようにデータを保持する量子ビットとパリティを検査する補助量子ビットを交互に並べて構成できる。計算中は補助量子ビットを用いてデータ量子ビットにスタビライザー測定と呼ばれる測定を適用し続けることでエラーを逐次的に検出、推定することができる。推定されたエラーは直接はフィードバックされず、パウリフレームと呼ばれる補正予定のパウリ演算を蓄積したテーブルを更新し後続の測定結果の補正に利用される [13]。図 1 におけるパッチ一つが 1 論理量子ビットの自由度があり、パッチの幅は符号距離に対応する。以降は量子ビットが 2 次元的に並んだチップをプレーンと呼び、論理量子ビットはプレーン上のブロックとして確保されるものとする。

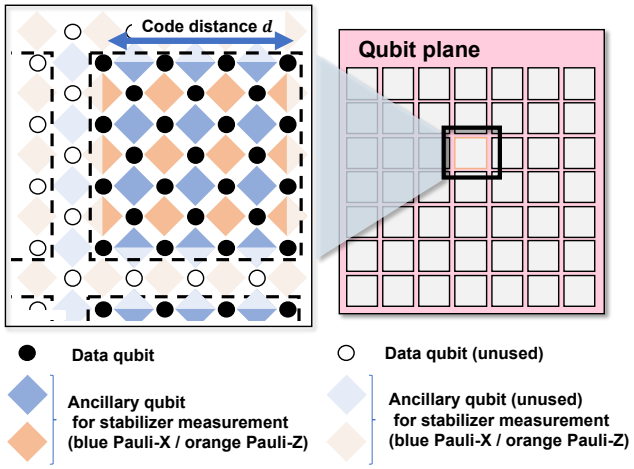

図 1 表面符号の配置例

我々は表面符号で符号化された論理ビットの上で、初期状態 $|0\rangle, |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), |A\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle)$ の生成、論理パウリ X, Y, Z 測定、論理 Hadamard gate$H$、複数の論理量子ビットに対するパウリ X^t, Z^t 測定を誤りに耐性がある形で行うことができる [6], [7], [14], [15], [16], [17]。さらに、二つの離れたプレーンの間に一定の品質の量子通信路があれば量子もつれ状態 $|\Phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle|11\rangle)$ を作ることもできる。これらの組み合わせで、我々は複数のプレーンの間で任意の定数量子ビットの量子ゲートを誤りに耐性のある形で行うことができる [7], [18], [19]。例えば CNOT gate は多体パウリ測定と Hadamard gate の組み合わせ、S gate は Y 測定を用いたゲートテレポーテーション、T gate は $|A\rangle$ を用いたゲートテレポーテーションで実現できる。この時、T gate を行うために消費される状態は $|A\rangle$ は魔法状態と呼ばれる。また、Toffoli gate は T gate7 個と Clifford gate に分解できる。なお、論理パウリ操作は先述のパウリフレームに補正を入れることで実現できるため、具体的な操作は不要である。

上記基本操作のうち、$|0\rangle, |+\rangle$ の生成と論理パウリ X, Z 測定以外の操作は操作の過程で符号距離 d に比例した回数スタビライザー測定を行わなければならない [15]。量子操作のレイテンシを特徴づけるスタビライザー測定を一回行う時間を code cycle、d code cycle の時間を 1 code beat と呼ぶ [10]。大まかな目安として code cycle は 1μs、符号距

© 2022 Information Processing Society of Japan

離 d は数十程度、例えば 21 μs 程度となると予想される[*1]ため、相対的に定数サイクルで終わる操作のレイテンシは無視することができる [20]。また、$|A\rangle$ と $|\Phi\rangle$ を高い忠実度で得るには注入と蒸留という操作を経る必要があるが、これらの操作の成功は一般に確率的であることが分かっている [7], [16]。

2.2 量子位相推定

$2^n \times 2^n$ のユニタリ行列 U の固有値と固有状態を $\{|\lambda_i\rangle, e^{i\theta_i}\}$ とする。また、2^n 次元の複素ベクトルを $|\psi\rangle = \psi_i |\lambda_i\rangle$ とする。量子位相推定は、与えられた行列 U の固有位相 θ_i を、$|\psi_i|^2$ の確率で $O(2^{-m})$ の精度でサンプリングする量子アルゴリズムである。量子位相推定を実行するには以下の二つが必要となる。第一に、$|\psi\rangle$ を自明な状態、例えば $|0\rangle^{\otimes n}$ から生成する回路が必要となる。第二に、制御ユニタリゲート control-U^{2^k} を実行する回路が必要となる。ここで、k は 0 以上 $m-1$ 以下の整数である。量子位相推定で必要となるゲート計算量は、状態生成にかかるコストを N_S、control-U^{2^k} の生成に必要なコストを $N_{U,S}$ とすると、m 桁の精度で値を知るために必要なコストは典型的には $O(N_S + \sum_{k=0}^{m-1} N_{U,S})$ となる。[*2]

量子位相推定の典型的な応用は、我々が興味がある物質のハミルトニアンを H としたとき、最大固有値が 1 以下になるよう規格化したハミルトニアンを $\tilde{H}$ として、$U = e^{i2\pi\tilde{H}}$ を構成し、さらに基底状態に十分近い状態 $|\psi\rangle$ を入力することで、ハミルトニアンの基底エネルギーを求めるものである。

2.3 Qubitization

Qubitization はハミルトニアン $H = \sum_{i=0}^{d-1} \lambda_i |\lambda_i\rangle\langle\lambda_i|$ の固有エネルギーを量子位相推定で計算するときに必要な演算子を Block encoding による方法で構成する手段を与える [3]。Block encoding とは、対象となるエルミート演算子 $H \in \mathcal{L}(\mathcal{H}_2)$ が[*3]、より大きな空間に対するユニタリ演算子 $U \in \mathcal{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$ とベクトル $|G\rangle \in \mathcal{H}_1$ について、$(\langle G| \otimes I)U(|G\rangle \otimes I) = H$ となっているような演算子 U とベクトル $|G\rangle$ を指す。以降では特に、U が $(\langle G| \otimes I)U^2(|G\rangle \otimes I) = I$ である場合を考える。[*4]この時、$|v_i\rangle := |G\rangle|\lambda_i\rangle$ と定義すると、

$$U_S |v_i\rangle = \lambda_i |v_i\rangle + \sqrt{1-\lambda_i^2}\,|w_i\rangle \tag{1}$$

となる。ここで $\{|v_i\rangle, |w_i\rangle\}_i$ は $2\dim(\mathcal{H}_2)$ 次元のベクトル空間 $V = \mathrm{span}(\{|v_i\rangle, |w_i\rangle\}_i)$ の正規直行基底となっており、U_S は全ての i について $\mathrm{span}(\{|v_i\rangle, |w_i\rangle\})$ を不変部分空間として持つ。従って、その作用は以下のような直和の形式で表現できる。

$$U_S = \bigoplus_{i=0}^{d-1} (\lambda_i Z + \sqrt{1-\lambda_i^2}X) \tag{2}$$

また、$|G\rangle$ に関する反射操作を $U_G = (2|G\rangle\langle G| - I)$ と定義すると、この操作は空間 V において $U_G = \bigoplus_{i=0}^{d-1} Z$ と振舞う。従って、$W = (U_G \otimes I)U_S$ とすると、

$$W = \bigoplus_{i=0}^{d-1} \exp(i \arccos(\lambda_i) Y) \tag{3}$$

である。すなわち、V の空間において操作 W と $|\lambda_0\rangle$ を入力として量子位相推定を行うと $\theta_i = \pm\arccos(\lambda_i)$ のいずれかを効率的に得る。どちらの値を位相として得た場合も、その余弦を計算すれば $\cos(\theta_i) = \lambda_i$ が得られる。

Qubitization に基づく量子位相推定は従来の Trotterization などに基づく実装よりも計算量理論的に軽量であるため、初期に実用的な分野で量子優位性が見込める量子アルゴリズムの一つである [8], [10]。また、Qubitization を拡張した量子特異値変換は多くの他の量子アルゴリズムを包含することが知られているため、Qubitization の最適化は幅広いアプリケーションの高速化にもつながる [12]。

2.4 Linear Combination of Unitary

Linear combination of Unitary (LCU) は、ハミルトニアンがエルミートかつユニタリな演算子 P_l の正の実数係数 α_l の線形結合として $H = \sum_l^L \alpha_l P_l$ のように表せるとき、これを用いて Block encoding した U および $|G\rangle$ を構築する方法を与える枠組みである [3]。以降では簡単のために項の数が $L = 2^{n_c}$ と累乗で書けるケースを考える。H が n_t-qubit に作用する 2^{n_t} 次元のエルミート行列であるような場合は任意のエルミート行列はパウリ行列の和に正係数で展開できる。この時、以下のような操作と状態 U_S と $|G\rangle$ を定義できる。

$$U_S = \sum_{l=0}^{L-1} |l\rangle\langle l| \otimes I \otimes P_l \tag{4}$$

$$|G\rangle = \sum_{l=0}^{L-1} \sqrt{\frac{\alpha_l}{\gamma}}\,|l\rangle|g_l\rangle \tag{5}$$

ただし、$|G\rangle$ は、U_S の最初の二つのテンソル積の空間におけるベクトルである。また $\gamma = \sum_l \alpha_l$ は規格化係数であり、$|g_l\rangle$ は l に依存した規格化された任意のベクトルである。上記の定式化は $U_S^2 = I$ を満たし、また、$(\langle G| \otimes I)U_S(|G\rangle \otimes I) = H/\gamma$ であることが確認できる。U_S のように第一レジスタがランク 1 の射影でありそれ以外のレジスタがユニタリとなる構造を持った操作は量子回路で

[*1] 符号距離は利用するデバイスや実現可能な物理エラーレートに強く依存する
[*2] 量子フーリエ変換などにかかるコストは相対的に無視できるとしている。
[*3] $\mathcal{L}(\mathcal{H})$ はベクトル空間 $\mathcal{H}$ の線形演算子を表す
[*4] 後述する Linear combination of unitary ではこの性質を満たす。また、満たさない場合でもこの条件を満たすよう調整できる。

© 2022 Information Processing Society of Japan

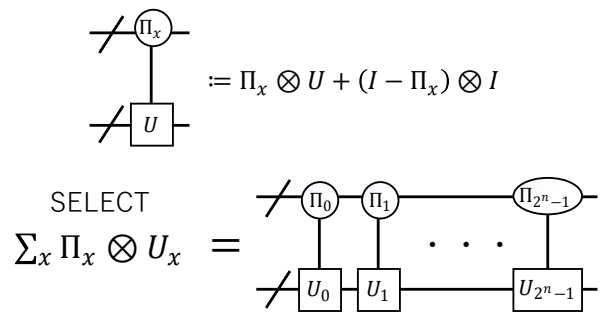

図 2 SELECT 回路の表現　ただし、Π_x は整数 $0 \leq x < 2^n$ に対して定義される演算子で $\Pi_x = |x\rangle\langle x|$ と定義される。

書くと図 2 のようになり、SELECT 操作と呼ばれる。また、$|0\rangle$ などの自明な状態から状態 $|G\rangle$ を生成するユニタリ U_P を PREPARE 回路と呼ぶ。PREPARE 回路を用いれば R_G も容易に構成できるので、$W = ((2|G\rangle\langle G| - I) \otimes I)U_S$ に対して位相推定を行った値の余弦を取ることで規格化されたエネルギー λ_i/γ を知ることが出来る。従って、$\log \gamma$ 桁だけ余分に計算の精度を確保すれば λ_i が所望の精度で得られる。

2.5 SELECT の設計

図 2 にあるように SELECT 操作は n_c-qubit の制御ユニタリを $L = 2^{n_c}$ 個繋げた積に分解できる。n_c-qubit の制御は $(n_c - 1)$ 個の Toffoli を用いて実現できるため、この実現には素朴には $O(L \log L)$ の魔法状態が必要となるが、実際には図 3 (a) に示すするように制御の値を出来るだけ Toffoli を再利用するように構成しなおすことで $O(L)$ の魔法状態まで減らすことが出来る [8]。この回路は以下のように説明される。ビット列 $x \in \{0,1\}^n$ に対して $x_{a:b}$ はビット列 x の a 桁目から b 桁目までの区間を抽出したものであるとする。また以降は混乱が無い限り $\{0,1\}^n$ を 0 以上 2^n 未満の整数の 2 進数表現と同一視する。ここで、関数 $C : \{0,1\}^n \times \{0,1\}^n \mapsto \{0,1\}^{n-1}$ を以下のように定義する。

$$C(x,y)_i = \begin{cases} 1 & x_{0:i+1} = y_{0:i+1} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

図 3 において Initialize は $|x\rangle|0\rangle \mapsto |x\rangle|C(x,0)\rangle$ を実現する操作で $O(n_c)$ 個の Toffoli で実現できる。さらに、$|x\rangle|C(x,y)\rangle \mapsto |x\rangle|C(x,y+1)\rangle$ とする操作を $U_{\text{inc},y}$ とすると、これは y の 0 となっている最も下位の桁[*5]を $z(y)$ としたとき、$2z(y)$ 個の Toffoli gate で実現できる。この時ビット $C(x,y)_{n-1}$ は $x = y$ となっているときのみ 1 となるので、これを制御ビットとして $|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes P_l$ を $0 \leq l < 2^{n_c}$ について作用すれば SELECT を実現できる。Finalize は $|x\rangle|C(x,2^\mu - 1)\rangle \mapsto |x\rangle|0\rangle$ と uncompute する操作である。この過程で必要となる Toffoli gate の数は $O(n_c + \sum_{y=0}^{2^{n_c}-1} z(y)) = O(2^{n_c})$ である。

[*5] ケット表記で一番右の桁を下位とし、桁は 0 桁目からカウントする

2.6 PREPARE の設計

PREPARE の操作は SELECT 回路の特殊系である Data lookup また QROM と呼ばれる回路を適用することで、パウリの項の数などに対して線形な魔法状態の数で実現することが出来る [8]。Data lookup 回路とはある関数 $f : \{0,1\}^n \to \{0,1\}^n$ について、$|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$ とする操作である。例えばハミルトニアンについて位相推定する場合、制御量子ビット数が n_c、対象量子ビット数が $n_c + n_e$ の Data lookup が PREPARE が必要となる [8]。ただし、$n_e = \lceil \log \epsilon^{-1} \rceil$ であり、ϵ は求める基底エネルギーの精度である。Data lookup 回路は作用の対象となる量子状態が $|0\rangle$ であることが分かっている SELECT の特殊系であるためさらなる効率化が可能である [21], [22]。具体的には 3 (b) のように取りうる状態を先に作って起き、control-SWAP を繰り返すことで第二レジスタに $f(x)$ が来るように制御できる。この時必要な Toffoli の数は $n_t 2^{n_c}$ と増えてしまうが、$n_t 2^{n_c}$ この補助量子ビットを要求する代償としてその測定深さを n_c まで下げることができる。この操作を SWAP Data-lookup と呼ぶ。さらに、SELECT 操作と SWAP を組み合わせることで補助量子ビットを用いることで必要な魔法状態の数をさらに減らす SelSWAP なども提案されている [21]。

PREPARE の計算量は準備すべき状態ベクトルの構造に強く依存する。もしハミルトニアンの係数が均質であるなど状態準備において効率的に実行できるような要素を持っているとき、この計算量はさらに削減することができる。例えばハミルトニアンの係数が全て同じ値である場合、要素数 L が 2 の累乗なら PREPARE は Hadamard gate のテンソル積として実現でき、L が 2 の累乗でない場合でも $O(n_c + n_e)$ 個の T gate で実現できる [8]。

3. DistSELECT の構成

Qubitization や量子特異値変換に基づく位相推定の効率は SELECT 回路や Data-lookup 回路の効率に還元できる。特に量子計算機での演算が得意な準備する状態ベクトル $|G\rangle$ が等しい重みの重ね合わせであるケースでは SELECT が全体のボトルネックとなる。従って、早期の誤り耐性量子計算機での量子優位性の証明を目指すには、SELECT 回路のシーケンスを出来るだけ高速に処理できることが望ましい。本研究では計算機の状況に合わせて誤り耐性量子計算機のリソース供給を最適化できるよう調整された SELECT 回路である DistSELECT を提案する。

この章では誤り耐性量子計算で効率的に動作する DistSELECT の構造を解説し、その性能の解析を行う。このために、まず 3.1 では誤り耐性量子計算における命令の

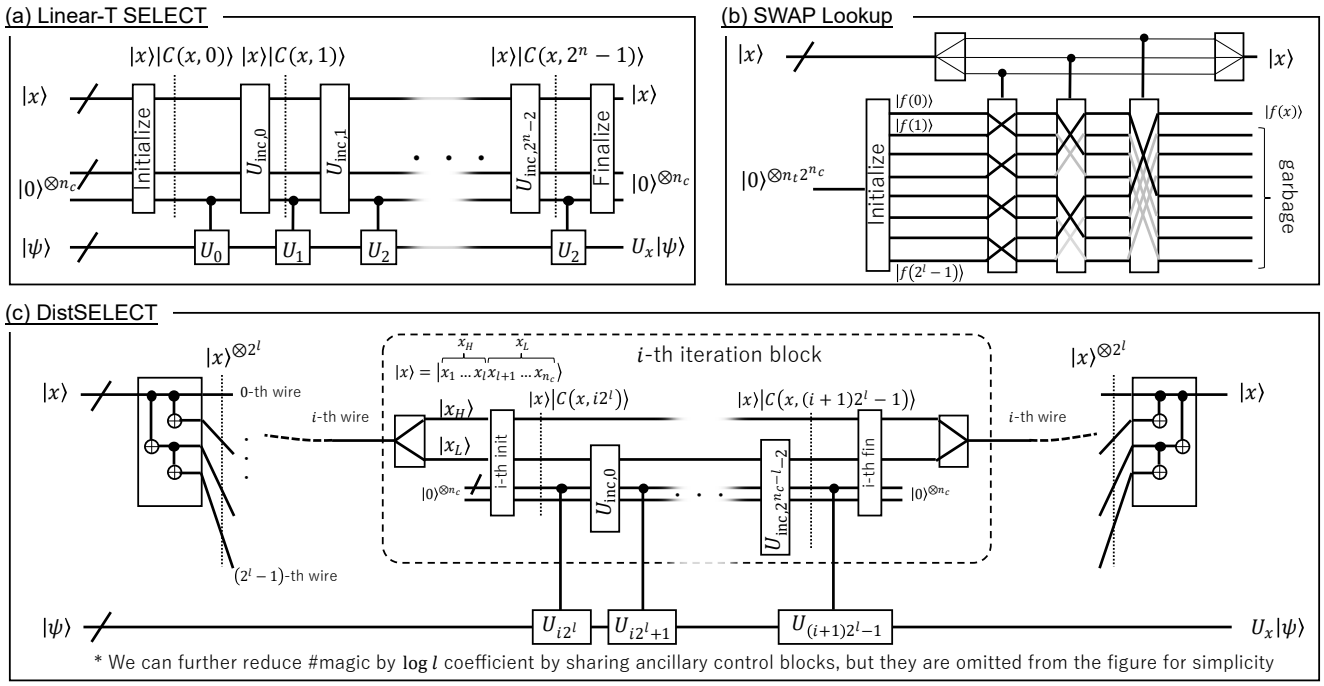

図 3 種々の SELECT 回路の構成　a), b), c) の構成はそれぞれ 1 の 2), 3), 6) に対応する。本研究で提案する構造は c) である。

実行速度を決めるいくつかの指標について解説を行い、その後にこれらの指標を最適化する手法として DistSELECT の構成や性能について解説する。

3.1 計算時間の見積もり

誤り耐性量子計算機が実行されるとき、次のような設計を考える。まず、誤り耐性量子計算のプログラムは節 2 で述べたような基本的な命令まで分解される。分解された命令はまず、一次元的な配列として与えられる。計算の実行時、誤り耐性量子計算機は直近の Q 個の命令をバッファし、その中で実行可能なものを並列に実行していく。完了した命令はバッファから削除され、新たな命令が外部からプッシュされる。上記のプロセスで code beat あたりに消費可能な命令の数を命令のスループットと定義する。この時、理想的には、バッファに入っているすべての命令が並列に実行され高いスループットが実現される。ところが、実際には種々の理由から命令は並行には実施できず、バッファの中で待機状態となる。このように、何らかの理由でバッファ内部の命令が実行できないとき、命令がストールすると呼ぶ。本研究ではこのストールを引き起こす要因をデータ依存性、トポロジー制約、リアクションの待機、魔法状態の枯渇、量子もつれの５種に分類する。この５種のうちいずれかが支配的になっている場合、支配的な要因が計算の速度を事実上決定する。解説する５種類のハザードの要因を図示したのが図 4 である。以降では各要素の概要と、その要素が支配的となったときに時間を表す指標を解説する。

3.1.1 データ依存性

バッファ内部の複数の命令のオペランドが重複している場合、二つの命令はデータ依存性を持っており直列にしか実行できない。つまり、後続の命令が先行する命令の完了を待機しなければいけない。例えば、図 4 (1) のように Q1,Q2 に対する多体パウリ XX 測定と Q1,Q3 に対する多体パウリ ZZ 測定が連続して配置されているとき、後続の命令は先行する XX 測定の完了を待機しなければならない。パウリ XX 測定は 1 code beat のレイテンシであるため、結果として並列性が損なわれる。データ依存性の少なさ、すなわち、命令の潜在的な並列性を評価する指標として回路深さを定義することが出来る。回路深さは個々の命令をノードとし、オペランドに重複がある二つの続く命令の間に有向エッジを用いたグラフを考えたとき、このグラフで最も長い経路として定義される。

3.1.2 トポロジー制約

多体パウリ XX 測定など複数の論理ビットを対象とする論理操作を行うには、確保されていないプレーン上のパスを確保して二つの論理ビットの間を接続しなければならない。多体パウリ測定が行われる 1 code beat の最中はこのパスは利用中であり、他の用途に用いることは出来ない。また、Hadamard gate や Y 測定などの１論理ビットに作用する命令においても、一時的に論理ビットの空間を拡張する必要があるため周辺にある未使用領域を消費する。論理命令を実行しようとしたときに上記のような制約を満たすパスや周辺のみ使用領域が無い場合、二つの多体パウリ測定は実行することができない。このようなトポロジカル

© 2022 Information Processing Society of Japan

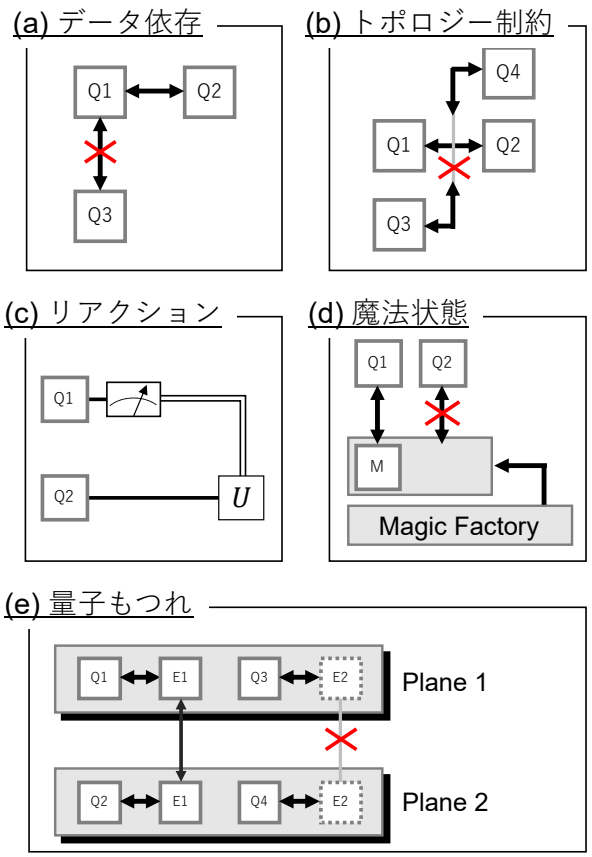

図 4 計算機に生じるハザードの類別 1) Q1 が既に利用中のため、Q1,Q3 に跨る命令が実施できない。 2) Q1,Q2 間のパスが利用中のため、Q3,Q4 間での格子手術が出来ない 3) 測定後のエラー推定が完了するまで、その測定結果に依存する操作を実行できない 4) 蒸留された魔法状態がバッファに無いため Q2 が魔法状態との格子手術を実施できない 5) 蒸留された量子もつれがバッファに無いため、異なるプレーンに属する Q3,Q4 の間で格子手術出来ない

な制約が、命令の並列性を損なう場合がある。例えば、図 4 (2) のように Q1,Q2 が二つをパスでつないで論理操作を行っているプロセスを表している。この時、Q1,Q2 の間を経由したパスで Q3,Q4 をつなぐことができないため、そのようなパスで接続する命令は Q1,Q2 の命令の完了を待機しなければならない。トポロジー制約によってプログラムがどの程度遅延するかは、論理量子ビットをどのようにプレーン上に設置するかに強く依存する。一般的には多くの補助量子ビットが必要になる場合にプレーンの利用可能な面積が圧迫されるためトポロジーの制約が強く効くが、具体的な影響については数値計算の 4 章にて具体的なスケジューリングを通して評価する。

3.1.3 リアクション

ある論理量子ビットを測定すると測定結果が得られる。後続の命令はその測定結果の値に依存して計算を行うことができる。表面符号に置ける誤りの推定は最小重みマッチング問題といったグラフの問題に帰着されるため、これを解いて推定するパイプライン全体は典型的に code beat よりも大きなレイテンシを要する [6], [23], [24], [25], [26]。従って、この測定値の値に応じて実行の詳細が決まるような命令は、論理測定の結果を待たねばならない。このためにかかる時間を Reaction time という。Reaction time は符号距離には依存するものの、主に量子計算機を制御する古典回路の性能で決まるため、その時間は code beat とは独立に決まる。回路深さのうち、測定を行ってパウリ以外の操作をフィードバックをする手続きのみの連鎖をカウントしたものを「測定深さ」と定義する。リアクションが必要なコストは、この測定深さに Reaction time を積算したもので見積もることができる。

論理パウリ測定の結果に基づいて演算を行わなければいけない典型的なゲートとして、CNOT gate、T gate、Toffoli gate などがある。このうち、Clifford gate という集合に属する CNOT gate はテレポートに必要なフィードバックが論理パウリ操作であり、これは Clifford gate の性質から後から遅延して補正を行うことができる。一方、T gate や Toffoli gate はフィードバックで必要となる操作がパウリ操作ではなく Clifford gate でないとならないためこうした遅延が出来ず、操作ができるようになるまで遅延が必須となる。従って、リアクションによるハザードは non-Clifford gate のゲートテレポーテーションで介在するハザードと言い換えることができる。

3.1.4 魔法状態の枯渇

魔法状態 $|A\rangle$ の生成には他の命令に比べ数倍〜10 倍程度長い時間を要する。従って、典型的には魔法状態を生成するファクトリと呼ばれる空間が専用で確保され、魔法状態を生成する。魔法状態はゲートテレポーテーションを行うと失われてしまうため、ファクトリの魔法状態の生成が計算が魔法状態を消費する速度に追いつかない場合、魔法状態を対象とした命令はストールしてしまう。図 4 (d) では魔法状態が枯渇しているため、Q2 が魔法状態に対するテレポーテーションを実施できない状態を表している。魔法状態の枯渇がどの程度潜在的に生じうるかは、回路が必要とする魔法状態の数から見積もることができる。

3.1.5 量子もつれの枯渇

単一のプレーンで十分な数の量子ビットを確保できないとき、複数のプレーンを量子通信で接続して分散された量子計算を行わねばならない。二つのプレーンに跨る量子操作を非局所操作とよび、非局所操作を行うためには二つのプレーンの間の量子もつれ状態 $|\Phi\rangle$ を消費する必要がある。しかし、魔法状態と同様に量子もつれ状態は低速かつ確率的にしか生成できないため、量子もつれが枯渇している際はテレポーテーションが実施できなくなる。図 4 (e) では、異なるプレーンに配置された Q1,Q2 は量子もつれ E1 を消費して非局所操作が出来ているが、Q3,Q4 は量子もつれが生成されていないため非局所操作が実行できない。量子もつれの必要数は、計算回路を有限サイズの複数のプレーン

© 2022 Information Processing Society of Japan

にマッピングしたとき、回路間でどの程度の非局所操作が必要となる数から見積もることができる。この値を最小化するためにどのように回路をマッピングするかは別の最適化問題となるが、その分配の戦略については 3.3 章で解説する。

3.2 DistSELECT 回路の設計

本研究では所与のパウリ行列の集合 $\{U_x\}$ に対して $U = \sum_x |x\rangle\langle x| \otimes U_x$ を行う DistSELECT 操作を提案する。以降では、第一レジスタの空間の次元を 2^{n_c}、第二レジスタの空間の次元を 2^{n_t} とする。LCU に基づくエンコードにおいては 2^{n_c} がハミルトニアンの項数、2^{n_g} がハミルトニアンの次元に対応する。DistSELECT 操作は $0 \leq n_c \leq n_c$ の整数を引数として回路を生成する。

DistSELECT の回路構造は図 3 (c) の通りであり、その設計の基本的な考え方は下記の通りである。通常の並列化計算では並列性を高めるため、ループ構文の添え字を並列度に応じてブロックに分割することが一般的に行われる。SELECT 操作では制御レジスタの値が 0 以上 2^{n_c} 未満のケースに応じて順番にループを行っている。従って、DistSELECT ではこのループを 2^l に分割し、i 番目のブロックが $i2^l$ から $(i+1)2^l$ 未満の添え字の処理を担当するように並列化する。ところが、3 (a) に見られるような U_{inc} により制御をインクリメントする処理を用いると、制御ビットが直列的な依存性を持つために測定深さを小さくすることができない。DistSELECT ではこの問題を制御ビットを必要に応じて CNOT を用いて重ね合わせ状態にし、Toffoli によるアドレスの AND 演算を並列化できるようにすることで解決している。DistSELECT から一部の最適化を除外したシンプルなモデルは 3 (c) に記載されている。ここでは量子ビットを $|x\rangle^{\otimes 2^l}$ にした後、i 番目のブロックで $|C(x, i2^l)\rangle)$ を構築する。その後、各ブロックはそれぞれが持つ $|x\rangle$ のレジスタを用いて U_{inc} を実施し、$|C(x,(i+1)2^l-1)\rangle$ の値まで更新を行う。最後にこれまでに生成した補助量子ビットとのもつれを全て逆演算で元に戻すことで計算を完了する。$U_{x\leq y}$ を x を 2^l で割ったあまりの値が y 以下である場合に U_x となり、そうでない場合に I となるような行列とすると、DistSELECT の計算過程は以下のようになる。

$$
\begin{aligned}
|x\rangle|\psi\rangle &\mapsto \left(\otimes_{i=0}^{2^l-1}|x\rangle|0\rangle\right)|\psi\rangle \mapsto \left(\otimes_{i=0}^{2^l-1}|x\rangle|C(x,i2^l)\rangle\right)|\psi\rangle \\
&\mapsto \left(\otimes_{i=0}^{2^l-1}|x\rangle|C(x,i2^l)\rangle\right)(U_{x\leq 0}|\psi\rangle) \\
&\mapsto \left(\otimes_{i=0}^{2^l-1}|x\rangle|C(x,i2^l+1)\rangle\right)(U_{x\leq 0}|\psi\rangle) \\
&\mapsto \left(\otimes_{i=0}^{2^l-1}|x\rangle|C(x,i2^l+1)\rangle\right)(U_{x\leq 1}|\psi\rangle) \\
&\cdots \\
&\mapsto \left(\otimes_{i=0}^{2^l-1}|x\rangle|C(x,i2^l+2^l-1)\rangle\right)(U_x|\psi\rangle) \\
&\mapsto \left(\otimes_{i=0}^{2^l-1}|x\rangle|0\rangle\right)(U_x|\psi\rangle) \mapsto |x\rangle(U_x|\psi\rangle)
\end{aligned}
$$

各ブロックでは $|x\rangle$ の下位 $n_c - l$ ビットに対してのみ Toffoli gate による AND 演算を行い条件付きの制御パウリ操作を行う。この時各ブロックにおけるインクリメントの測定深さと魔法状態の数はどちらも $O(2^{n_c-l})$ である。この操作が 2^l 個並列して行われるため、測定深さはそのままに魔法状態の数は 2^{n_c} となる。また、各ブロックでの $|C(x,i2^l)\rangle$ の生成と uncomputation には l の値に依存せず $O(n_c)$ の操作が必要となる。この値は魔法状態数に対しては無視できる値だが、$l \sim n_c$ での測定深さを考える場合には無視できない値となる。上記の過程では $|x\rangle$ の値の重ね合わせ状態を作るために多くの CNOT を行う必要がある。この処理は 2^l 個の n_c-qubit のブロックに木構造状に CNOT を並列して行うことで、$n_c 2^l$ の数の CNOT を l の non-Clifford gate の深さで行うことができる。

3.3 複数チップでの分散処理に必要なコスト

SELECT 操作を複数の量子ビットが集積化されたチップを用いて行うことを考える。この時、チップごとの役割の割り当てについて複数の設計を考えることができる。

(1)（生成と計算での分割）プレーン 1 では魔法状態生成以外の全ての処理を行い、それ以外のプレーンでは魔法状態生成のみを行う。

(2)（機能での分割）制御量子ビットを配置するプレーンと、制御対象の量子ビットの配置するプレーンで分ける

(3)（制御対象の分割）n_c の制御量子ビットを DistSELECT で各プレーンに配置し、各プレーンが U_{ib} から $U_{(i+)b}$ までの制御パウリ操作を担当する。

回路を見て明らかなように、量子回路上で最もアクセスが頻繁に生じるのは各ブロックでの $|x\rangle$ および $|C(x,y)\rangle$ の下位ビットと、そこから制御される n_t の対象量子ビットの間の操作であり、その頻度は 2^{n_c} である。従って、1), 2) の方式を採用すると並列数に寄らず常に 2^{n_c} に比例する通信が必要となってしまう。一方で、3) に基づく処理を行った場合、ハミルトニアンが適切に分割されれば 2^{n_c} より大幅に小さいコストで分散が可能となる。従って、本稿では主に 3) について検討を行う。現実的な制約では 1,2 の方式での分散も考えられるが、これらとの比較は将来的な課題として残す。

n_t 量子ビットに作用するハミルトニアンを $H = \sum_i \alpha_i P^{(i)}$ と LCU でパウリ行列の線形和に分解できたとしたとする。i 番目のブロックはパウリ行列 P_i が非自明に作用する量子ビットの集合を w_i とする。この時、n_t 個の量子ビットを 2^m 個のノードに $A_1, ...A_{2^m}$ と分散して保持すると、j 番目のプレーンは $b = 2^{n_c-m}$ として $P^{(jb)}$ から $U^{((j+1)b)}$ の範囲の制御パウリ操作を担当することになる。このパウリが作用する量子ビットが自身のプレーンにない場合、量子もつれを消費して他のプレーンから一時的に自身

© 2022 Information Processing Society of Japan

のプレーンに移動しなければならない。j 番目のプレーンが他のプレーンから取り寄せる必要がある量子ビットの数は $\cup_{k=0}^{b} w_{jb+k} \setminus A_j$ であるので、我々は $\sum_j |\cup_{k=0}^{b} w_{jb+k} A_j|$ を最小化するように $\{w_i\}$ の割り当てや、$P^{(i)}$ の並びを決定しなければならない。

上記の最適化はある種のグラフ分割の問題に還元されるため一般のケースについて効率的に解くことは難しいが、対象となるハミルトニアンが分解しやすい構造を持っている場合は筋の良い分割と必要な量子もつれの個数を具体的に見積もることができる。例えば 2 次元の $\sqrt{n_t} \times \sqrt{n_t}$ 格子状に並んで隣接相互作用する n_t 個のスピンについては、2^m 個のノードで分散することで一辺が $\sqrt{n_t 2^{-m}}$ の正方形のブロックに分割できる。この時、j 番目のプレーンに j 番目のブロックに含まれる制御対象の量子ビットを割り当て、ハミルトニアンの項についてもブロック内部のものがそのプレーンの担当添え字になるように適当なパディングを入れて並び替えることができる。この時、ブロックの辺の長さに比例して隣接するブロックと通信をする必要があるが、そのコストは各ブロックについて $O(\sqrt{n_t 2^{-m}})$ であり、2^m 個のブロック全体で $O(\sqrt{n_t 2^m})$ と見積もることができる。この手続きは図 5 のようにリレー上に四方にテレポーテーションを行うことで実現できる。同様の考え方は隣接相互作用であれば多次元の場合にもストレートに適用することができる。

上記の設計に基づいて並列化を実行した場合、各 SELECT 操作は分散、計算、集約の順番に行われる。その過程を図示したものが図 6 である。$2^l \geq 2^m$ のケースを考えると、毎回 SELECT の操作を行うたびに n_c の量子ビットを各ノードに転送する必要があるため、最初と最後に $O(n_c 2^m)$ の通信コストが生じる。これに制御パウリを実施する際にテレポートが必要な量子ビットの数 E を加えたものが必要なコストとなる。

なお、3) の分割の考え方は、少なくとも各ブロックの制御部、すなわち、$2n_c$ 個の論理ビットが少なくも各プレーンの中に納まることを前提としている。これは、実用的な応用では n_c はハミルトニアンの項数の対数になる。量子超越性を示すには典型的に 10^3 程度の量子ビットに関する位相推定が行えれば十分であり、その対数は 10 程度であることから量子ビット数に対して 2,3 乗程度の項数があると思うと高々 50 論理ビット程度である。段数 1 の魔法状態蒸留を行うために約 24 論理ビット程度の論理ビットのスペースを用いることを踏まえると、量子超越性を示すことができる誤り耐性量子計算機において単一のプレーンがこの程度のサイズを許容すると考えるのは許容可能考えられる。

3.4 関連する研究との比較

DistSELECT の構造を、3.1 章の指標に基づいて評価し他

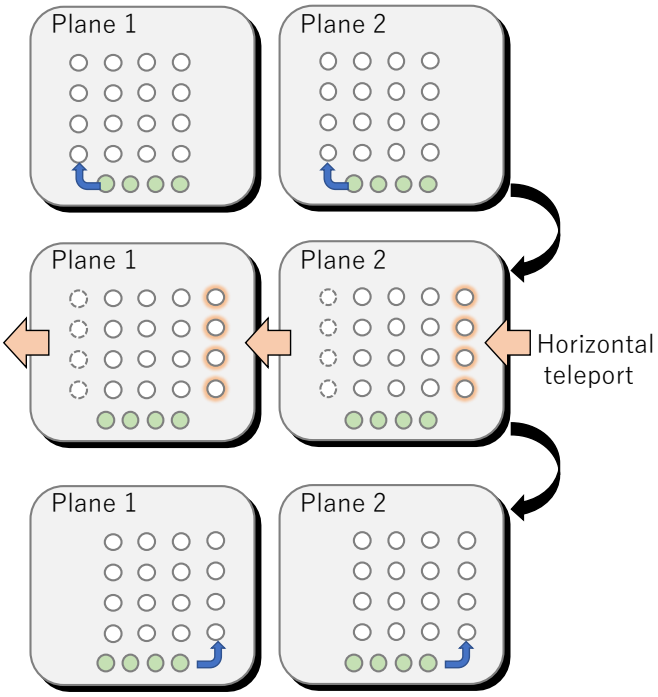

図 5 SELECT の制御対象となる量子ビットのテレポートの手続き。緑色の丸は分散された制御量子ビットを表し、白丸は制御対象となる量子ビットを表す。各プレーンは自身の担当の添え字のブロックについて青い矢印で示されるように制御パウリ操作を実行する。自身のブロックに無い量子ビットについては、隣接するプレーンから量子テレポーテーションで移動し制御パウリを実施する。

の SELECT や QROM の設計と比較したものが表 1 である。表において読み出し専用に〇がついているものは、対象となる量子ビットが 0 状態に初期化されている場合のみに利用できる手法であり、PREPARE のみに利用できる。本提案手法は必要となる補助量子ビットの数が l に関して指数関数的に増加するものの、魔法状態数を維持したまま測定深さを軽減することができる。また、複数のチップに対して分散処理を行うような場合でも、並列化する台数が 2^l より小さい限りはその非局所な操作数を $n_c 2^l$ とハミルトニアンを分割したときのカット数で抑えることができる。

4. 数値評価

4.1 SELECT 操作の評価

誤り耐性量子計算機で本提案を実装したときのコストを具体的に評価するため、本研究で提案する SELECT 回路を実際に構成し要求される性能指標を数値的に評価した。ここではハミルトニアンとして N スピンの隣接相互作用のハイゼンベルグ模型を採用し、その係数は全て一様だとした。係数が全て同一だとみなせるとき 2.6 章で述べたように PREPARE のコストは相対的に無視することができ、従って位相推定全体の計算時間はここで評価する SELECT の効率に比例するとみなせる。評価した結果は図 7 にプロットされている。この表において、x 軸は DistSELECT の上位

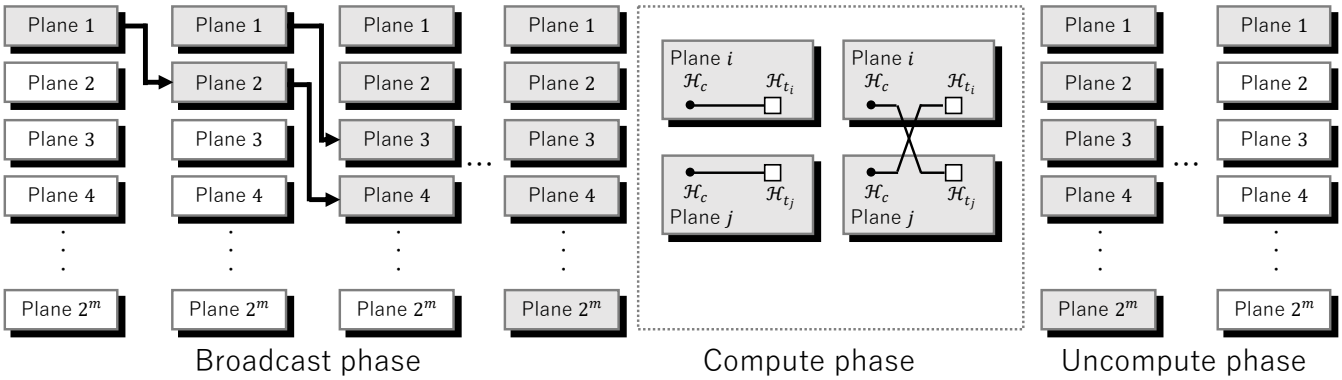

図 6 DistSELECT を分散配置したときに想定されるマッピング

手法	読み出し専用	補助量子ビット	魔法状態数	測定深さ	非局所操作数
1) Naive implementation		n_c	$n_c 2^{n_c}$	$n_c 2^{n_c}$	$2^{n_c} + E$
2) Sawtooth [8]		n_c	2^{n_c}	2^{n_c}	$2^{n_c} + E$
3) SWAP [21]	○	$n_t 2^{n_c}$	$n_t 2^{n_c}$	n_c	$n_t 2^{n_c} + E$
4) SelSWAP [21], [22]	○	$n_c - l + n_t 2^l$	$2^{n_c-l} + n_t 2^l$	$n_c + 2^{n_c-l}$	$2^{n_c-l} + n_t 2^l + E$
5) Predecode [27]		$2^{n_c/2}$	2^{n_c}	$2^{n_c/2}$	$2^{n_c} + E$
6) DistSELECT (提案手法)		$n_c 2^l$	2^{n_c}	$n_c + 2^{n_c-l}$	$n_c 2^m + E$

表 1 各制御機構の比較評価　評価値は支配的な項のみを記載し対数的に小さい項や定数係数は無視した。n_c は制御量子ビット数、n_t は対象量子ビット数、E は分散時のハミルトニアンの並列度に対応する。ただし、提案手法の非局所操作数ではノード数が 2^l より小さいとしている。

何ビットをブロック分割するかというパラメータ l である。$l = 0$ は分割を行わないケースに対応しており、その性能は文献 [8] の設計と一致している。図より事前に評価した通りパラメータ l を大きくし、論理量子ビットを増加させることで、Clifford gate や Toffoli gate の深さが指数関数的に減少することが分かる。特に大きいサイズのハミルトニアンを扱う場合は、追加の補助量子ビット数がハミルトニアン自体が作用する量子ビット数 n_t に比べ十分小さい範囲であれば l を大きくしても相対的なオーバーヘッドを小さく抑えられる。

4.2 トポロジー制約によるハザードの評価

3.1 章で列挙した値のうち、トポロジーの制約に基づく性能の劣化は実際のスケジューリングを通さなければ評価を行うことが難しい。そこで、この節では具体的な論理量子ビットのマッピングと貪欲法によるスケジューリングを通して、`DistSELECT` の実行時に生じるトポロジー制約によるハザードの影響を評価する。この数値計算では全ての論理量子ビットは単一のプレーンに収まるとしている。また、魔法状態のファクトリをどこに置くかによって定まる影響を排除しまたトポロジー制約によるハザードの影響のみを観測するため、T gate の命令によるレイテンシは簡単のために無視し、格子手術によるパスの干渉のみを調べた。魔法状態の配置も含めた全体的な最適化と評価は将来的な課題とする。プレーンには 31×31 の論理量子ビットを確保するブロックが一つ飛ばしに並べる形で配置し、論理量子ビットの配置は対象量子ビット、制御量子ビットの順番に左上から順番に割り振った。論理命令の順序付けは貪欲法で定めるとして、命令のキューのサイズは無限大とした。この時、`DistSELECT` 回路を実行する際の振る舞いを、様々な `DistSELECT` を特徴づけるパラメータ l に関して調べた。上記の条件で計算を行った場合に、8×8 のハイゼンベルグモデルについて、code beat ごとの処理命令数や累積処理命令数をプロットしたものが図 8 である。右図の累積命令数における点線はトポロジーの影響を無視した場合の処理命令数を表す。従って、点線から実線へどの程度処理速度が劣化するかから、トポロジー制約が量子計算機の速度低下に与える影響を見積もることができる。図より $l = 0$ のケース、すなわち文献 [8] の設計においてはトポロジーの制約は性能に影響を及ぼさないことが分かる。これは、$l = 0$ のケースではそもそも命令が直列的につながっており、データ依存性によるハザードが支配的でトポロジーの影響が生じないためである。一方、大きな l になるとデータ依存性が解消されるため、トポロジーの制約による速度低下が拡大することになる。

次に、様々なハイゼンベルグモデルのスピン数 N について、l の値を変化させながら `DistSELECT` 全体の実行に必要な code beat の数を計算した。その結果をプロットしたものが 9 である。図より、パラメータ l を大きくすることは SELECT の処理に必要な時間を削減することが分か

© 2022 Information Processing Society of Japan

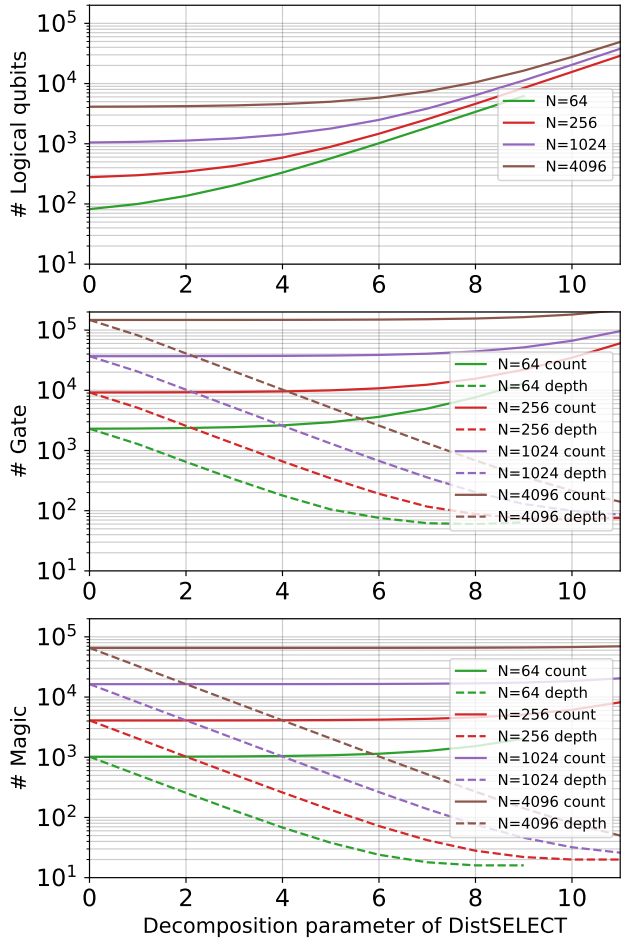

図 7 DistSELECT が必要とするコストの一覧

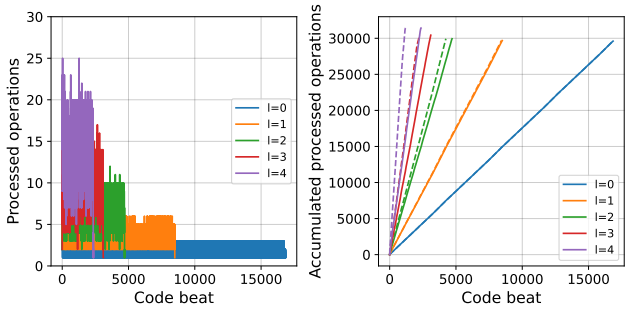

図 8 左図：DistSELECT 操作が各 code beat で処理する命令する。右図：累計で処理した命令数。点線はトポロジー制約を無視した場合の性能を表す。

る。また、スピン数 N に殆ど依存しない比率でトポロジー制約により処理の速度が低減していることが分かる。特に $l = 4$ ではおおよそ倍程度の速度低下が生じている。これは言い換えると量子ビットの最適なマッピングや命令のスケジューリングによって倍程度計算が早くなることを意味しており、こうした領域では命令の順序最適化が重要な意味を持つことを示唆している。

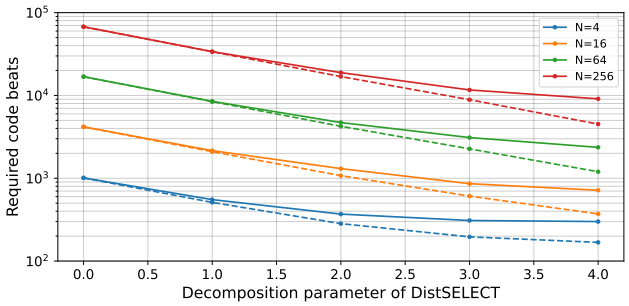

図 9 SELECT 回路を処理するのに必要な code beat 数。点線はトポロジー制約を無視した場合の性能を表す。

5. 結論と今後の展望

本研究では初期の誤り耐性量子計算で重要な役割を果たす SELECT 回路の効率を改善する設計を提案した。提案した SELECT 回路は分散処理において高い性能を発揮するだけでなく、パラメータのチューニングにより、必要な論理量子ビットの数を犠牲にして測定深さを改善することもできる。また、具体的に Qubitization の回路を分解した回路を題材として具体的に必要案サイクル数や数値計算によりその性能を評価した。これにより、提案手法が利用可能な論理量子ビットのサイズに応じて動的に高い性能を示すことを明らかにした。

本研究の今後の展開としては下記がある。現状の課題で述べたように、効率的に誤り耐性量子計算機のリソースを活用するには実行の状況に応じて動的に SELECT 回路の構成を決定できることが望ましい。一定でないリソースやハザードの状況に応じて今回提案した DistSELECT の構成を変更することで、より効率的な量子アルゴリズム実行が可能になると期待される。今回の数値計算ではトポロジー制約を評価するために、論理量子ビットが一つ置きに間を開けて配置し、貪欲法によるスケジューリングを行った。従来の設計では回路の直列性からこうした素朴な手法でもトポロジーの制約は無視できるが、DistSELECT で並列化を行い回路深さを削減すると、量子ビットの割り当てとスケジューリングが高速化の上で重要であることが明らかになった。魔法状態のファクトリの設置なども含め、効率的な割り当てやスケジューリングを考えるのも今後の課題となる。SELECT 回路においては制御論理量子ビットの上位ビットに相当する論理ビットなどはデータを保持するだけに利用され殆ど計算に使われていないことが分かる。演算子が作用しない論理量子ビットは長い周期のコードサイクルで誤り訂正を行うべきである。なぜなら、コードサイクルが長い量子誤り訂正は発熱が少なく、従って負荷が小さいからである。こうした動的なコードサイクルの変更を行うことで、誤り耐性量子計算機での発熱を大幅に軽減できると予想される。また、従来の研究により、誤り耐性量子

© 2022 Information Processing Society of Japan

計算機においても誤り抑制が適用できることが知られている [28]。従って、仮想２量子ビットゲートの枠組みを用いることでノード間のクリティカルな通信を削減したり、魔法状態の供給が追い付かないときに疑確率分解を用いることで大規模なストールを削減する手法を考えることもできる。こうした誤り耐性量子計算のコストを削減する技術を積み重ねることで、総合的に効率的な誤り耐性量子計算機が実現できると期待される。

参考文献

[1] Shor, P. W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM review*, Vol. 41, No. 2, pp. 303–332 (1999).
[2] Harrow, A. W., Hassidim, A. and Lloyd, S.: Quantum algorithm for linear systems of equations, *Physical review letters*, Vol. 103, No. 15, p. 150502 (2009).
[3] Low, G. H. and Chuang, I. L.: Hamiltonian simulation by qubitization, *Quantum*, Vol. 3, p. 163 (2019).
[4] Kitaev, A. Y.: Quantum computations: algorithms and error correction, *Russian Mathematical Surveys*, Vol. 52, No. 6, pp. 1191–1249 (1997).
[5] Bravyi, S. B. and Kitaev, A. Y.: Quantum codes on a lattice with boundary, *arXiv preprint quant-ph/9811052* (1998).
[6] Fowler, A. G., Mariantoni, M., Martinis, J. M. and Cleland, A. N.: Surface codes: Towards practical large-scale quantum computation, *Physical Review A*, Vol. 86, No. 3, p. 032324 (2012).
[7] Fowler, A. G. and Gidney, C.: Low overhead quantum computation using lattice surgery, *arXiv preprint arXiv:1808.06709* (2018).
[8] Babbush, R., Gidney, C., Berry, D. W., Wiebe, N., McClean, J., Paler, A., Fowler, A. and Neven, H.: Encoding electronic spectra in quantum circuits with linear T complexity, *Physical Review X*, Vol. 8, No. 4, p. 041015 (2018).
[9] Kivlichan, I. D., Gidney, C., Berry, D. W., Wiebe, N., McClean, J., Sun, W., Jiang, Z., Rubin, N., Fowler, A., Aspuru-Guzik, A., Neven, H. and Babbush, R.: Improved Fault-Tolerant Quantum Simulation of Condensed-Phase Correlated Electrons via Trotterization, *Quantum*, Vol. 4, p. 296 (online), DOI: 10.22331/q-2020-07-16-296 (2020).
[10] Lee, J., Berry, D. W., Gidney, C., Huggins, W. J., McClean, J. R., Wiebe, N. and Babbush, R.: Even More Efficient Quantum Computations of Chemistry Through Tensor Hypercontraction, *PRX Quantum*, Vol. 2, No. 3 (online), DOI: 10.1103/prxquantum.2.030305 (2021).
[11] Gilyén, A., Su, Y., Low, G. H. and Wiebe, N.: Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, ACM, (online), DOI: 10.1145/3313276.3316366 (2019).
[12] Martyn, J. M., Rossi, Z. M., Tan, A. K. and Chuang, I. L.: Grand Unification of Quantum Algorithms, *PRX Quantum*, Vol. 2, No. 4 (online), DOI: 10.1103/prxquantum.2.040203 (2021).
[13] Riesebos, L., Fu, X., Varsamopoulos, S., Almudever, C. G. and Bertels, K.: Pauli frames for quantum computer architectures, *Proceedings of the 54th Annual Design Automation Conference 2017*, pp. 1–6 (2017).
[14] Horsman, C., Fowler, A. G., Devitt, S. and Van Meter, R.: Surface code quantum computing by lattice surgery, *New Journal of Physics*, Vol. 14, No. 12, p. 123011 (2012).
[15] Litinski, D.: A game of surface codes: Large-scale quantum computing with lattice surgery, *Quantum*, Vol. 3, p. 128 (2019).
[16] Li, Y.: A magic state's fidelity can be superior to the operations that created it, *New Journal of Physics*, Vol. 17, No. 2, p. 023037 (2015).
[17] Brown, B. J., Laubscher, K., Kesselring, M. S. and Wootton, J. R.: Poking holes and cutting corners to achieve Clifford gates with the surface code, *Physical Review X*, Vol. 7, No. 2, p. 021029 (2017).
[18] Nielsen, M. A. and Chuang, I.: Quantum computation and quantum information, AAPT (2002).
[19] Ross, N. J. and Selinger, P.: Optimal ancilla-free Clifford+T approximation of z-rotations, *arXiv preprint arXiv:1403.2975* (2014).
[20] Gidney, C. and Fowler, A. G.: Efficient magic state factories with a catalyzed $|CCZ\rangle$ to $2|T\rangle$ transformation, *Quantum*, Vol. 3, p. 135 (2019).
[21] Low, G. H., Kliuchnikov, V. and Schaeffer, L.: Trading T-gates for dirty qubits in state preparation and unitary synthesis, *arXiv preprint arXiv:1812.00954*, (online), DOI: 10.48550/ARXIV.1812.00954 (2018).
[22] Berry, D. W., Gidney, C., Motta, M., McClean, J. R. and Babbush, R.: Qubitization of Arbitrary Basis Quantum Chemistry Leveraging Sparsity and Low Rank Factorization, *Quantum*, Vol. 3, p. 208 (online), DOI: 10.22331/q-2019-12-02-208 (2019).
[23] Edmonds, J.: Paths, trees, and flowers, *Canadian Journal of mathematics*, Vol. 17, No. 3, pp. 449–467 (1965).
[24] Holmes, A., Jokar, M. R., Pasandi, G., Ding, Y., Pedram, M. and Chong, F. T.: NISQ+: Boosting quantum computing power by approximating quantum error correction, *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, IEEE, pp. 556–569 (2020).
[25] Ueno, Y., Kondo, M., Tanaka, M., Suzuki, Y. and Tabuchi, Y.: QECOOL: On-Line Quantum Error Correction with a Superconducting Decoder for Surface Code, *arXiv preprint arXiv:2103.07526* (2021).
[26] Ueno, Y., Kondo, M., Tanaka, M., Suzuki, Y. and Tabuchi, Y.: QULATIS: A Quantum Error Correction Methodology toward Lattice Surgery, *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, IEEE, (online), DOI: 10.1109/hpca53966.2022.00028 (2022).
[27] Phalak, K., Alam, M., Ash-Saki, A., Topaloglu, R. O. and Ghosh, S.: Optimization of Quantum Read-Only Memory Circuits, *arXiv preprint arXiv:2204.03097*, (online), DOI: 10.48550/ARXIV.2204.03097 (2022).
[28] Suzuki, Y., Endo, S., Fujii, K. and Tokunaga, Y.: Quantum Error Mitigation as a Universal Error Reduction Technique: Applications from the NISQ to the Fault-Tolerant Quantum Computing Eras, *PRX Quantum*, Vol. 3, No. 1 (online), DOI: 10.1103/prxquantum.3.010345 (2022).

© 2022 Information Processing Society of Japan