量子振幅推定を用いた円周率推定法

能任　琢真
（技術研究所）

Quantum Circuit to Estimate PI Using Quantum Amplitude Estimation

Takuma Noto

量子コンピュータ用のアルゴリズムである量子振幅推定を活用した、モンテカルロ計算の高速化が期待されている。本研究は、モンテカルロ計算の代表的な問題である円周率推定を量子回路で作成し、古典計算との比較を行った。はじめに基本的な算術回路である量子積算回路をベースに2種類の量子二乗回路を提案し、さらに$4n+1$個の量子ビットで2^{2n}個のサンプリングを行う円周率推定回路を実現した。量子シミュレータでサンプル数と計算量を変化させ、古典計算と比較することでその特徴を考察した。

The quantum algorithm for the Monte Carlo method using quantum amplitude estimation will enable high-speed calculations. This study presents two types of quantum squaring circuits based on quantum multipliers and a quantum circuit for estimating the pi value using the squaring circuits and by quantum amplitude estimation. The quantum circuit for estimating the pi value was implemented in $4n+1$ qubits at 2^{2n} sampling. The quantum circuit was demonstrated using a quantum computer simulator by changing sampling numbers and computational complexity to compare with a classical calculation.

1. はじめに

量子コンピュータの発展により、多様な分野にわたって現在主流の古典計算機では到達できない高速計算が実現することが期待されている。量子コンピュータの基本要素である量子ビットは、0と1の情報を同時に保持することのできる「重ね合わせ」や量子論的な相関である「量子もつれ」といった古典ビットにはない特徴があり、これらを活用して、古典的なアルゴリズムに比べ計算量を劇的に減らせることが数学的に証明された量子アルゴリズムがいくつも報告されている。その1つである量子振幅推定(Quantum Amplitude Estimation: QAE)[1]を用いたモンテカルロ計算は、計算量を古典計算の平方根オーダーに減らせるアルゴリズムとして注目されており[2]、特に金融計算における活用を狙った研究が先行している。一方、モンテカルロ計算は、放射線輸送計算を始めとする様々な分野のシミュレーションや、AI、コンピュータグラフィックなど多くの分野で現在使われていることから、実用的な量子コンピュータが登場した際には幅広い分野で応用されることが期待できる。

近年の量子コンピュータは、将来実現が期待されている理想的な誤り耐性量子コンピュータに比べると、量子ビットが少なく、エラーが生じるため NISQ (Noisy Intermediate-Scale Quantum computer)と呼ばれ、それに適したアルゴリズムも検討されている。また、量子コンピュータの動作検証のためのシミュレータやワークフレームが普及しており、計算規模にもよるが30量子ビット程度までであれば、一般のパソコンでも計算が可能である。モンテカルロ計算においても、従来の量子振幅推定を使ったアルゴリズムは、量子振幅推定精度を上げるための長い量子回路と、量子フーリエ変換(QFT)の逆演算のための大量の量子ビットを必要としていたが、NISQ 向けに必要な量子ビットを減らし、ノイズがある状況でも動作するアルゴリズムも開発されている[3-5]。

本研究は、モンテカルロ計算の有名な練習問題である、円と正方形の面積比を使った円周率推定法の量子回路を組み、シミュレータで検証することで、古典計算との違いや、その特徴を考察した。なお、円周率推定を行うだけであれば、より効率的な量子アルゴリズムが存在するが[6,7]、本研究

はモンテカルロ計算の検証を行うため、古典的なアルゴリズムを量子回路で実装した。

古典的で複雑な数値問題を解くには、加算回路(Adder)や乗算回路(積和演算回路、MAC)といったいくつかの基本的な算術回路が必要になる。この時、古典回路では入力が2つ、出力が1つといったように、情報が消失する非可逆的な演算が一般的であるが、量子コンピュータでの操作は基本的にユニタリ演算であることが求められ、情報は失われず逆演算によりもとに戻る可逆性がある。

例えば、量子回路においては、加算回路は次式の動作が期待される。

$$|x\rangle|y\rangle \rightarrow |x\rangle|x+y\rangle \tag{1}$$

ここで$|x\rangle$, $|y\rangle$, $|x+y\rangle$ はそれぞれ整数値x, y, $x+y$を表す量子状態であり、2つの量子レジスタ*が入力され、2つの量子レジスタが出力されていることを表している。この操作は$|x\rangle$と$|y\rangle$が任意の整数の重ね合わせ状態であっても実行され、例えばxが0と1, yが2と4をそれぞれ等確率で測定できる状態の重ね合わせであれば、$x+y$として2, 3, 4, 5が等確率で観測される重ね合わせ状態が得られる。これまでに桁上げ伝搬加算回路(Ripple-Carry Adder)、桁上げ先見加算回路(Carry Look-ahead Adder)、さらにそれらの組み合わせなど、古典的なアルゴリズムに基づく量子回路が報告されている[8),9)]。例えば、桁上げ伝搬加算回路に基づくnビットの2進数整数の量子加算回路では、必要なゲート数が$O(n)$かつ補助量子ビットなしで実装可能な回路が提案されている[9)]。また新しいアルゴリズムとして、量子フーリエ変換(QFT)を用いた量子加算回路が提案されており[10)]、補助ビットが不要かつゲート数$O(n^2)$で実装可能と報告されている。

量子乗算回路もまた、種々の回路が報告されているが[11)-15)]、いわゆる筆算のように桁をずらして加算を繰り返し行う場合は、桁数に応じた量子加算回路が必要である[14)]。一方、QFT加算回路を発展させた量子乗算回路は、補助量子ビットなしで、ゲート数$O(n^3)$で実装が可能である[16)-18)]。

* 量子ビットの集まり

本論文が検証する円周率推定回路は、シミュレータで動作させるため、必要な量子ビット数の少ない量子二乗回路が適切と考え、回路の実装を行った。2章では量子二乗回路のベースとなった2種類の量子乗算回路を紹介し、その後に本研究で検証した2種類の量子二乗回路を示した。3章では、量子二乗回路を使って円周率推定回路を組み、シミュレータでの検証結果と、古典計算の比較及び考察を行った。

2. 算術回路

2.1 量子乗算回路

量子乗算回路は、3つの量子レジスタa, b, c, に対する操作からなる。ここで、第1レジスタ$|a\rangle_1$はn個の量子ビットのテンソル積$|a_{n-1}\rangle_1 \otimes |a_{n-2}\rangle_1 \otimes \cdots \otimes |a_0\rangle_1$からなり、整数値$a = 2^{n-1}a_{n-1} + 2^{n-2}a_{n-2} + \cdots + 2^0 a_0$を表す。同様に、第2レジスタ$|b\rangle_2$は$m$個の量子ビットのテンソル積$|b_{m-1}\rangle_2 \otimes |b_{m-2}\rangle_2 \otimes \cdots \otimes |b_0\rangle_2$であり、整数値$b = 2^{m-1}b_{m-1} + 2^{m-2}b_{m-2} + \cdots + 2^0 b_0$を表す。第3レジスタ$|c\rangle_3$は$l$個の量子ビットのテンソル積$|c_{l-1}\rangle_3 \otimes |c_{l-2}\rangle_3 \otimes \cdots \otimes |c_0\rangle_3$であり、整数値$c = 2^{l-1}c_{l-1} + 2^{l-2}c_{l-2} + \cdots + 2^0 c_0$を表す。これらの量子レジスタが、量子乗算回路により次式のように状態が変化することが期待される。

$$|a\rangle_1|b\rangle_2|c\rangle_3 \rightarrow |a\rangle_1|b\rangle_2|c + ab \bmod 2^l\rangle_3 \tag{2}$$

ここで、右辺の第3レジスタの剰余(mod)は$c+ab$が2^lを超えるとオーバーフローすることを表している。通常、$c=0$かつ$l \geq m+n$であればオーバーフローは生じない。

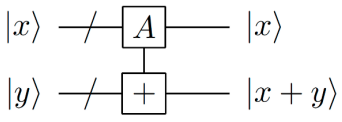
2.1.1 古典的な量子乗算回路

古典的な乗算アルゴリズムとしては、一般的な筆算のように、桁をシフトして加算を繰り返す手法が知られている[14)]。この考え方をベースに、量子乗算回路は量子加算回路を繰り返すことで実装できる。**図－1** は量子加算回路を示したもので、第1レジスタ$|x\rangle$に回路Aが、第2レジスタ$|y\rangle$にA

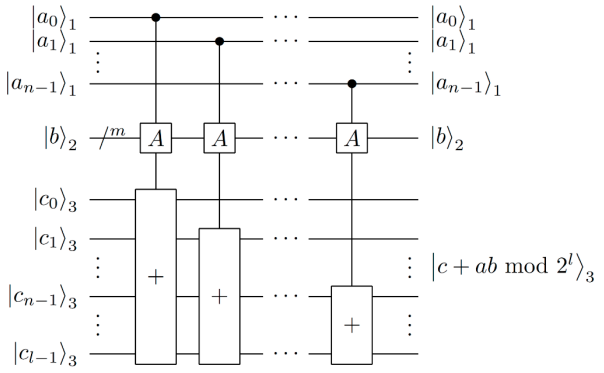
と制御関係にある回路＋が作用し、第 2 レジスタの状態を$|x+y\rangle$に変化させている。この量子加算回路を使うと、量子乗算回路は、**図－2** に示すように、$s=0,1,\ldots,n-1$である量子ビット$|a_s\rangle_1$を制御量子ビットとした制御量子加算回路を作用させ、第 2 レジスタの値を第 3 レジスタの上位$l-s$ビットに加算していくことで実装できる。一連の制御量子加算回路により、第 3 レジスタは次式のように変化する。

$$
\begin{aligned}
|c\rangle_3 &\rightarrow |c+2^0a_0b \bmod 2^l\rangle_3 \\
&\rightarrow |c+2^1a_1b+2^0a_0b \bmod 2^l\rangle_3 \\
&\vdots \\
&\rightarrow |c+2^{n-1}a_{n-1}b+\cdots+2^1a_1b+2^0a_0b \bmod 2^l\rangle_3 \\
&= |c+ab \bmod 2^l\rangle_3
\end{aligned}
\tag{3}
$$

この量子乗算回路はn個の制御量子加算回路から構成され、使用する量子加算回路に応じた補助量子ビットが必要である。一例として、2 つのn量子ビットの乗算において、補助量子ビットが不要かつゲート数が$O(n)$である量子加算回路を使用すれば、量子乗算回路もまた補助量子ビットなしで、ゲート数は$O(n^2)$で実装することが可能である。



図－1　量子加算回路



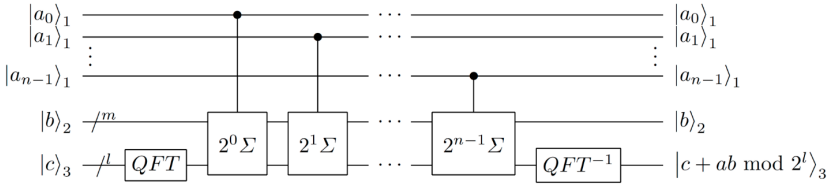
図－2　量子加算回路を使った量子乗算回路

2.1.2　QFT 乗算回路

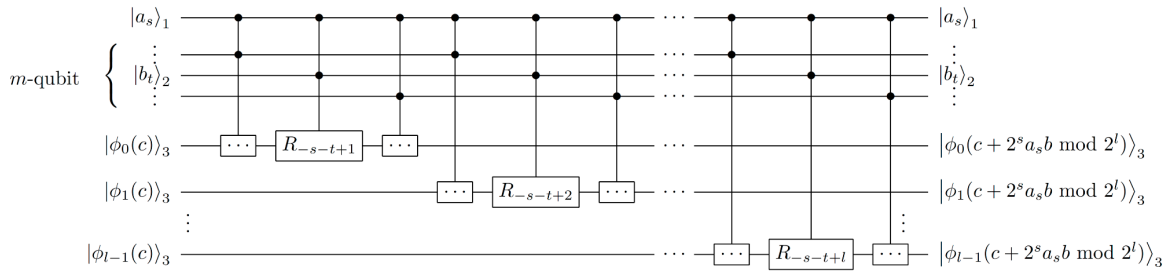
図－3 に示す QFT を使用した量子乗算回路は、$O(n^3)$のゲート数で構成可能であり、補助量子ビットを必要としない[18]。はじめに、第 3 レジスタにQFT回路を作用させ、状態を$|c\rangle_3 \rightarrow |\varphi(c)\rangle_3$に変化させる。ここで、QFT 後の第 3 レジスタのu番目$(u=0,1,\ldots,l-1)$の量子ビットの状態は、次式のように表すことができる。

$$
|c_u\rangle_3 \rightarrow \frac{1}{\sqrt{2}}\left(|0\rangle_3 + e^{2\pi i 0.c_u c_{u-1}\cdots c_0}|1\rangle_3\right) = |\varphi_u(c)\rangle_3
\tag{4}
$$

その後、第 3 レジスタの位相を、**図－3、4** に



図－3　QFT を使った量子乗算回路



図－4 制御$2^s\Sigma$回路

示す制御$2^s\Sigma$回路により回転させる。この回路は第1レジスタs番目の量子ビット$|a_s\rangle_1$と第2レジスタt番目の量子ビット$|b_t\rangle_2$を制御ビットとする制御・制御R_jゲートで構成され、第3レジスタu番目の量子ビットに対し、$j=-s-t+u+1$をパラメータとした次式のユニタリ演算により回転させるものである。

$$R_j=\begin{pmatrix}1 & 0\\ 0 & e^{\frac{2\pi i}{2^j}}\end{pmatrix} \tag{5}$$

なお、$j\le 0$ではR_jゲートは恒等ゲート(I ゲート)として作用する。

ここで、$|a_s\rangle_1$による制御を省略し、$s=0$としたΣ回路による操作を考えると、これは第2レジスタと第3レジスタによるQFT加算回路における位相制御をしており[10]、第3レジスタの状態は$|\varphi(c)\rangle_3\to|\varphi(c+b \bmod 2^l)\rangle_3$に変化する。これに対し、$|a_s\rangle_1$を制御ビットとした制御$2^s\Sigma$回路は、第3レジスタの位相に$2^s a_s b$が加えられるため、第3レジスタは$|\varphi(c)\rangle_3\to|\varphi(c+2^s a_s b \bmod 2^l)\rangle_3$に変化する。このため、制御$2^s\Sigma$回路を$s=0,1,\dots,n-1$と繰り返し作用させると、第3レジスタは次式のように変化する。

$$\begin{aligned}|\varphi(c)\rangle_3 &\to |\varphi(c+2^0a_0b \bmod 2^l)\rangle_3\\ &\to |\varphi(c+2^1a_1b+2^0a_0b \bmod 2^l)\rangle_3\\ &\vdots\\ &\to |\varphi(c+2^{n-1}a_{n-1}b+\cdots+2^0a_0b \bmod 2^l)\rangle_3\\ &= |\varphi(c+ab \bmod 2^l)\rangle_3\end{aligned} \tag{6}$$

最終的に、第3レジスタに逆QFT(QFT^{-1})を行うことで、$|c+ab \bmod 2^l\rangle_3$が得られ、第3レジスタには第1と第2レジスタの積が加算される。

2.2 量子二乗回路

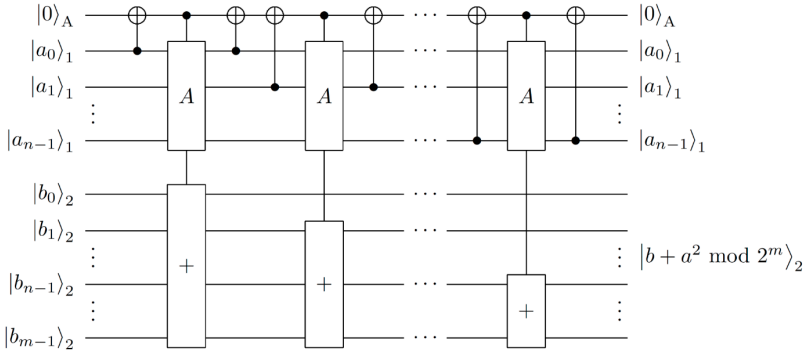
量子二乗回路は、重ね合わせ状態にある整数値を二乗する量子回路である。第1レジスタにn量子ビットからなる$|a\rangle_1$、第2レジスタにm量子ビットからなる$|b\rangle_2$を入力し、量子二乗回路により状態は次式のように変化することが期待される。

$$|a\rangle_1|b\rangle_2\to|a\rangle_1|b+a^2 \bmod 2^m\rangle_2 \tag{7}$$

ここで、右辺の第2レジスタの剰余は$b+a^2$が2^mを超えるとオーバーフローすることを表している。通常、$b=0$かつ$m\ge 2n$であればオーバーフローは生じない。以下では、2.1節で示した量子乗算回路をベースに、入力する量子レジスタを省略して量子二乗回路を実装する方法を記す。

2.2.1 古典的量子二乗回路

2.1.1項で示した古典的な量子乗算回路をベースにした量子二乗回路は、**図－5**に示すように制御量子加算回路と状態$|0\rangle$で初期化された1つの補助量子ビット$|0\rangle_A$を使い実装することが可能である。第1レジスタs番目の量子ビットの状態$|a_s\rangle_1$をCNOTゲートにより補助量子ビットへ複製した後に、補助量子ビットを制御ビットとした制御加算回路により、第2レジスタの上位$m-s$桁の状態を変化させることで、第2レジスタに$2^s a_s a$を加算することが出来る。したがって、一連の制御量子加算回路を適用させることにより、第2レジスタの



図－5 量子加算回路を使った量子二乗回路

状態は$|b+a^2 \bmod 2^m\rangle_2$に変化する。

この量子二乗回路はn個の量子加算回路が必要であり、量子加算回路が必要とする補助量子ビット数をN_Aとすると、$1+N_A$個の補助量子ビットを必要とする。回路に必要なゲート数のオーダーについても、2.1.1 項で示した 2 つのn量子ビットの乗算に必要な量子乗算回路と同様に、ゲート数が$O(n)$である量子加算回路を使用すれば、量子二乗回路はゲート数は$O(n^2)$で実装することが可能である。

2.2.2 QFT 二乗回路

2.2.1 項で示した QFT を使用した乗算回路をベースにした量子二乗回路を**図－6** に示す。はじめに、第 2 レジスタ$|b\rangle_2$を QFT により$|\varphi(b)\rangle_2$に変化させる。ここで、第 2 レジスタu番目$(u=0,1,\dots,m-1)$の量子ビットの状態は次式のように変化する。

$$|b_u\rangle_2 \to \frac{1}{\sqrt{2}}\left(|0\rangle_2 + e^{2\pi i 0.b_u b_{u-1}\dots b_0}|1\rangle_2\right) = |\varphi_u(b)\rangle_2 \tag{8}$$

ここで、$s=0,1,\dots,n-1$である一連の$2^s\Sigma_s$回路を$|a\rangle_1$と$|\varphi(b)\rangle_2$に作用させ、各回路で第 2 レジスタの位相に$2^s a_s a$の情報を加えていく。**図－7** に示すように、$2^s\Sigma_s$回路では第 2 レジスタのu番目の量子ビットに対し、$|a_s\rangle_1$ と $|a_t\rangle_1$ $(t=0,1,\dots,n-1, t\neq s)$を制御ビットとする制御・制御$R_j$ゲート及び$|a_s\rangle_1$を制御ビットとする制御$R_{j'}$ゲート$(j'=-2s+u+1)$で構成される。QFT 後の第 2 レジスタは$2^s\Sigma_s$回路を作用させることで$|\varphi(b)\rangle_2 \to |\varphi(b+2^s a_s a \bmod 2^m\rangle_2$に変化し、一連の$2^s\Sigma_s$回路によって第 2 レジスタは$|\varphi(b+a^2 \bmod 2^m)\rangle_2$に変化する。最終的に逆 QFT を行うことで、第 2 レジスタの状態は第 1 レジスタを 2 乗した値を格納した$|b+a^2 \bmod 2^m\rangle_2$になる。

2.2.1 項に示した量子二乗回路と比較すると、この QFT 二乗回路は補助量子ビットを必要としない点では有利であるが、ゲートサイズは 2 つの n ビット整数に対する QFT 乗算回路と同じ$O(n^3)$であるため、回路規模は大きくなる。

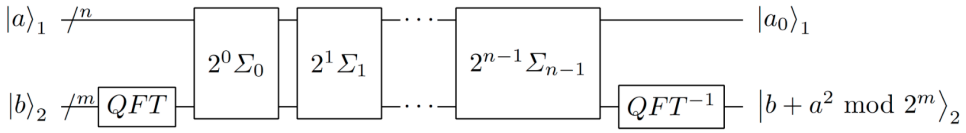
3. 円周率推定量子回路

3.1 モンテカルロ計算

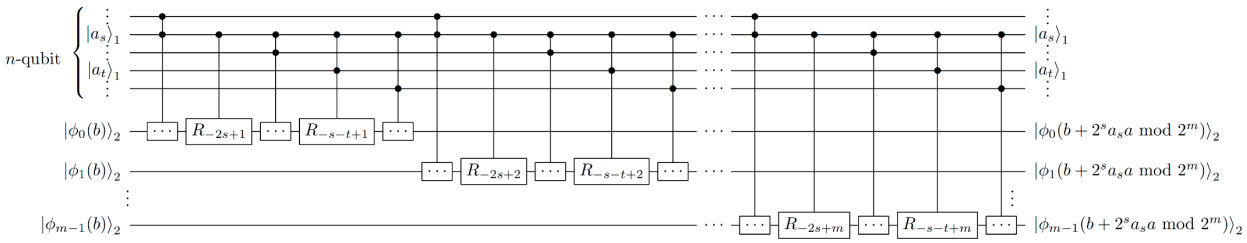
古典的なモンテカルロ計算では、乱数を用いてランダムなサンプリングを繰り返し、期待値を得ることで問題を解く方法である。解の精度はサンプル数を増やすことで向上し、誤差がϵである結果を得るために必要なサンプル数は$O(1/\epsilon^2)$である。

量子コンピュータでは、量子振幅推定を使用して期待値を評価するアルゴリズムが提案されている[2)]。このアルゴリズムの原理を以下に述べる。

nビットの2進数である$x\in\{0,1\}^n$に対し、$f(x)$を古典的な実数値関数、$p(x)$を確率分布関数とする



図－6　QFT を使用した量子二乗回路



図－7　$2^s\Sigma_s$回路

と、$f(x)$の期待値$\mathbb{E}[f(x)]$は次式のように表すことができる。

$$\mathbb{E}[f(x)] = \sum_{x=0}^{2^{n-1}} f(x)\, p(x) \tag{9}$$

この式の$f(x)p(x)$はサンプル数(2^n回)だけ行われることから、古典計算誤差がϵであるために必要な計算量もまた$O(1/\epsilon^2)$である。

一方で、量子アルゴリズムではn量子ビットからなる第 1 レジスタ$|x\rangle_1$と 1 つ量子ビットからなる第 2 レジスタ$|0\rangle_2$に作用し、第 2 レジスタを次式のように回転させるユニタリ演算子$\mathcal{R}$を用意する。

$$\mathcal{R}|x\rangle_1|0\rangle_2 = \sqrt{f(x)}|x\rangle_1|1\rangle_2 + \sqrt{1-f(x)}|x\rangle_1|0\rangle_2 \tag{10}$$

さらに、$|0\rangle$で初期化された第 1 レジスタを、次式で示す重ね合わせ状態に変化させるユニタリ演算子$\mathcal{P}$を用意する。

$$\mathcal{P}|0\rangle_1 = \sum_{x=0}^{2^n-1} \sqrt{p(x)}|x\rangle_1 \tag{11}$$

これら$\mathcal{P}$と$\mathcal{R}$、さらに第 2 レジスタに作用する恒等演算子I_2を用い、初期化された量子レジスタに作用させると、式(9)で示した$\mathbb{E}[f(x)]$の平方根が次式のように第 2 レジスタが$|1\rangle_2$である時の量子振幅に現れ、$|1\rangle_2$の測定確率が$f(x)$の期待値になる。

$$\begin{aligned} &R(P \otimes I_2)|0\rangle_1|0\rangle_2 \\ &= \sum_{x=0}^{2^n-1} \sqrt{f(x)}\sqrt{p(x)}\,|x\rangle_1|1\rangle_2 \\ &\quad + \sum_{x=0}^{2^n-1} \sqrt{1-f(x)}\sqrt{p(x)}\,|x\rangle_1|0\rangle_2 \end{aligned} \tag{12}$$

したがって、第 2 レジスタが$|1\rangle_2$である状態の量子振幅を推定することで$\mathbb{E}[f(x)]$を得ることが可能である。式(11)の左辺の演算子を$\mathcal{A} = \mathcal{R}(\mathcal{P} \otimes \mathrm{I}_2)$とすると、量子振幅推定回路には$\mathcal{A}$と$\mathcal{A}^{-1}$が$O(1/\epsilon)$だけ必要であることがわかっているため、これを古典的なモンテカルロ計算に比べると、計算量は平方根のオーダーで減らすことが可能である[6]。

3.2 量子回路

有名なモンテカルロ計算による円周率推定法と同様に、正方形とその 2 辺を共有する四半円の面積比から円周率を求める。正方形と四半円が共有している 2 辺がそれぞれx軸およびy軸上にあり、平面上の(x,y)座標をそれぞれn個の量子ビットで表す時、xとyはそれぞれ $0 \le x \le 2^n - 1,\ 0 \le y \le 2^n - 1$をとることができるため、1 辺の長さが$2^n$である正方形の内部の$2^{2n}$個の座標を表すことができる。**図－8** はサンプリングの様子を示したものであり、サンプルの座標を表す黒点が等間隔に並んでいる。さらに、次式のように、任意の(x,y)が半径2^nの四半円の内部にある場合 1 を返し、外部にある場合は 0 を返す関数$f(x,y)$が必要である。

$$f(x,y) = \begin{cases} 1 & \text{if } x^2 + y^2 < 2^{2n} \\ 0 & \text{if } x^2 + y^2 \ge 2^{2n} \end{cases} \tag{13}$$

サンプリングの確率分布を$p(x,y)$とすると、次式から$\pi/4$を推定することができる。

$$\frac{\pi}{4} \approx \sum_{y=0}^{2^n-1} \sum_{x=0}^{2^n-1} f(x,y)p(x,y) \tag{14}$$

量子回路では、3 つの量子レジスタを操作することでπを求める。第 1 レジスタはx座標を表すn量子ビット、第 2 レジスタは同様にy座標を表すn量子ビット、第 3 レジスタは量子振幅推定のために使用する 1 量子ビットからなる。また、この回路における$\mathcal{P}$は第 1 と第 2 レジスタに、$\mathcal{R}$は第 1～3 レジスタに作用し、I_3は第 3 レジスタに対する恒等演算子とすると、本回路が行う演算は次式となる。

$$
\begin{aligned}
&\mathcal{R}(\mathcal{P}\otimes I_3)|0\rangle_1|0\rangle_2|0\rangle_3\\
&=\mathcal{R}\sum_{y=0}^{2^n-1}\sum_{x=0}^{2^n-1}\sqrt{p(x,y)}\,|x\rangle_1|y\rangle_2|0\rangle_3\\
&=\sum_{y=0}^{2^n-1}\sum_{x=0}^{2^n-1}\sqrt{f(x,y)}\sqrt{p(x,y)}\,|x\rangle_1|y\rangle_2|1\rangle_3\\
&\quad+\sum_{y=0}^{2^n-1}\sum_{x=0}^{2^n-1}\sqrt{1-f(x,y)}\sqrt{p(x,y)}\,|x\rangle_1|y\rangle_2|0\rangle_3
\end{aligned}
\tag{15}
$$

したがって、式(15)による操作の後に、第 3 レジスタが$|1\rangle_3$であるときの量子振幅を推定することで、$\pi/4$の期待値が得られる。

$\mathcal{P}$はxとyを均等な確率分布を持った重ね合わせ状態にするものであるため、第 1 と第 2 レジスタの各量子ビットにアダマールゲート(H ゲート)を作用させることで実装できる。

$\mathcal{R}$は第 1 と第 2 レジスタの状態に応じて、第 3 レジスタを回転させ、$|1\rangle_3$の振幅を$\sqrt{f(x,y)}$に変化させる。ここで、式(12)より$f(x,y)$は 0 と 1 しか取らないため、次式が成り立つ。

$$
\sqrt{f(x,y)}|1\rangle_3+\sqrt{1-f(x,y)}|0\rangle_3=|f(x,y)\rangle_3 \tag{16}
$$

$\mathcal{R}$を実装するには、$2n$個の量子ビットが$|0\rangle$で初期化された補助量子レジスタ$|0\rangle_A$を加えた上で、**図－9** および次式に示す操作を行う。

$$
\begin{aligned}
&|x\rangle_1|y\rangle_2|0\rangle_3|0\rangle_A\\
&\to|x\rangle_1|y\rangle_2|0\rangle_{3A} && \text{(a)}\\
&\to|x\rangle_1|y\rangle_2|x^2+y^2 \bmod 2^{2n+1}\rangle_{3A} && \text{(b)}\\
&\to|x\rangle_1|y\rangle_2\left|\overline{f(x,y)}\right\rangle_3|x^2+y^2 \bmod 2^{2n}\rangle_A && \text{(c)}\\
&\to|x\rangle_1|y\rangle_2|f(x,y)\rangle_3|x^2+y^2 \bmod 2^{2n}\rangle_A && (d)\\
&\to|x\rangle_1|y\rangle_2|f(x,y)\rangle_3|0\rangle_A && \text{(e)}
\end{aligned}
\tag{17}
$$

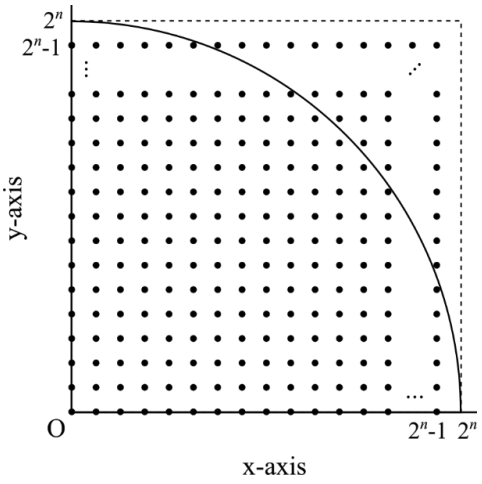
(a) 補助量子レジスタと第 3 レジスタを合わせ、第 3 レジスタを最上位ビットとした 3A レジスタとして扱う。

(b) 量子二乗回路(SQ)によりxとyをそれぞれ 2 乗した値を 3A レジスタに加える。

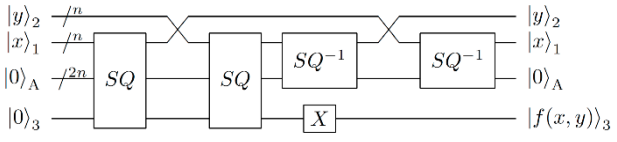
(c) 3A レジスタをもとの 1 量子ビットからなる第 3 レジスタと、$2n$量子ビットからなる補助量子レジスタに分割すると、第 3 レジスタは$x^2+y^2>2^{2n}$であるときだけ$|1\rangle_3$になるため、この 2 つの量子レジスタの状態は$\left|\overline{f(x,y)}\right\rangle_3|x^2+y^2 \bmod 2^{2n}\rangle_A$で表すことができる。

(d) 第 3 レジスタに NOT ゲート(X ゲート)を作用させることで、$|f(x,y)\rangle_3$が得られる。

(e) 量子二乗回路はユニタリ演算であるため、補助量子ビットに対してxとyの二乗の逆演算(SQ^{-1})を行うことで、状態は$|0\rangle_A$に戻る。



図－8　円周率推定のためのサンプリングの様子



図－9　円周率推定の$\mathcal{R}$回路

3.3　シミュレーション

図－9 に示した回路と量子振幅推定回路を Qiskit 0.19.2[19,20]により実装し、シミュレータにより円周率推定を行った。量子ビット数の増加を避けるため、回路$\mathcal{R}$中の量子二乗回路は 2.2.2 項に示した QFT 量子二乗回路を、量子振幅推定には最尤推定法による量子振幅推定法 [2]を使用することで、$4n+1$個の量子ビットを使って2^{2n}個のサンプリングを行う量子回路を実装することが可能となった。量子振幅推定の計算量は、$\mathcal{A}=\mathcal{R}(\mathcal{P}\otimes \mathrm{I}_3)$

と $\mathcal{A}^{-1}$の実行回数とみなすことが可能であり、最尤推定法を使った場合、量子振幅推定の推定誤差にϵ_A対し$O((1/\epsilon_A \ln(1/\epsilon_A))$になることが報告されている[2)]。この手法では$\mathcal{A}$と$\mathcal{A}^{-1}$が含まれる振幅増幅回路の長さを変え、尤度関数を作成して推定値を求める。k番目の反復測定においては、振幅増幅がm_k回含まれる回路により測定される。シミュレーションでは$m_0 = 0,\ m_k = 2^{k-1}$として、$k = 0,1$ (kの最大値：$k_{max} = 1$)と$k = 0,1,\ldots 5$ ($k_{max} = 5$)の2パターンで行った。計算量を$\mathcal{A}$及び$\mathcal{A}^{-1}$が呼び出された数とすると$k_{max} = 1$の時の計算量は400であり、推定精度は$O(10^{-2})$、$k_{max} = 5$の時の計算量は6,800であり、推定精度は$O(10^{-3})$である。

シミュレーションではnを2から6まで変化させ、それぞれのnに対して100回ずつ円周率の推定を行い、平均値と分散を評価した。加えて、量子回路と同じサンプリングを古典計算で実施した。

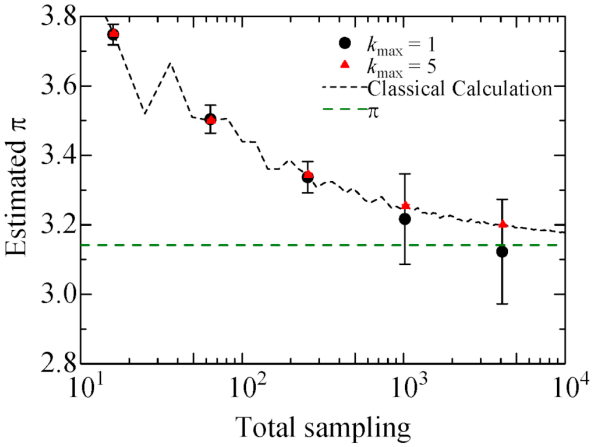
3.4 結果と考察

量子シミュレーションで得られた推定値の平均値及び標準分散、及び古典計算による結果を**図－10**に示す。量子シミュレーションの結果はいずれも誤差内で古典計算に一致した。特に、計算量を増やすと古典計算の結果によく一致するが、サンプル数が少なければ真値からは外れた値になっている。古典的なモンテカルロ計算では計算量とサンプル数は一致するため、計算量を増やせば真値に近づけることが可能であるが、量子回路ではサンプル数は主に量子ビット数に、計算量は回路長に依存する。したがって、量子回路による最終的な誤差は、サンプル数による誤差ϵ_Sと量子振幅推定による誤差ϵ_Aを合わせた$\epsilon_S + \epsilon_A$になると考えられ、どちらか一方だけ誤差を減らしても結果が真値に近づかない。

古典的なモンテカルロ計算では乱数を使ったサンプリングを行うのに対し、今回実装した量子アルゴリズムでは、多次元数値積分と同様に各次元で等間隔なサンプリングを行い、その重ね合わせ状態を計算に用いている。多次元数値積分におけるサンプリングと同様に考えると、dを問題の次元とした時に誤差ϵ_Sを得るために必要なサンプル数は$O(1/{\epsilon_S}^d)$である。言い換えると、サンプル数Nに対して$\epsilon_S = O\left(1/\sqrt[d]{N}\right)$が成り立ち、今回の円周率推定においては$d = 2$、$N = 2^{2n}$より$\epsilon_S = O(1/2^n)$である。したがって、$\epsilon_A = O(10^{-3})$の時、$n = 10$程度であれば誤差が$O(10^{-3})$である期待値が得られる。

従来の数値積分では問題の次元が増えると必要な計算量、すなわちサンプル数が指数関数的に増加するという「次元の呪い」があった。一方で量子アルゴリズムではサンプリングに使う量子ビット数を増やすとで、サンプル数だけ指数関数的に増加させることが可能であり、計算量は前述の通りモンテカルロ計算の平方根のオーダーとなる。なお、オリジナルの量子振幅推定を用いる場合、推定誤差は$O(1/\epsilon_A)$であることから[2)]、各次元には$O(\log_2(1/\epsilon_A))$の量子ビットが要求される。

一方、ϵ_Aを減らすには、計算量を増やす必要があるが、量子回路の長さは量子ビットが維持できる時間(コヒーレンス時間)や量子ゲートの操作時間に依存する。また、扱う量子ビットや量子ゲートが増えるにつれ、量子コンピュータは低いエラー率が要求される。したがって、量子コンピュータにより現在のモンテカルロ計算を高速化するには、量子コンピュータに誤り耐性が実装された上で、問題が要求する精度や次元に応じた要求性能を満たしている必要がある。



図－10 量子シミュレーション及び古典計算による円周率推定の結果

4. 結論

量子二乗回路を使った円周率推定回路を実装した上で、量子コンピュータを使ったモンテカルロ計算についての考察を行った。

量子二乗回路は既知の量子乗算回路をベースにして、古典的な加算回路のアルゴリズムを使ったものと QFT を使ったものの 2 つを検討した。回路長は前者の方が短く、必要な量子ビット数は後者の方が少ないので優れている。

量子振幅推定を用いた円周率推定回路は、$4n+1$個の量子ビットを使い、2^{2n}個のサンプリングによる円周率推定が可能であることを示した。また、推定値には量子ビット数で決まるサンプル数に依存する誤差ϵ_Sと量子振幅推定の計算量に依存する誤差ϵ_Aが含まれることも示した。高い精度で推定値を得るには、必要な精度に応じた量子ビット数と回路長を満たす量子コンピュータが必要である。

謝辞

本研究を進めるにあたり University College London の Sougato Bose 教授, 紅林秀和先生にご支援・ご指導いただきました。ここに謝意を表します。

＜参考文献＞

1) Brassard, G., Hoyer, P., Mosca, M., Tapp, A.: Quantum amplitude amplification and estimation. AMS Contemporary Mathematics. 305, pp.53–74, 2002
2) Suzuki, Y., Uno, S., Raymond, R., Tanaka, T., Onodera, T., Yamamoto, N.: Amplitude estimation without phase estimation. Quantum Inf. Process. 19, 75, 2020
3) Tanaka, T., Suzuki, Y., Uno, S., Raymond, R., Onodera, T., Yamamoto, N.: Amplitude estimation via maximum likelihood on noisy quantum computer. Quantum Inf. Process. 20, 293, 2021
4) Uno, S., Suzuki. Y., Hisanaga, K., Raymond, R., Tanaka, T., Onodera, T., Yamamoto, N.: Modified Grover operator for quantum amplitude estimation. New J. Phys. 23, 083031, 2021
5) Montanaro, A.: Quantum speedup of Monte Carlo methods. P. Roy. Soc. A-Math Phy. 471, 20150301, 2015
6) Bochkin, G.A., Doronin, S.I., Fel'dman, E.B., Zenchuk, A.I.: Calculation of π on the IBM quantum computer and the accuracy of one-qubit operations. Quantum Inf Process. 19, 257, 2020
1. Asfaw A., Bello L., Ben-Haim Y., Bravyi S., Capelluto L., Vazquez A.C., Ceroni J., Chen R., Frisch A., Gambetta J., Garion S., Gil L., Gonzalez S.D.L.P., Harkins F., Imamichi T., McKay D., Mezzacapo A., Minev Z., Movassagh R., Nannicni G., Nation P., Phan A., Pistoia M., Rattew A. Schaefer J., Shabani J., Smolin J., Temme K., Tod M., Wood S., Wootton J.: Estimating Pi Using Quantum Phase Estimation Algorithm. Learn Quantum Computation using Qiskit. https://qiskit.org/textbook/ch-demos/piday-code.html , 2020 (Accessed 18 September 2020)
7) Vedral, V., Barenco, A., Ekert, A.: Quantum networks for elementary arithmetic operations. Phys. Rev. A. 54, pp.147–153, 1996
8) Takahashi, Y., Tani, S., Kunihiro, N.: Quantum addition circuits and unbounded fan-out. Quantum Info. Comput. 10, pp.872–890, 2010
9) Draper, T.G.: Addition on a quantum computer. arXiv:quant-ph/0008033, 2000
10) Álvarez-Sánchez, J.J., Álvarez-Bravo, J.V., Nieto, L.M.: A quantum architecture for multiplying signed integers. J. Phys.: Conf. Ser. 128, 012013, 2008
11) Kotiyal, S., Thapliyal, H., Ranganathan, N.: Circuit for reversible quantum multiplier based on binary tree optimizing ancilla and garbage bits. In: 2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems. pp. 545–550, 2014
12) Babu, H.Md.H.: Cost-efficient design of a quantum multiplier–accumulator unit. Quantum Inf. Process. 16, 30, 2016
13) Parent, A., Roetteler, M., Mosca, M.: Improved reversible and quantum circuits for Karatsuba-based integer multiplication. arXiv:1706.03419 [quant-ph], 2017
14) Dutta, S., Bhattacharjee, D., Chattopadhyay, A.: Quantum circuits for Toom-Cook multiplication. Phys. Rev. A. 98, 012311, 2018
15) Maynard, C.M., Pius, E.: A quantum multiply-accumulator. Quantum Inf. Process. 13, pp.1127–1138, 2014

16) Pavlidis, A., Gizopoulos, D.: Fast quantum modular exponentiation architecture for Shor's factoring algorithm. Quantum Info. Comput. 14, pp.649–682, 2014

17) Ruiz-Perez, L., Garcia-Escartin, J.C.: Quantum arithmetic with the quantum Fourier transform. Quantum Inf. Process. 16, 152, 2017

18) Aleksandrowicz, G., Alexander, T., Barkoutsos, P., Bello, L., Ben-Haim, Y., Bucher, D., Cabrera-Hernádez, F.J., Carballo-Franquis, J., Chen, A., Chen, C.F., Chow, J.M., Córcoles-Gonzales, A.D., Cross, A.J., Cross, A., Cruz-Benito, J., Culver, C., González, S.D.L.P., Torre, E.D.L., Ding, D., Dumitrescu, E., Duran, I., Eendebak, P., Everitt, M., Sertage, I.F., Frisch, A., Fuhrer, A., Gambetta, J., Gago, B.G., Gomez-Mosquera, J., Greenberg, D., Hamamura, I., Havlicek, V., Hellmers, J., Herok, Ł., Horii, H., Hu, S., Imamichi, T., Itoko, T., Javadi-Abhari, A., Kanazawa, N., Karazeev, A., Krsulich, K., Liu, P., Luh, Y.,Maeng, Y.,Marques, M.,Martín-Fernández, F.J.,McClure, D.T., McKay, D., Meesala, S., Mezzacapo, A., Moll, N., Rodríguez, D.M., Nannicini, G., Nation, P., Ollitrault, P., O'Riordan, L.J., Paik, H., Pérez, J., Phan, A., Pistoia, M., Prutyanov, V., Reuter,M., Rice, J., Davila, A.R., Rudy, R.H.P., Ryu, M., Sathaye, N., Schnabel, C., Schoute, E., Setia, K., Shi, Y., Silva, A., Siraichi, Y., Sivarajah, S., Smolin, J.A., Soeken, M., Takahashi, H., Tavernelli, I., Taylor, C., Taylour, P., Trabing, K., Treinish, M., Turner,W., Vogt-Lee, D., Vuillot, C.,Wildstrom, J.A.,Wilson, J.,Winston, E.,Wood, C.,Wood, S., Wörner, S., Akhalwaya, I.Y., Zoufal, C.:Qiskit: An open-source framework for quantum computing., 2019 https://doi.org/10.5281/zenodo.2562110

19) Gambetta, J., Treinish, M., Kassebaum, P., Nation, P., Rodríguez, D.M., González, S.D.L.P., Hu, S., Krsulich, K., Zdanski, L., qiskit-bot, Yu, J., Travis-S-IBM, Gomez, J., McKay, D., Gacon, J., Naveh, Y., Wood, S., lerongil, Capelluto, L., Ishizaki, K., abbycross, tigerjack, Garion, S., Dague, S., RohitMidha23, Marques, M., GEORGE, M., Schwarm, J., AlbinoZenci, Cruz, A.: Qiskit/qiskit: Qiskit 0.19.2., 2020 https://doi.org/10.5281/zenodo.3829023