量子計算

西村治道（大阪府立大学理学系研究科）

1　はじめに

量子計算は現在のコンピュータで行える計算ではない（本特集の渡辺氏の解説でいう「一部の例外」である）．量子計算は量子コンピュータでしか行えない．では，量子計算や量子コンピュータとは何なのか？それを解説する前に量子コンピュータで効率的に解ける最も有名な問題－整数の因数分解問題－を紹介する．

入力例. 自然数 N

答え. N の非自明な因数（N が素数なら「素数である」と判定）

非常に単純な問題である．しかし計算量は別である．この問題を解く我々が知る簡単なアルゴリズムは「2 から $\sqrt{N}$ までの自然数で N を割ってみる」ことである．N を割り切る数 p が見つかれば p を出力すればよく，見つからなければ「素数」と判定すればよい．しかしこのアルゴリズムは $\Omega(\sqrt{N})$ の計算量を必要とする．整数の因数分解問題の入力サイズは $\log N$ なので指数時間のアルゴリズムとなってしまう．N が大きくなると到底現実的な時間で計算を終了できるものではない．ではもっと効率の良いアルゴリズムはあるのか？確かに上記の簡単なアルゴリズムよりよい方法は知られている．しかしそれでもアルゴリズムの計算量は入力サイズの多項式では押さえられず効率的とはいえない．実はこの問題，現在のコンピュータにおける効率的なアルゴリズムが存在しないと予想されている．さらに重要なことに，インターネット上で情報をやり取りする主要な暗号系（RSA 暗号など）の安全性はこの問題が効率的に解けないというのが大前提なのである（詳しくは田中氏の解説参照のこと）．そんな問題が量子コンピュータを使えば効率的に解けてしまうとわかったのである．この事実は 1994 年に MIT の計算機科学の研究者である Shor によって発見され，彼のアルゴリズムは Shor のアルゴリズムと呼ばれている．彼の発見は現代のインターネット社会の基盤をひっくり返すようなインパクトを人々に与えた．「量子コンピュータを作ろう」という実験の研究，「量子コンピュータで他に何ができるのか」などの理論の研究に取り組む人は飛躍的に増えた．今では量子計算は（著者の贔屓目の可能性もあるが）計算量の世界でも一定の居場所を確保しているように感じる．

2　量子計算の概念誕生まで

量子計算を説明するにあたって，量子計算の概念が誕生するまでの歴史を眺めるのは「なぜ量子計算？」かを知る上で悪くないだろう．大雑把に言えば今日のコンピュータは，ハードの面では基本素子を微小化することで高速化してきた．そしてその基本素子を司る物理法則は，我々人間の直感にマッチする「古典力学」である．（そんなわけで現在のコンピュータによる普通の計算は量子計算屋の間では「古典計算」と呼ばれる．）しかしながら原子レベルまで基本素子が微小化すると

基本素子は「量子力学」というミクロレベルの物理系が従うとされる力学の影響を無視できなくなってくる．そのようなわけで Benioff は量子力学の原理に基づいて現在のコンピュータと同じ計算を行えることを示した．一方，同時期に Feymann はもっとポジティブに量子力学を用いる可能性を指摘した．量子力学的現象の量子力学に基づいたコンピュータによるシミュレーションの可能性である．このようなことを考えた理由は，量子力学的現象は量子力学の「重ね合わせの原理」により，従来の古典力学に基づいたコンピュータで模倣しようとすると一般に指数的な時間がかかるとされているからである．ここまでの話は今日の「量子計算」と呼ばれる概念ではないが，「量子計算」を芽生えさせるきっかけとなったことは疑いのないものであろう．

「量子計算」の生みの親は，Oxford 大学の理論物理学者 Deutsch である．今から四半世紀前の 1985 年，彼は「重ね合わせの原理」が量子力学に基づいたコンピュータを考えたときにある種の「並列計算」を実現すると考えたのである．通常の並列計算とは，コンピュータを何台も並列に走らせてそれらで共同作業をさせることを指す．一方，「量子並列計算」ではたった 1 台の量子コンピュータがその量子力学的性質から並列的に作業をこなすのである．これこそまさに高速化の源！Deutsch は量子コンピュータの数学的モデルを提案し，量子コンピュータによる新しい計算の可能性を示唆したのである．

3　量子計算－普通の計算との違い

では量子計算を数学的に紹介しよう．まず重要なのは「何が情報の基本単位なのか？」である．通常のコンピュータでは情報の基本単位は「ビット」である．つまり取りうる状態は 0 か 1 かである．量子コンピュータでは基本単位からして違う．基本単位は「量子ビット」とよばれるもので，数学的にはビット 0 に対応するベクトル $|0\rangle = {}^T(1,0)$（T は転置）とビット 1 に対応するベクトル $|1\rangle = {}^T(0,1)$ を正規直交基底とする 2 次元複素内積空間内の単位ベクトル $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$ （ゆえに $|\alpha|^2 + |\beta|^2 = 1$）で表される．ベクトルを $|0\rangle$ のように「ケット記号」とよばれる $|\ \rangle$ で囲んで表すのは物理の慣習であるが，$\mathbf{e}_0$ などと表すよりビットとの対応が分かりやすい表現なので本稿では積極的に利用する．$|\phi\rangle$ のような状態を $|0\rangle$ と $|1\rangle$ の重ね合わせ状態といい，このような状態が取れるこそ量子力学の「重ね合わせの原理」である．感覚的には $|\phi\rangle$ は「重み」α でビット 0，「重み」β でビット 1 が「共存」しているような状態である．

ところで我々が最終的に計算したい問題は（整数の因数分解問題のように）入力例も答えもやはりビットに基づく．そこで量子ビットから我々がビットの情報を読み出すには $|\phi\rangle$ に対して「測定」と呼ばれる物理的作用を施す．残念ながら量子ビットからは測定で 2 ビット以上の情報を読み出せるわけではない．$|\phi\rangle$ を測定すると量子力学の原理により，確率 $|\alpha^2|$ でビット 0 を，確率 $|\beta|^2$ でビット 1 を得られることになる．つまり量子ビット 1 個を測定するだけでは（偏りがあるかもしれない）コインを振って 1 ビットを得るようなもので多くのことは期待できない．

次に量子コンピュータで 2 ビットに対応する概念を説明する．そのため，ベクトルの掛け算といえる「テンソル積」の概念を導入しよう．一般に m 次元ベクトル $\mathbf{a} = {}^T(a_1, \ldots, a_m)$ と n 次元ベクトル $\mathbf{b} = {}^T(b_1, \ldots, b_n)$ のテンソル積とは，$\mathbf{a} \otimes \mathbf{b}$ で表される $m \times n$ 次元ベクトルで

$\mathbf{a} \otimes \mathbf{b} = (a_1b_1, a_1b_2, \ldots, a_1b_n, a_2b_1, \ldots, a_nb_m)$ となる．このとき $00, 01, 10, 11$ に対応するベクトルは $2\times2 = 4$ 次元ベクトル $|0\rangle\otimes|0\rangle = {}^T(1,0,0,0)$, $|0\rangle\otimes|1\rangle = {}^T(0,1,0,0)$, $|1\rangle\otimes|0\rangle = {}^T(0,0,1,0)$, $|1\rangle \otimes |1\rangle = {}^T(0,0,0,1)$ である．以下では $|0\rangle \otimes |0\rangle$ はテンソル記号を省略して $|0\rangle|0\rangle$ と書いたり，もっと簡略化して $|00\rangle$ と書いたりもする．このとき，２量子ビットはこれらを正規直交基底とする４次元複素内積空間内の単位ベクトル $\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$ となる．２つの独立な量子ビット $|\phi_1\rangle = \alpha|0\rangle + \beta|1\rangle$ と $|\phi_2\rangle = \gamma|0\rangle + \delta|1\rangle$ はテンソル積が持つ双線形性を用いて $|\phi_1\rangle|\phi_2\rangle = (\alpha|0\rangle + \beta|1\rangle)(\gamma|0\rangle + \delta|1\rangle) = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle$ と書き直せるので２量子ビットである．一方で，すべての２量子ビットが独立な２つの量子ビットに分解できるわけではない．例えば $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ は $(\alpha|0\rangle + \beta|1\rangle)(\gamma|0\rangle + \delta|1\rangle)$ の形には表すことができない．このような場合，２つの量子ビットは互いに相関があることを意味している．

２量子ビットの概念を一般に拡張すると，m ビットに対応する m 量子ビットは 2^m 次元単位ベクトル $|\psi\rangle = \sum_{x\in\{0,1\}^m} \alpha_x|x\rangle$ （すなわち，$\sum_{x\in\{0,1\}^m} |\alpha_x|^2 = 1$）であり，実に 2^m 種類のビットパターンすべてを重ね合わせていることになる．このように指数的な個数のビット列を量子状態として「共存」させられることは，量子計算特有の長所である．とはいえ，m 量子ビット $|\psi\rangle$ を測定してしまえば，確率 $|\alpha_x|^2$ で値 $x \in \{0,1\}^m$ を得ることになり，「共存」は終わりを迎える．量子計算特有の何かを成すならやはり「共存」している間，ということになる．

最後に扱うのはどのように計算を進めるかである．量子力学で認められた量子状態の操作は数学的にはユニタリ変換で表現される．それで量子計算は最初に必要な数の量子ビット，例えば m 量子ビット $|0^m\rangle$（0^m は 0 を m 個並べた m ビット列），を準備してそれに自分が選んだユニタリ変換 $U_1, U_2, \ldots, U_t$ を順々に掛けていくことで状態を変化させる．最後に最終状態 $|\psi_f\rangle$ を測定して計算結果を得るのである．ここで注意が必要なことは選んでよいユニタリ変換 U_i である．U_i を好きに選んでしまっては計算量の概念が意味を成さなくなる．普通の計算で言えば $f : \{0,1\}^m \to \{0,1\}^n$ を計算するのに関数 f 自身を１ステップとして認めてしまうようなものである．このような場合，普通の計算では $AND(x,y) = x \wedge y$ （AND ゲート）および $NOT(x) = \neg x$（NOT ゲート）（ただし，$x, y \in \{0,1\}$）という２種類の関数を組み合わせることで f を表現できることが知られている．同じように量子計算でも選んでよいユニタリ変換は何種類かの簡素なユニタリ変換とされる．例えば（理論的には）次の２種類のユニタリ変換があれば十分なことが知られている．(i) $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ で定義されるユニタリ変換 H （アダマールゲート）(ii) $T|x\rangle|y\rangle|z\rangle = |x\rangle|y\rangle|(x \wedge y) \oplus z\rangle$ で定義されるユニタリ変換 T（Toffoli ゲート）．ただし，$x, y, z \in \{0,1\}$ で $\oplus$ は排他的論理和である．(ii) は第３量子ビットのみが第１，２量子ビットを条件に変化するので第３量子ビットを標的量子ビットと呼ぶ．(ii) は $z = 0$ とすれば AND ゲート，$x = y = 1$ とすればと NOT ゲートの代わりに使えるので，量子計算は自然に普通の計算を含んでいることになる．

では具体例として量子計算（の手続き）$\mathcal{Q}_{ex}$ を与えよう．

量子計算 $\mathcal{Q}_{ex}$．(i) 初期状態 $|000\rangle$ を準備する．

(ii) 最初の 2 つの量子ビットそれぞれに H を施す．すると状態は

$$|000\rangle \to \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)|00\rangle \to \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)|0\rangle = \frac{1}{2}(|000\rangle+|010\rangle+|100\rangle+|110\rangle)$$

と変化する．最初の量子ビットに H を施すときは 2，3 番目の量子ビットは不変で，2 番目の量子ビットに H を施すときは最初と 3 番目の量子ビットが不変であることに注意せよ．

(iii) 第 3 量子ビットを標的量子ビットとして T を施す．すると状態は

$$|\psi_3\rangle = \frac{1}{2}(|000\rangle+|010\rangle+|100\rangle+|111\rangle)$$

に変化する．

(iv) 最後に最初の 2 つの量子ビットそれぞれに H を施す．この操作により $|000\rangle$ は $\frac{1}{2}(|000\rangle+|010\rangle+|100\rangle+|110\rangle)$，$|010\rangle$ は $\frac{1}{2}(|000\rangle-|010\rangle+|100\rangle-|110\rangle)$，$|100\rangle$ は $\frac{1}{2}(|000\rangle+|010\rangle-|100\rangle-|110\rangle)$，$|111\rangle$ は $\frac{1}{2}(|001\rangle-|011\rangle-|101\rangle+|111\rangle)$ に変化する．ユニタリ変換の線形性から $|\psi_3\rangle$ は

$$|\psi_f\rangle = \frac{1}{4}(3|000\rangle+|010\rangle+|100\rangle-|110\rangle+|001\rangle-|011\rangle-|101\rangle+|111\rangle)$$

となることが分かる．最終状態 $|\psi_f\rangle$ を観測すると 000 を確率 9/16 で，$010, 100, 110, 001, 011, 101, 111$ のそれぞれを確率 1/16 で得ることになる．$\mathcal{Q}_{ex}$ の計算量は H が 4 回，T が 1 回用いられたので計 5 回とカウントする．

$\mathcal{Q}_{ex}$ には量子計算の普通の計算との違いが幾つか現れている．その 1 つ目は T の使用回数である．実は T は本質的には普通のコンピュータでも実現できる．というのも T は 3 ビット (x, y, z) を受け取ったとき 3 ビット $(x, y, x \wedge y \oplus z)$ を出力する関数と考えられるからである．普通のコンピュータで $000, 010, 100, 110$ の 4 つの入力に対して T の出力を計算したいとしよう．その場合，対応する出力 $000, 010, 100, 111$ を得るには T を 4 回使用する必要がある．ところが量子計算 $\mathcal{Q}_{ex}$ では (ii) で $000, 010, 100, 110$ を量子重ね合わせで表現し，(iii) で T をたった 1 回使うことで対応する出力 $000, 010, 100, 111$ の量子重ね合わせを得ているのである．これこそが Deutsch のいう「量子並列計算」である．2 つ目の違いは (iv) で得られる効果にある．$|\psi_f\rangle$ を測定すると 000 だけが高い確率で得られる．一方，010 などは $|\psi_f\rangle$ をユニタリ変換の線形性を使って纏め上げるまでの式の途中では 000 同様に 3 箇所に出現しているのにマイナスの符号を持つものを含むためベクトル同士がキャンセルしあって最終的には低い確率でしか得られない．このような「負の干渉」と呼ばれる効果は量子計算ならではであり，欲しくない出力に「負の干渉」を引き起こすようにアルゴリズムを設計することで，欲しい値を高い確率で出力させることを可能にするのである．

4 量子計算が効率的に解く問題

量子計算が得意とする問題は，何らかの隠れた（主に代数的な）構造を抽出することである．例えば Simon が量子アルゴリズムを発見したのは次のような関数の「周期」発見問題（Simon の問題と呼ばれる）である．

入力例. 関数 $f : \{0,1\}^n \to \{0,1\}^{n-1}$. ただし，$f$ はある $s \neq 0^n$ に対して $f(x) = f(x \oplus s)$（$x \oplus s$ は x と s の各ビットの排他的論理和を取って出来上がった n ビット）をみたす「2対1」関数

答え. 「周期」$s = s_1 s_2 \cdots s_n$

この問題における計算量のコストは s を見つけるために関数 f を評価すべき回数とする．通常の計算では $f(x_0) = f(x_0 \oplus s)$ となるような x_0 を見つけるまで様々な x に対して $f(x)$ を評価する必要があり，その回数は $\Omega(2^n)$ にもなってしまう．しかしながら量子計算では f の評価回数は高々 $O(n)$ でよく，計算量を指数的に改良する．以下の f の評価回数が1回である量子アルゴリズム $\mathcal{A}_{sm}$ がサブルーチンとして使われる．

量子アルゴリズム $\mathcal{A}_{sm}$. (i) 初期状態 $|0^n\rangle|0^{n-1}\rangle$ を準備する．便宜上，最初の n 量子ビットを第1レジスタ，残りの $n-1$ 量子ビットを第2レジスタとする．

(ii) 第1レジスタの各量子ビットに H を施す．その結果，すべての $x \in \{0,1\}^n$ の量子重ね合わせ $\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle|0^{n-1}\rangle$ を得る．

(iii) 各 x に対して f を量子並列的に評価し，$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle|f(x)\rangle$ を得る．

(iv) 第1レジスタの各量子ビットに H を施して，結果を測定し，第1レジスタの値を出力とする．

量子アルゴリズム $\mathcal{A}_{sm}$ と量子計算 $\mathcal{Q}_{ex}$ の類似性に注目して欲しい．$\mathcal{A}_{sm}$ でも (iii) では量子並列的に関数を評価し，(iv) では「負の干渉」を使って欲しくない値が出現しないような変換を施しているのである．(iv) で得られる出力 $z = z_1 z_2 \cdots z_n \in \{0,1\}^n$ はその最終状態から内積 $\langle z, s\rangle := \sum_{i=1} z_i s_i$ が 0 であるという性質をみたすことが確認できる（詳細は文献 [4] など参照せよ）．この事実を使って $O(n)$ 回くらい $\mathcal{A}_{sm}$ を走らせれば，「1次独立なベクトル」$z^1, \ldots, z^{n-1}$ が得られる．後は y に関する方程式 $\langle z^1, y\rangle = 0, \ldots, \langle z^n, y\rangle = 0$ に対して，普通のコンピュータで掃き出し法のアルゴリズムを実行すれば隠れた $y = s$ を抽出することができる．

Simon のアルゴリズムは Shor のアルゴリズムの大いなるヒントとなった．というのも，Shor のアルゴリズムは整数の因数分解問題を解いたとされるが，実際には本質的に量子計算を用いたのは次の「周期」発見問題（位数発見問題と呼ばれる）だったからである．（位数発見問題を効率的に解きさえすれば，整数の因数分解問題もまた効率的に解けることは以前から知られていた.）

入力例. 自然数 N および N と互いに素な自然数 a（$< N$）

答え. a の位数 r，すなわち $a^x = 1 \bmod N$ をみたす最小の自然数.

位数発見問題における隠れた構造とは，関数 $f(x) = a^x \bmod N$ が $f(x) = f(x + r)$ なる周期関数ということであり，Shor はこの周期（＝位数）r を量子アルゴリズムを使って効率的に抽出したのである．

5 量子計算と NP 完全問題

Shor のアルゴリズムが発見された後，「量子計算の底知れぬ能力なら NP 完全問題（岩間氏の解説参照せよ）も効率的に解けるのではなかろうか」とは多くの人が考えたことである．今なお Shor

のアルゴリズムとともに量子計算における 2 大アルゴリズムといえる Grover のアルゴリズムも，当初は NP 完全問題を解くことを目的として頑張ってたら出来上がったアルゴリズムだと風の噂に聞いたことがある．Grover のアルゴリズムは，次のようなタイプの問題に対して適用可能である．

入力例. n ビット列 x および関数 $f : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$．ただし，$m$ は n の多項式以下.

答え. $f(x,y) = 1$ となるような $y \in \{0,1\}^m$ が存在すれば 1，存在しなければ 0．

実は NP に属する問題は（NP 完全問題も含め）すべてこの問題の形にみなすことが可能である．関数 f は NP 問題ごとに異なる．この問題を解くために普通のコンピュータで虱潰し的に $f(x,y) = 1$ となる y を次々試していくなら f を最悪 $\Omega(2^m)$ 回評価しなければならない．Grover が示したことは量子アルゴリズムを使えばこの問題を解くには f を $O(\sqrt{2^m})$ 回評価すれば十分だということである．しかし，（驚くことに Grover のアルゴリズムと独立かつほぼ同時期に別の研究者たちによって示されたことなのだが）Grover のアルゴリズムは最適であった．つまり，量子計算を使っても（f に関する情報を f を評価する以外に得る方法がない場合は）f を評価する回数は $\Omega(\sqrt{2^m})$ 必要だというわけである．だが，このことは NP 完全問題が量子計算で効率的に解けないことを言っているわけではない．実際の NP 完全問題は関数 f がもっと情報を含んだ形で与えられていて，関数 f の評価をしなくても別の方法で効率的に解けてしまうかもしれないからである．そうはいっても Grover のアルゴリズムの最適性の結果が出て以降は，NP 完全問題に対する量子アルゴリズムは望み薄という雰囲気が漂っているのもまた事実である．

6 最後に

量子計算の雰囲気を理工系の学部学生にわかってもらおうと奮闘してみたが，筆者の力不足のせいで伝わってるかどうか正直微妙な気もする．いくらかの情報源を記すことでお許しいただきたい．量子計算の最もスタンダードなテキストは [5] であろう．このテキストは量子計算だけでなく量子情報，量子暗号，果ては実験に至るまでカバーした量子情報科学の王道テキストである．本稿の結果の参考文献もこのテキストを参照して欲しい．ネックは分厚いことである．[3] は「量子版 NP」に関する記述もあり，かなり高度な内容となっている．[2] は比較的新しいテキストであり，量子アルゴリズムに関して詳しい．[4] は（著者独特の癖のある部分が散見されるものの）かなり丁寧に書かれた初学者には優しいテキストである．[1] は計算量理論のテキストであるが，量子計算の章もあり，コンパクトにまとまってて主要なことだけ知りたい人にはよいかもしれない．

参考文献

[1] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, Cambridge, 2009.

[2] P. Kaye, R. Laflamme, and M. Mosca. *An Introduction to Quantum Computing*. Oxford University Press, 2007.

[3] A. Y. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*. Graduate Studies in Mathematics, Vol. 47, American Mathematical Society, Rhode Island, 2002.

[4] N. D. Mermin. *Quantum Computer Science: An Introduction*. Cambridge University Press (2007).（邦訳: 量子コンピュータ科学の基礎，木村元訳，丸善, 2009.）

[5] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2000.（邦訳：量子コンピュータと量子通信（３分冊），木村達也訳，オーム社, 2005.）