動かしながら始める量子コンピュータ

第2回　最初に見つかった量子アルゴリズムをシミュレーションしてみる

束野 仁政

最近，ニュースなどで扱われることが増えてきた量子コンピュータですが，どのような原理で動作し，どのような問題を解決し，どのように扱えるのかを紹介します．今回は量子コンピュータ特有のアルゴリズム「ドイッチュ - ジョサのアルゴリズム」を紹介します．このアルゴリズムは，量子コンピュータによって古典コンピュータよりも高速に問題が解けることが分かった最初のアルゴリズムです．

具体的なアルゴリズムの解説後，Qiskitを利用してプログラミングを行います．プログラミングを通じて，量子コンピュータが動作する様子を見てみましょう．

量子コンピュータの可能性を最初に示した「ドイッチュ - ジョサのアルゴリズム」

● あのファインマン先生らの概念がホントだと可能性を示した

1980年代にファインマンやドイッチュらによって量子コンピュータの概念が提唱されました．しかし，当時はまだ「量子コンピュータが古典コンピュータより高速に問題を解けるアルゴリズム」は見つかっていませんでした．

1992年，ドイッチュとジョサがこのようなアルゴリズムを初めて見つけました(1)．これが，ドイッチュ - ジョサ注1のアルゴリズムです．このアルゴリズムそのものは実用上の重要な問題を解決するものではありませんが，最初に見つかった「量子コンピュータが古典コンピュータより高速に問題を解けるアルゴリズム」として意義があります．ここでは，改良版のドイッチュ - ジョサのアルゴリズムについて解説します(2)．

● 考える問題

まず，古典コンピュータの問題から解説します．1ビットの値を入力とし，1ビットの値を出力する関数を考えます．このような関数は4通りあります(**表1**)．ここで，全ての出力が同じ値となる関数を定数関数と呼びます．例えば，$f_{00}(0)=0$，$f_{00}(1)=0$であるため，関数f_{00}の出力は全て0になります．そのため，関数f_{00}は定数関数であり，同じように関数f_{11}も定数関数です．

次にバランス関数は「$f(x)=0$になるxの個数」と「$f(x)=1$になるxの個数」が等しい関数のことです．例えば，$f_{01}(0)=0$，$f_{01}(1)=1$となるため，「$f(x)=0$になるxの個数」と「$f(x)=1$になるxの個数」はそれぞれ1個です．そのため，関数f_{01}はバランス関数であり，同じように関数f_{10}もバランス関数です．

表1に示した，いずれかの関数がブラック・ボックス関数fとして与えられたとき，関数fが定数関数かバランス関数かを判定するにはどうすればよいでしょうか．ここで，ブラック・ボックス関数とは，「具体的な関数の中身は不明だが，実行はできる関数」のことです．単純な方法としては，過半数の入力について関数$f(x)$の値を確認することで定数関数かバランス関数かを判定できます．関数fの入力が**表1**のように1量子ビットの場合，関数fを2回実行し$f(0)$と$f(1)$の値を**表1**と照らし合わせることで判定できます．この方法の場合，過半数の入力について関数$f(x)$の値を確認する必要があるため，Nビット（入力が2^N通りある）の場合は関数fを$2^{(N-1)}+1$回実行する必要があります．

● 古典コンピュータだと1回で判定できない

関数fを1回実行しただけで定数関数かバランス関数か判定できないでしょうか．0を入力として関数fを実行すると，$f(0)=0$または$f(0)=1$となります．$f(0)=0$の場合，f_{00}とf_{01}の2つの可能性が考えられるため，この情報だけでは定数関数かバランス関数かを判定できません．また，$f(0)=1$の場合もf_{10}とf_{11}の2つの可能性が考えられるため，同じように，定数関数かバラ

注1：文献(2)では「ドイッチュ」だけの表記になっていますが，多くの文献に従い「ドイッチュ - ジョサ」と表記します．