
量子計算の古典的検証
Classical Verification of Quantum Computation

Agenda Part Ⅰ

「量子計算の古典的検証」問題とは何か？

- 「量子計算の古典的検証」とは何か？
- 量子コンピュータの働きをチェックし検証する-- 問題の難しさ
- Mahadevのアプローチ (1)-- 対話型証明
- Mahadevのアプローチ (2)-- 暗号化 --

Agenda Part II

背景

- 「量子超越性」をめぐる論争
- ファインマンの洞察
- ベルの定理とその実験での検証
- Interactive Proofと証明概念の転換
- 複雑性の新しいクラスBQPの発見

Agenda Part III

準備

- Learning with Errors
- Universal Quantum Circuit
- Trapdoor Claw-Free Functions

Agenda Part IV

Mahadev

- Quantum Certification Protocol


Part I
「量子計算の古典的検証」問題とは何か？

Agenda Part Ⅰ

「量子計算の古典的検証」問題とは何か？

- 「量子計算の古典的検証」とは何か？
- 量子コンピュータの働きをチェックし検証する-- 問題の難しさ
- Mahadevのアプローチ (1)-- 対話型証明
- Mahadevのアプローチ (2)-- 暗号化 --


「量子計算の古典的検証」とは何か？
量子計算の古典的検証
Classical Verification of Quantum Computation

「量子計算の古典的検証」とは？

「量子計算」というのは、量子コンピュータが行なう計算のことです。「古典的検証」を行うのは人間です。人間が古典的手段を使って、量子コンピュータが行なった計算が正しいかどうかをチェックすることを、「量子計算の古典的検証」と言います。

「古典的手段」とは、主要には、古典的コンピュータのことを指します。これも、ピンとこない言い回しかもしれません。それは、量子コンピュータとの対比で古典的と言われているだけで、最新鋭のスーパーコンピュータを含む、普通のコンピュータのことです。

「量子計算の古典的検証」とは、量子コンピュータの行なった計算が正しいものであるかを、人間が普通のコンピュータを使って、確かめると言うことです。

ショアのアルゴリズムの場合

量子コンピュータのアルゴリズムとして有名な素因数分解を行う「ショアのアルゴリズム」の場合でしたら、「量子計算の古典的検証」は簡単です。

入力した数に対して量子コンピュータが出力した素因数を、コンピュータで実際に掛け算を行なって、それが入力したものと一致しているか確かめればいいわけですから。

ただ、量子コンピュータが行う計算の正しさが、コンピュータで簡単にチェックできるとは限りません。

Google IBM論争の場合

2019年のGoogleの量子優越性実証実験についての、GoogleとIBMの論争は、直接的には、50数個のqubitを持つ量子コンピュータの「古典的検証」の手法をめぐるものでした。

Googleは、検証にはスーパー・コンピュータを使っても「一万年」かかるとしたのに対して、IBMはやり方を工夫すれば、「二日半」で検証できると主張しました。

IBMは、現在、400 qubitの量子コンピュータを開発しているといいます。こうしたマシンでの「量子計算」の「古典的検証」は、かつて IBMがGoogleに対して主張したような手法で、高速化できるのでしょうか？　それは、不可能です。

ファインマンの場合

量子コンピュータのアイデアは、ファインマンの古典的なコンピュータ（スーパーコンピュータを含む普通のコンピュータのことです）では、量子の法則に従う自然をシミレートできないという問題意識から始まりました。

「だから、量子の法則に従う量子コンピュータを作ろう！」

そもそも、量子コンピュータは、普通のコンピュータでシミレートできない計算をするものとして考えられていました。

「量子計算の古典的検証問題」とは？

一方で、普通のコンピュータにない量子コンピュータの能力に期待しながら、他方では、その成果を普通のコンピュータを通じて、人間が利用できると考えているわけです。

「量子コンピュータ」対「古典コンピュータ＋人間」の対比で言うと前者と後者の計算能力は明らかに異なっています。普通のコンピュータと同様に、人間の能力は、「古典的」なものであることに、注意してください。

「量子計算の古典的検証問題」というのは、この問題を、原理的にキチンと考えようということです。直観的には、二つの陣営の能力差が大きすぎて、「量子計算の古典的検証」は不可能に思えます。

「量子計算の古典的検証問題」の現実的・実践的意味

それでは、現在のものより、さらに大きな量子コンピュータが完成した時、我々は、それが正しく動作していることをチェックする方法も、その量子コンピュータが行った計算が正しいのかの検証も、原理的には存在しないということになります。

もしも、将来、量子コンピュータの利用が進むなら、おそらくそれはクラウド上の量子コンピュータに、みながアクセスする形になると思うのですが、その時、量子コンピュータに成りすました偽の量子コンピュータが出現した時、それを偽物と見破る方法はないのでしょうか？

逆に、もしも、「量子コンピュータなんか信じない」という人が存在したら、その人に、量子コンピュータのパワーを納得させる方法はあるのでしょうか？

「量子計算の古典的検証問題」の解決 Urmila Mahadevの登場

「量子計算の古典的検証問題」は、2004年にGottesmanによって定式化されたと言われています。はやくから、問題は認識されていて、活発に研究されてきました。

ただ、この問題が解決されたのは最近のことです。彗星のように登場した若い大学院生 Urmila Mahadev によってこの問題は、肯定的に解決されたのです。

「量子計算の古典的検証は可能である！」

量子コンピュータにとっても人間にとっても意味のある、とても大事な発見でした。


Urmila Mahadev


量子コンピュータの働きをチェックし検証する
-- 問題の難しさ --
量子計算の古典的検証
Classical Verification of Quantum Computation

マハデフ、「量子計算の古典的検証問題」を解く

Mahadevは、量子コンピュータが行う計算についての最も基本的な問題である、「量子計算の古典的検証問題」を解きました。

我々人間が、量子コンピュータにある計算を実行させた時、量子コンピュータが我々の指示通りに計算を行ったのか、我々の予想もしなかった量子の奇妙な振る舞いで我々の意図とは違うデタラメな（我々にとって）計算をしたのかを、我々人間がチェックできるのかという問題は、ひとまず「大丈夫。人間がチェックできる」という形で解決されました。

ここでは、まず、この問題が難しい問題であることを確認してみたいと思います。

普通のコンピュータ･プログラムを チェックする時、我々が行うこと

我々が、プログラムをデバッグしようとする時、まず、プログラムの中の変数がどのような値を取っているかをチェックします。プログラム実行中のコンピュータの「状態」は、各変数へどのような値が割り当てられているかで決まります。

次に、プログラムのステップが進んだ時の、変数の値の変化をチェックします。変数の値の変化は、コンピュータの状態の変化をもたらすのですが、コンピュータの「ふるまい」というのは、この状態の変化の継起する系列に他なりません。

コンピュータの振る舞いが、正しいものかは、最終的には変数への値の割り当てに帰着する状態の変化を追いかければチェックできます。

量子コンピュータがとる状態

n個のqubitからなる量子コンピュータの場合、それがとる「状態」は、次のように表現することができます。

1-qubit の場合

$$\alpha_0|0> + \alpha_1|1>$$

2-qubit の場合

$$\alpha_{00}|00> + \alpha_{01}|01> + \alpha_{10}|10> + \alpha_{11}|11>$$

3-qubit の場合

$$\alpha_{000}|000> + \alpha_{001}|001> + \alpha_{010}|010> + \alpha_{011}|011> + \alpha_{100}|100> + \alpha_{101}|101> + \alpha_{110}|110> + \alpha_{111}|111>$$

n-qubit からなる量子コンピュータの状態

一般に、n-qubit からなる量子コンピュータの状態は、次のように表されます。ここで、$\{ 0, 1 \}^n$ は、0と1からなるn個の並びを著しています。これは2^n個存在します。

n-qubit からなる量子コンピュータの状態

$$\sum_{x \in \{ 0,1 \}^n} \alpha_x |x >$$

このことは、n-qubit からなる量子コンピュータのある一つの状態は、2^n個存在する α_xの値で決まることを意味します。普通のコンピュータで言えば、2^n個の変数があることに相当します。

量子コンピュータの働きを
チェックし検証することの難しさ（1）

量子コンピュータの状態を決めているのは、この2^n個の変数です。我々は、この2^n個の変数の変化をトレースできるでしょうか？

普通のコンピュータ・プログラムで変数の数nが 50を超えるのは珍しくないかもしれません。ただ、nが大きくなるにつれ、2^n個の変数のトレースは、我々の手に余るようになるのは明らかです。

量子コンピュータの働きを
チェックし検証することの難しさ（2）

量子コンピュータの働きをチェックし検証することの難しさは、実は、もう一つあります。それは、我々は、量子コンピュータの状態を正確に知ることはできないという問題です。

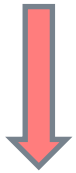
量子コンピュータの状態は、先に見たように2^n個の状態の「重ね合わせ」になっているのですが、その状態を知ろうとして観測すると、その「重ね合わせ」の状態は失われてしまいます。

観測で得られるのは、n個の古典bit だけです。

実際に観測される量子コンピュータの状態

n-qubit からなる量子コンピュータの状態

$$\sum_{x \in \{0,1\}^n} \alpha_x |x> \qquad 2^n \ bit$$



$$n \ bit$$

観測される量子コンピュータの状態

状態とその変化を正確に知らないで、その振る舞いの検証はできない

普通のコンピュータでは、その振る舞いの正しさを検証するためには、コンピュータの状態とその変化を知ることが不可欠です。それは、量子コンピュータでも同じです。

普通のコンピュータのデバッグでは、コンピュータの状態とその変化は、基本的には変数の値をトレースすることで可能なのですが、量子コンピュータでは、そもそも、状態を決定している変数の値に直接アクセスすることができないのです。

マハデフは、こうした難しさをどのように解決したのでしょうか？


Mahadevのアプローチ (1)
-- 対話型証明 --
量子計算の古典的検証
Classical Verification of Quantum Computation

原理が異なる二つのシステムの境界での相互作用

前回、普通のコンピュータのデバッグや動作検証のようなやり方は、量子コンピュータの検証には使えそうもないという話をしました。

そのことは、現在の量子コンピュータが、普通のコンピュータと対比でいうと、ハードとソフトの分離もされていないし、OSとアプリケーションの区別もない状態にあることと、少し関係はあります。ただ、量子コンピュータには開発キットもデバッガーもないことを言いたかったわけではありません。

「量子計算の古典的検証」の問題は、原理が異なる二つシステムの境界での相互作用はいかにして可能なのか、あるいは、それにはどのような限界があるのかという、量子コンピュータと我々の関係にとって極めて本質的な問題です。

Mahadevの方法への期待

量子コンピュータの状態の遷移を正確には追えないだけではありません。我々は、現状では、量子計算の出発点となる、任意のqubitの状態を量子コンピュータ上にセットするというプリミティブな操作でも、その検証を含めれば、思い通りには量子コンピュータをコントロールできていないのです。

Mahadevの「量子計算の古典的検証」問題の肯定的解決は、こうした現状を変えて、量子コンピュータに対する人間のコントロール能力を大きく拡大する可能性を秘めているのです。

Mahadevの二つのアプローチ

Mahadevは、「古典的検証問題」に二つの方向からアプローチします。

第一のアプローチは、今回のセッションで紹介する「対話型証明 = Interactive Proof 」の手法を使います。「検証問題」の解決を可能にする、量子コンピュータと人間の「対話のプロトコル」を見つけようとします。

第二のアプローチは、量子コンピュータでも破れないと考えられている、「Post量子暗号」、先日セミナーを行ったLWE(Learning with Errors)を使います。量子コンピュータの中に、量子コンピュータも改変できない、我々は知っているが量子コンピュータには秘密の状態を作ります。これについては次回のセッションで。

Interactive Proof 紹介

Interactive Proofの想定は、いささか奇妙なものです。
ここでは、その概略を紹介しようと思います。

Interactive Proofの想定
Interactive Proofには、二人の人物が登場します

- 一人は「証明者 Prover」です。
  彼は、全知全能で、どんな問題も瞬時に答えを返す能力をもっています。ただし、彼は誠実ではなく、時々、人を欺く嘘をつきます。

- もう一人は、「検証者 Verifier」です。
  彼は、普通の人間です。理性をもっていて、証明者の主張を検証しようとします。彼は、証明者の言うことを、盲信はしません。

「不実な全能者」と「理性ある人間」
二人の対話は何を明らかにするか？

- Interactive Proofは、この両者「不実な全能者」と能力は限られているが「理性的な人間」が対話を繰り返すことで、何かを証明しようというシステムです。

- 基本的には、検証者が証明者の主張を受け入れた時、すなわち、証明者が証明者の主張が正しいと検証者を納得させることができた時、証明は終わります。

- ただし、ある場合には、証明者の検証者の説得は失敗し、検証者は証明者の主張を拒否して、証明が終わります。

証明者。全知全能である。
どんな問題も瞬時に答えを返す能力をもつ。ただし、時々嘘をつく。

Question

Answer

IP

Interactive Proof

二人が、対話を繰り返すと
何が証明できるか？

検証者。普通の人間である。
理性をもっていて、証明者の主張を検証しようとする。盲信はしない。

Interactive Proofの最初の成功例「グラフの非同型問題」

- 同じ頂点数 n を持つグラフG_0とG_1が与えられた時、この二つのグラフG_0とG_1が同型でないことを決定する問題を考える。これを「グラフの非同型問題」という。

- グラフの「同型問題」は、「グラフG_0とG_1は同型である」という証明が与えられた時、その証明が正しいことの検証は簡単にできる。証明が与える「頂点と頂点の対応、辺と辺の対応」をチェックすればよい。グラフの「同型問題」は、NP問題である。

- ところが、「グラフの非同型問題」については、非同型であることを多項式時間でチェックするアルゴリズムが与えられない限り、この問題が、NPに属するかは、よくわからない。

Interactive Proofのスタイルで グラフの非同型問題を考える

ここでは、次のようなInteractive Proofのプロトコルで、グラフの非同型問題を考える。

1. グラフG_0とG_1は、ProverにもVerifierにも与えられている。
2. Verifierは、ランダムに0,1の値 i を選んで、G_0とG_1のいずれかのG_i (iは、0または1)の頂点をスクランブルしたグラフHをProverに送る。G_iとHは同型である。
3. Proverは、送られたHが、G_0から作られたものなら0を、G_1から作られたものなら1を、j として返す。
4. Verifierは、i=jなら受理し、そうでないならrejectする。

「グラフの非同型問題」の解決

1986年　Goldreich, Micali, and Wigerson

- グラフG_0とG_1が同型でないなら、Proverは、確実に、Hからjを導くことができる。Proverのパワーで、HがG_0とG_1のどちらかに同型かを調べればいい。

- ただ、グラフG_0とG_1が同型の場合、Proverは、確実には、Hからjを導くことができない。 HはG_0ともG_1とも同型だから。この時、ProverはVerifierに、ランダムに0か1を送ることになる。i=jである確率は1/2になる。

「Bellの定理」とInteractive Proof

1990年代、Interactive Proofは快進撃を始める。
Shamirの IP=PSPACEの証明、BabaiらのMIP=NEXPの証明はは、その代表的な成果である。

21世紀に入って、Interactive Proofと「Bellの定理」の類似が発見される。

1990年 Shamir

IP =PSPACE

Answer

Question

$\mathcal{IP}$

Interactive Proof

1991年 Babai, Fortnow, Lund

MIP= NEXP

Answer a

Answer b

Question x

Question y

MIP

Multi Prover
Interactive Proof

「Bellの定理」とInteractive Proofの類似の発見

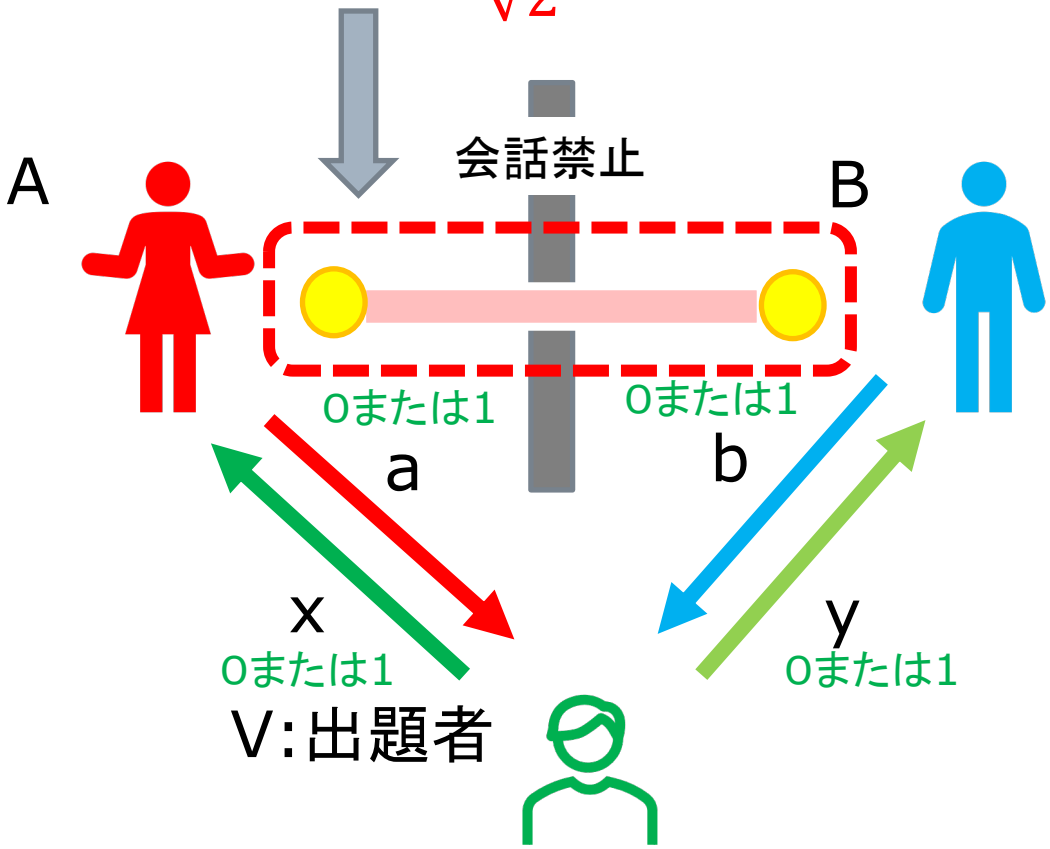
2004年、コンピュータ・サイエンティストのRichard Cleve, Peter Hoyer, Ben Toner, John Watrousらは、Bellの思考実験がInteractive Proofと極めて似ていることに気づく。

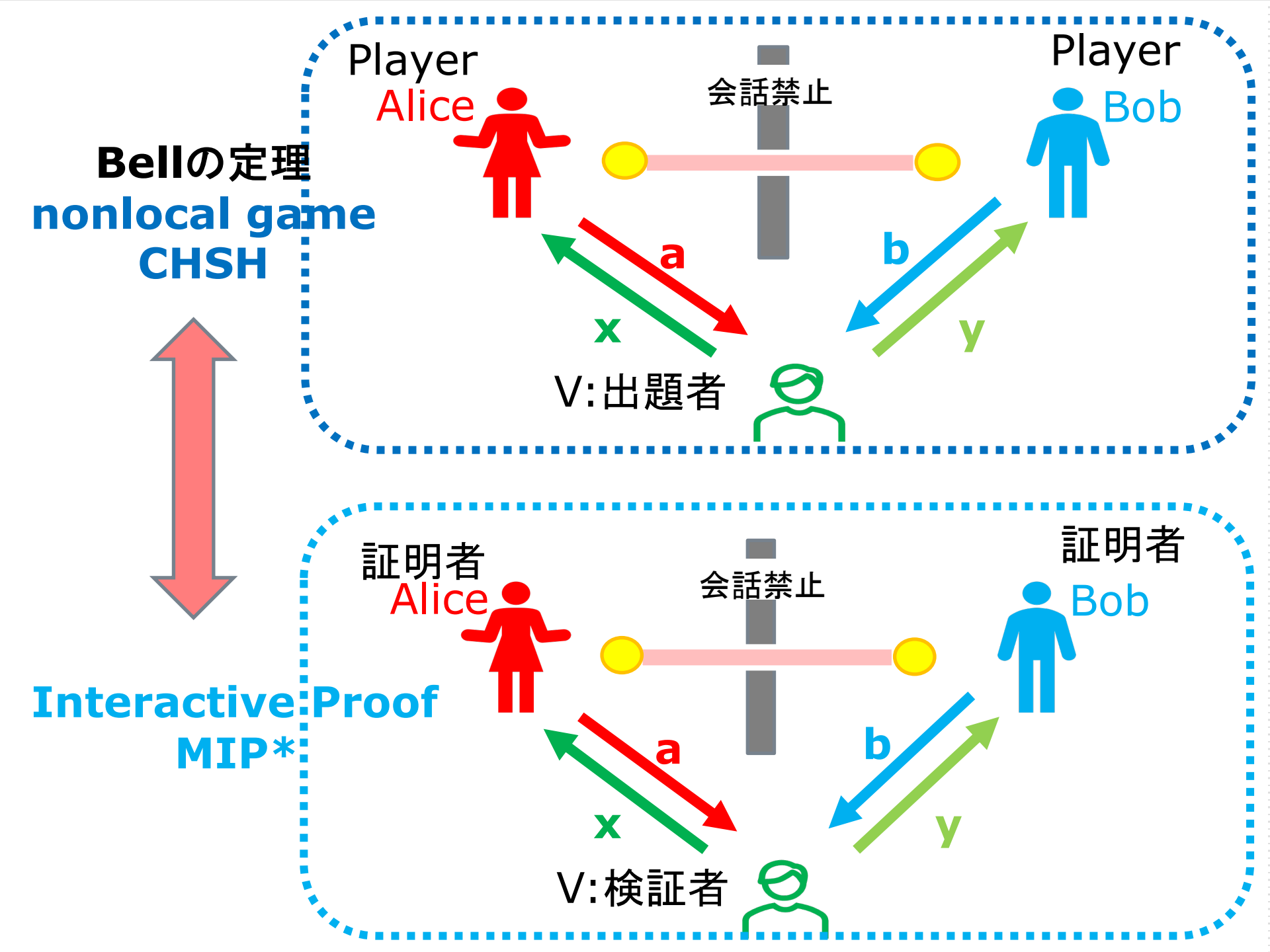
彼らは、量子的なMulti-Prover Interactive Proof MIP*を研究することを提案する。

Bellの定理のCHSHゲーム

A,Bはentangleした量子を共有する

$$|EPR>=\frac{1}{\sqrt{2}}(|00>+|11>)$$




Bellの定理
nonlocal game
CHSH
Player
Alice
会話禁止
Player
Bob
a
b
x
y
V:出題者
Interactive Proof
MIP*
証明者
Alice
会話禁止
証明者
Bob
a
b
x
y
V:検証者

Entanglement

Answer a

Answer b

Question x

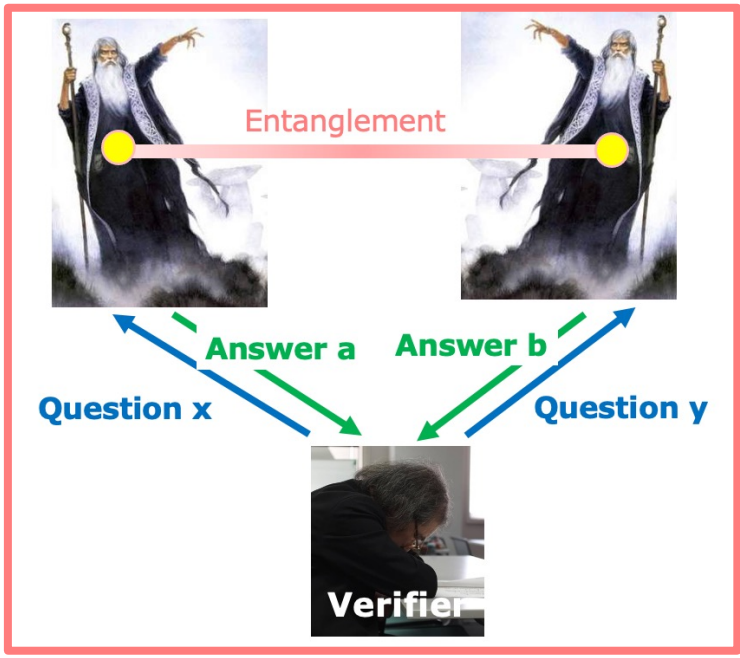
Question y

MIP^*

Multi Prover
Interactive Proof
with Entanglement

Interactive Proof の基本的達成

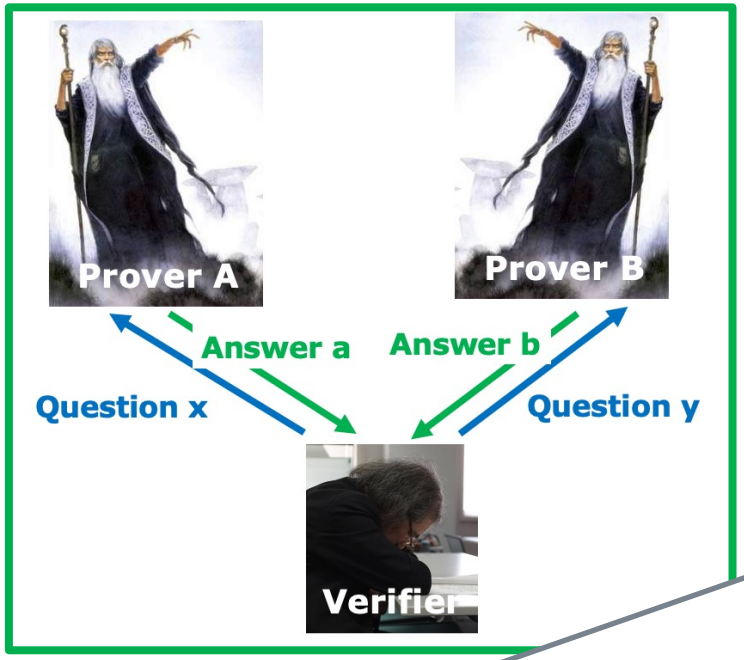
MIP*= RE



2020年 JNVWY

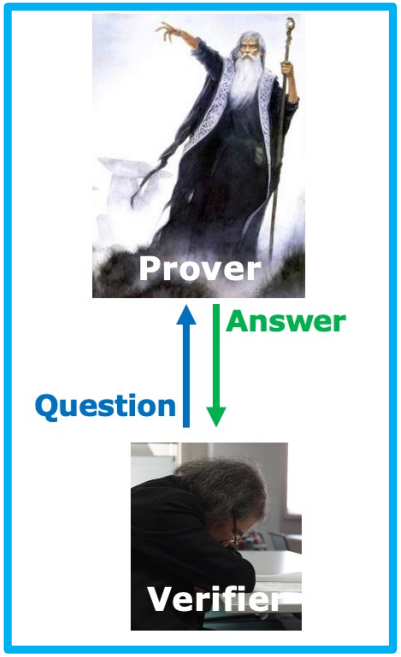
1991年 Babai, Fortnow, Lund

MIP= NEXP



1990年 Shamir

IP = PSPACE



対応する複雑性のクラス


Mahadevのアプローチ (2)
-- 暗号化 --
量子計算の古典的検証
Classical Verification of Quantum Computation

人間は、量子コンピュータをコントロールできるか？

「量子計算の古典的検証」問題というのは、簡単に言えば、量子コンピュータを人間のコントロールのもとにおこうと思ってもなかなか思うようにはいかないという問題です。

それには理由があります。複雑な状態を理解する能力にしても、計算能力にしても、我々は、何一つ量子コンピュータにはかなわないのですから。

その上、彼らは秘密主義で、自分の手の内を人間には見せません。複雑な内部の状態をチェックしようとすると、別の姿に、簡単な見掛けに、姿を変えてしまいます。

なぜ、暗号化技術が必要なのか？

Mahadev の「量子計算の古典的検証」問題へのアプローチの特徴は、先に見た「対話型証明」の手法の利用とともに、「暗号化」の技術を積極的に利用することです。

なぜそこに「暗号化」技術が出てくるのでしょう？

それは、圧倒的な力の差がある量子コンピュータに対して、彼らを出し抜いて人間が少しでも優位に立つ手段として「暗号化」が利用できる可能性があるからです。

彼らが秘密主義なら、我々も、彼らに対して「秘密」を持つことができると、状況は少し変わります。

「ポスト量子暗号」技術の特徴

量子コンピュータは、既存の暗号技術への脅威として意識されてきました。ショアのアルゴリズムは、素因数分解や離散対数問題を解くことが難しいことに基礎を置く、RSA暗号や楕円曲線暗号を簡単に解きうるからです。

ただ、その中で追求されてきた「ポスト量子暗号」技術は、次のような特徴を持ちます。

普通のコンピュータで実現できて、
量子コンピュータでも破られない暗号

「量子優越性」をチェックする

量子コンピュータでも破れない暗号があり、かつ、その暗号の解き方を我々が知っているなら、それは、量子コンピュータに対する我々の数少ない優位性として利用できます。

例えば、ポスト量子暗号技術 LWE(Learning with Errors) で暗号化したデータを、量子コンピュータに送り込みます。そしてそのデータの処理を量子コンピュータに実行させます。

量子コンピュータは、その暗号化されたデータが本当はどういうデータなのかはわからないまま、暗号化されたままのデータを処理した結果を人間に返します。

我々は、そのデータの元の形を知っていますので、返ってくる答えを知っています。何度もこうした対話を繰り返せし、返ってくる答えと予想した答えを比較すれば、少なくとも、対話の相手が量子コンピュータかそうでないかの判断がつくようになります。

これは、「量子計算の古典的検証」の一つの課題であり、現在は、スーパーコンピュータが膨大な時間をかけて検証するしかない「量子優越性」のチェックが簡単に行えることを意味します。

量子コンピュータをクラウドに

データの中身は秘密のまま、計算能力の高いマシンに必要な処理だけをさせる暗号化を「準同型暗号」といいます。

Mahadevが行ったことは、「ポスト量子暗号を準同型暗号化して、その量子コンピュータ版を作って、それを通じて量子コンピュータを検証する」というふうにまとめられます。

Mahadevらの暗号化技術は、「量子計算の古典的検証」としてだけではなく、量子コンピュータをクラウドとして利用する道を開く可能性も秘めているのです。


Part II
背景

Agenda Part II

背景

- 「量子超越性」をめぐる論争
- ファインマンの洞察
- ベルの定理とその実験での検証
- Interactive Proofと証明概念の転換
- 複雑性の新しいクラスBQPの発見

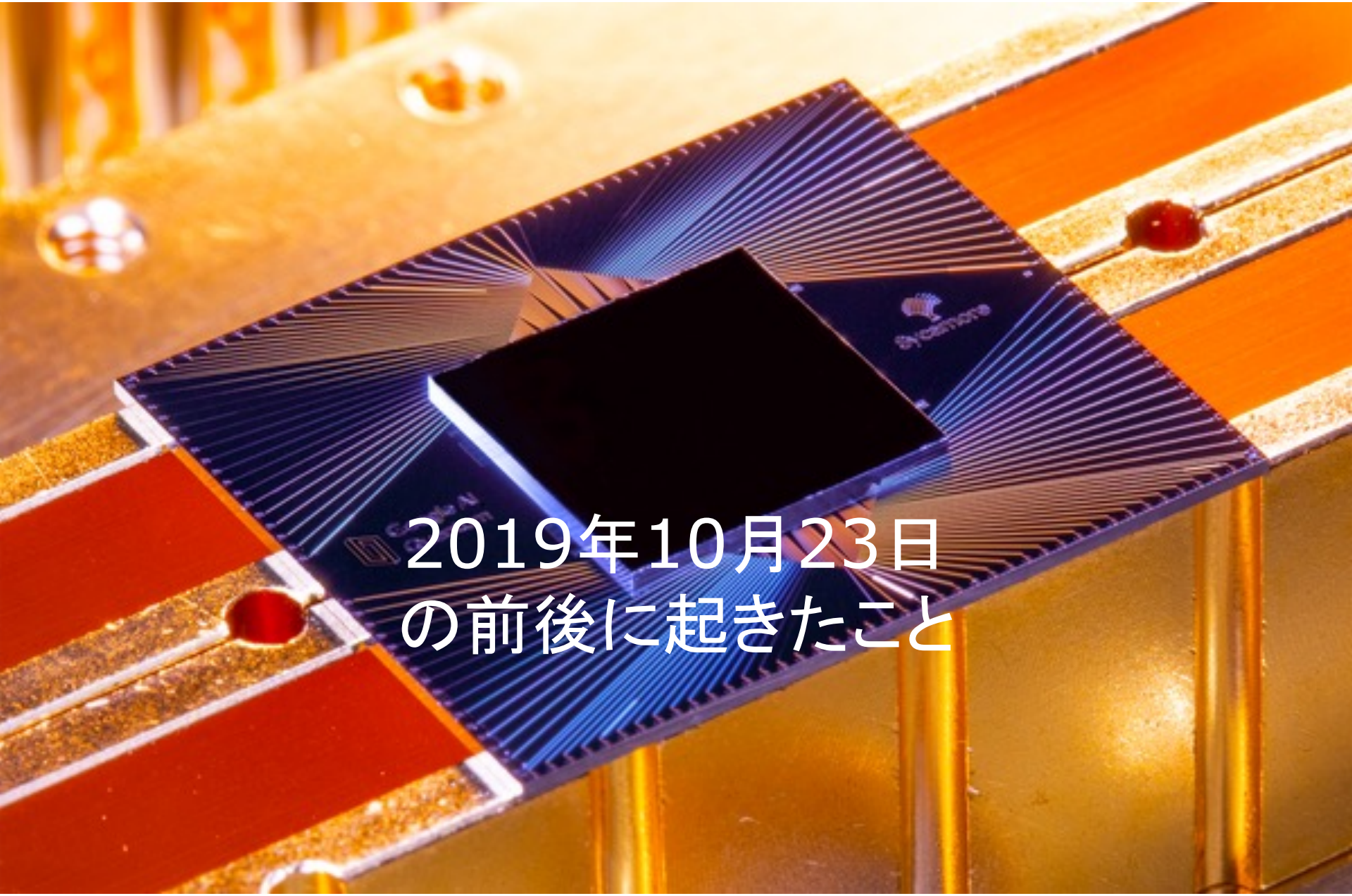

「量子超越性」をめぐる論争
-- 3年前の10月に起きたことを振り返る --
量子計算の古典的検証
Classical Verification of Quantum Computation

「量子計算の古典的検証」問題とは何か？

11月のセミナーでは、量子コンピュータの計算の正しさを、古典コンピュータ（現在の、普通のコンピュータのこと）で検証できるのかという、「量子計算の古典的検証」問題を取り上げようと思う。

これがどういう問題かを示すために、今回は、ちょうど3年前の2019年の10月に起きた、「量子超越性」をめぐるGoogleとIBMの論争を、改めて紹介する。

2020年2月のマルレク https://www.marulabo.net/docs/q-supremacy/
「量子コンピュータの現在 -- 量子優越性のマイルストーンの達成 --」を参照されたい。


2019年10月23日
の前後に起きたこと

Article | Published: 23 October 2019

Quantum supremacy using a programmable superconducting processor

Frank Arute, Kunal Arya, [...] John M. Martinis

Nature **574**, 505–510(2019) | Cite this article

Abstract

The promise of quantum computers is that certain computational tasks might be executed exponentially faster on a quantum processor than on a classical processor[1]. A fundamental challenge is to build a high-fidelity processor capable of running quantum algorithms in an exponentially large computational space. Here we report the use of a processor with programmable superconducting qubits[2,3,4,5,6,7] to create quantum states on 53 qubits, corresponding to a computational state-space of

2019年10月23日
Natureに論文発表

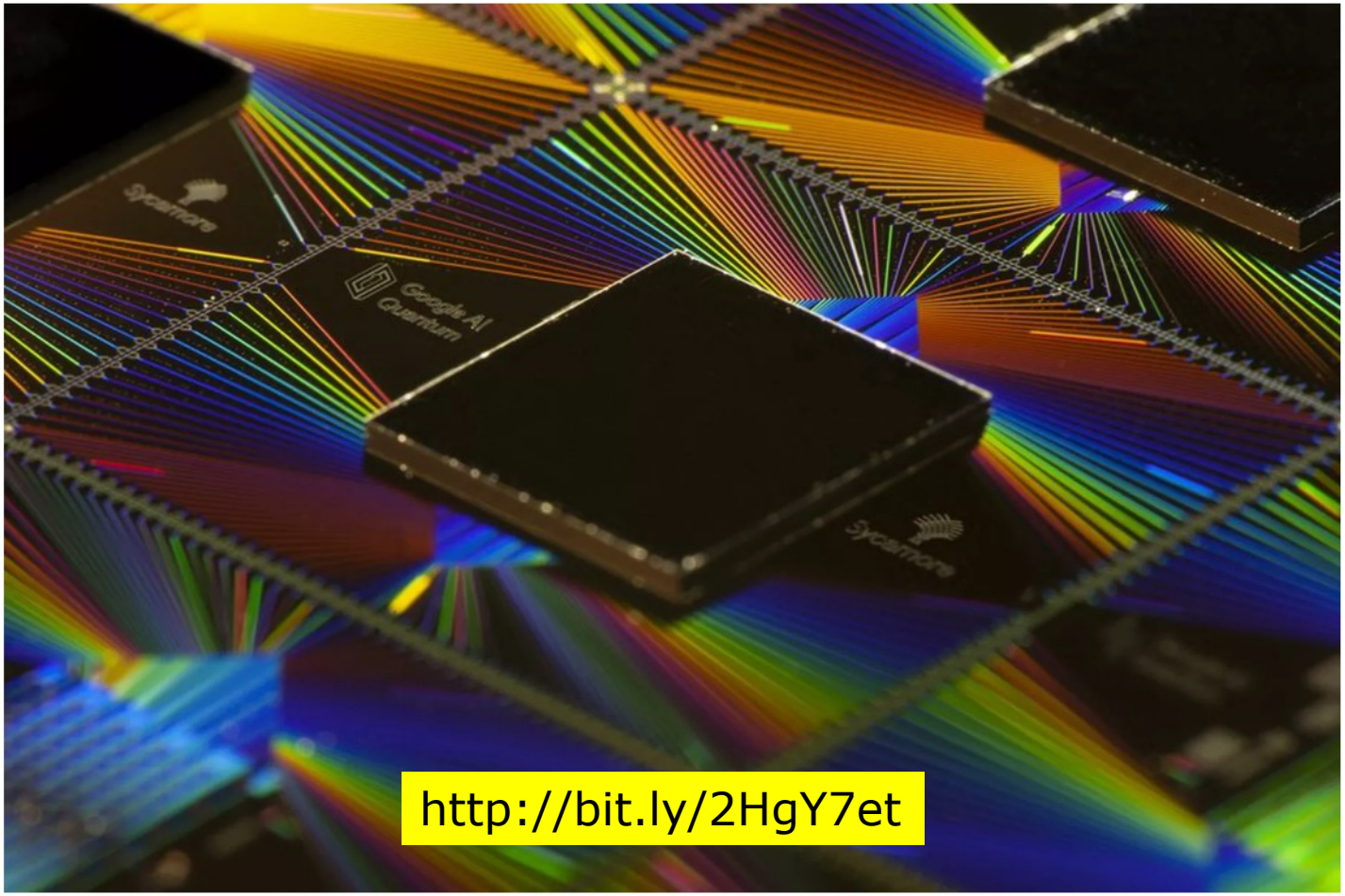
circuit a million times—our benchmarks currently indicate that the equivalent task for a state-of-the-art classical supercomputer would take approximately 10,000 years. This dramatic increase in speed compared to all known classical algorithms is an experimental realization of quantum supremacy[8,9,10,11,12,13,14] for this specific computational task, heralding a much-anticipated computing paradigm.

Google confirms 'quantum supremacy' breakthrough

Its research paper is now available to read in its entirety

By Jon Porter | @JonPorty | Oct 23, 2019, 6:31am EDT

SHARE



http://bit.ly/2HgY7et

ビットコイン、7500ドル割れ　量子コンピューター警戒

2019/10/23 23:59

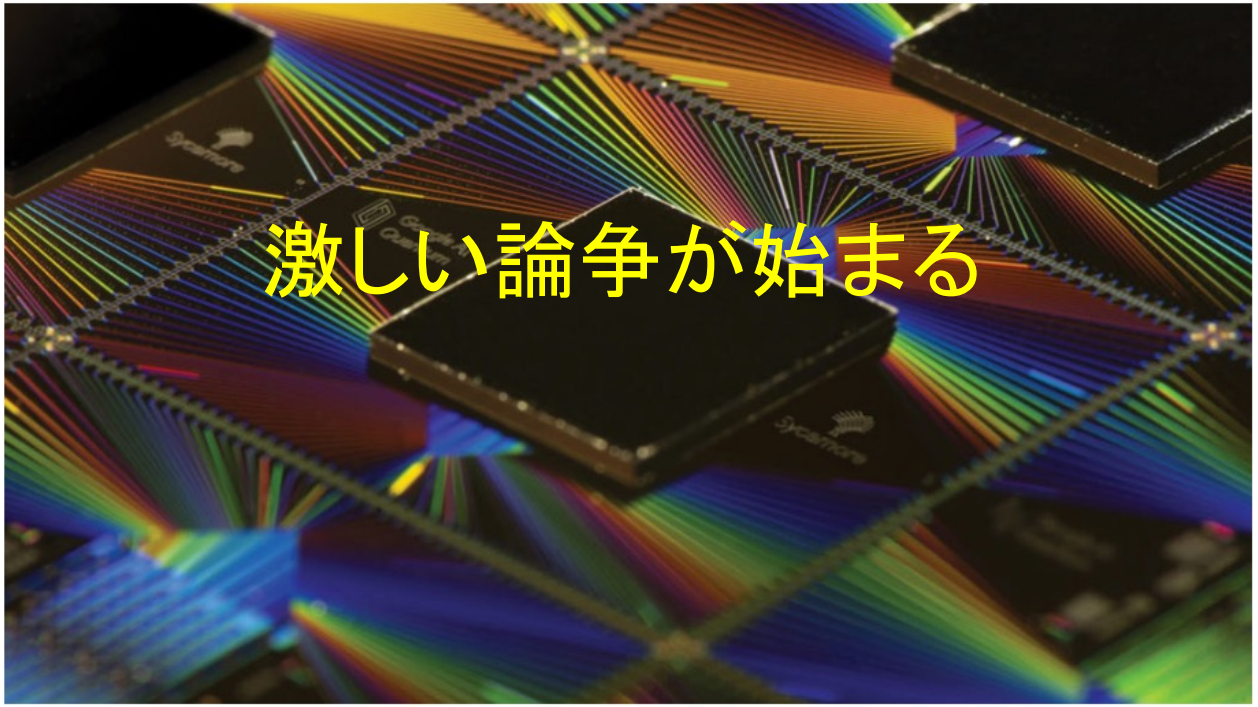
https://s.nikkei.com/2SgvCE0

ScienceNews
INDEPENDENT JOURNALISM SINCE 1921

YEAR IN REVIEW QUANTUM PHYSICS

Google claimed quantum supremacy in 2019 — and sparked controversy

Competitors questioned whether the milestone had truly been achieved



Google's quantum computer Sycamore performed a calculation that would take thousands of years with a classical supercomputer, researchers claimed in 2019. An array of quantum computer chips is shown.

GOOGLE

http://bit.ly/31MgLV6

IBM論文の指摘

"Leveraging Secondary Storage to Simulate Deep 54-qubit Sycamore Circuits"

Edwin Pednault, John A. Gunnels, Giacomo Nannicini, Lior Horesh, Robert Wisnieff **2019/10/22**

https://arxiv.org/abs/1910.09534

最強のスーパーコンピュータを使えば、10,000年ではなく、2.5日でシミレートできる

Googleの発表の一日前に公開されたIBMの5名の研究者による論文は、興味深いものである。

彼らは、オークリッジの国立研にある、世界最強のスーパーコンピュータであるSummitを使えば、Google の 53-54 qubitのSycamore プロセッサーの出力のシミレートを、Googleのいう10,000年ではなく、2.5日でシミレートできるという。

これは、本当かもしれない。

2^{53} 個の量子の状態を、メモリー上ではなく、250ペタバイトのディスク上に置く

彼らのアイデアは、シミレートすべき2^{53} 個の量子の全状態を、メモリー上ではなく、Summitの250ペタバイトのディスク上に置くというものであった。（Googleのスーパーコンピュータでのシミレートは、量子の状態をメモリーの上に置いていた。）

もちろん、シミレーションのアルゴリズムも、違ったものになる。ただ、それによって、Sycamoreのシミレーションは、10,000年ではなく、2.5日で可能になるという。

本当かもしれない。

チェックしてみよう

ここでは、2^{53} 個の量子の全状態を、メモリー上ではなく、Summitの250ペタバイトのディスク上に置けるかどうかをチェックしてみよう。

$2^{10} \approx 1K$ キロ とすると、
$2^{20} \approx 1M$ メガ
$2^{30} \approx 1G$ ギガ
$2^{40} \approx 1T$ テラ
$2^{50} \approx 1P$ ペタ である。

量子の状態は、二つの複素数の組みで表されるから、四つの実数の組みと考えていい。実数が4バイト(32ビット)で表現されるなら、量子の状態は16バイトで表現される。

$2^{53} = 2^3 \cdot 2^{50} =$ 8ペタ個の量子の状態は、
8ペタ x 16バイト = 64ペタバイトの容量を持つ。

64ビットの実数なら、128ペタバイトになる。ギリギリ、セーフ？

IBM論文に対する Aaronsonの指摘

“Quantum supremacy: the gloves are off”

Scott Aaronson **2019/10/23**

https://www.scottaaronson.com/blog/?p=4372

アーロンソン反論を開始する

アーロンソンは、このブログ投稿で、自分がGoogleのNature論文の査読者であったことを初めて明かす。 査読者なので論文公開まで、「批判」に反論できなかったという。また、Googleのスーパー・コンピュータでのシミレーションが、自分の提案したアルゴリズムに基づいていたことを認める。（それは、Googleの論文の注を見ればわかる。）

彼は、このIBM論文を、Google論文の最初のリークに対するIBMのフィナンシャル・タイムズでの冷笑的なコメントよりずっとマシなものだとする。 さらに、Googleも、もっとしっかりやれば良かったのにと言う。

ただ、アーロンソンは、次のように語る。
「この分析は、Googleの実験では量子優越性が、達成されていないことを意味するのであろうか？　断じてそうではない。」

量子プロセッサ Sycamoreは、世界最速のスーパー・コンピュータSummitより、1,200倍早い

Sycamoreは、500万個のデータサンプルを得るのに、約3分かかった。IBMの論文では、それは 1万年ではなく 2.5日で計算できると言う。でも、3分 vs. 2.5日は、1,200倍の違いだ。
量子プロセッサ Sycamoreは、世界最速のスーパー・コンピュータSummitより、1,200倍早いのだ。

しかし、もっと重要なのは、実験で利用された「基本的演算」の数の比較だ。量子コンピュータの場合、基本演算の数は量子ゲートの数だが、スーパーコンピュータの場合、「基本的演算」は「浮動小数点演算 FLOP」だ。

アーロンソンの試算によれば、今回の実験での基本演算の数は、Sycamoreが $5x10^9$ 量子ゲート、Summitが2×10^{20} FLOPsで、その比は、400億倍にもなると言う。

53qubitではなく 55qubitになれば、Summitが二台必要になる。70qubitの量子コンピュータでは？

全てをハードディスクに格納すると言うスタイルでは、例えば、量子プロセッサーのqubitが実験時のsycamoreの53qubitから55qubitに増えれば、明らかにSummitの250ペタバイトのハードディスクの容量を超えることになる。55qubitでは、Summitが2台必要になる。

5qubit増えて60qubitになれば、qubitの状態の数は $2^5=32$倍になるから、さらに32台のSummitが必要になる。
70qubitになればどうだろう？ 60qubitから10qubit増えると、状態の数は、$2^{10}=1024$倍になる。だから、70qubitの量子プロセッサーをシミレートするには、3,000台以上のSummitが必要だと言うことになる。

（Googleは 2018年には、72qubitのBristleCornを開発している。）

「なぜ、Googleの量子優越性のマイルストーンは重要なのか？」

アーロンソンは、一般の読者向けに、ニューヨーク・タイムス紙に、「なぜ、Googleの量子優越性のマイルストーンは重要なのか？」という記事を投稿している。こちらの記事も参照されたい。

"Why Google's Quantum Supremacy Milestone Matters" Scott Aaaronson, **2019/10/30**

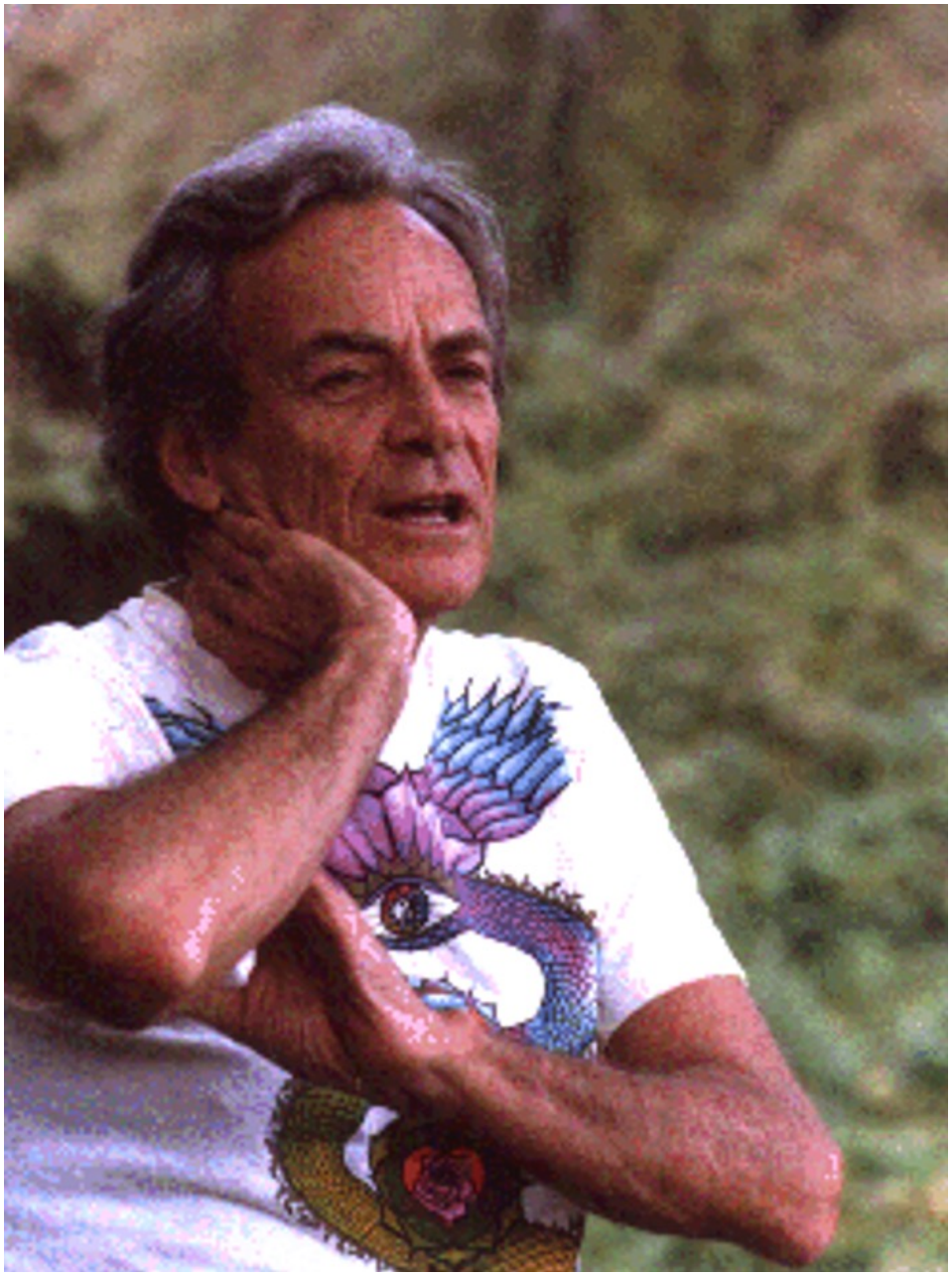
https://www.nytimes.com/2019/10/30/opinion/google-quantum-computer-sycamore.html


ファインマンの洞察
-- 自然をシミレートする量子コンピュータ --
量子計算の古典的検証
Classical Verification of Quantum Computation

“Simulating Physics with Computers”

Richard P. Feynman

1982年



Simulating Physics with Computers

1. Introduction
2. Simulating Time
3. Simulating Probability
4. Quantum Computers – Universal Quantum Simulators
5. Can Quantum Systems be Probabilistically Simulated by a Classical Computers?
6. Negative Probabilities
7. Polarization of Photons – Two State Systems
8. Two-Photon Correlation Experiment

https://s2.smu.edu/~mitch/class/5395/papers/feynman-quantum-1981.pdf

「物理をコンピュータでシミレートする」の構成

1. はじめに
2. 時間をシミレートする
3. 確率をシミレートする
4. 量子コンピュータ – 万能量子シミレータ
5. 量子システムは、古典的なコンピュータで確率論的にシミレートされることができるか？
6. 負の確率
7. 光子の偏極 – 二つの状態を取るシステム
8. 二つの光子の相関実験

物理をコンピュータでシミレートする

1. はじめに
2. 時間をシミレートする
3. 確率をシミレートする
4. 量子コンピュータ – 万能量子シミレータ
5. 量子システムは、古典的なコンピュータで確率論的にシミレートされることができるか？
6. 負の確率
7. 光子の偏極 – 二つの状態を取るシステム
8. 二つの光子の相関実験

物理は「万能コンピュータ」でシミレートできるか？

最初の問いは、物理学をシミレートするのに、どのようなコンピュータを、我々は利用しようとしているかと言うことである。

（コンピュータ科学が発達して、見かけは違っていても、それらは本質的に同じものであることを明らかにした）

それゆえ、私の問いは、物理は「万能コンピュータ」でシミレートできるかと言うことである。

（それは、たとえば、セルラー・オートマトンのようなものである。）

コンピュータは、正確に
自然と同じように振る舞うか？

コンピューターが、正確に自然と同じように振る舞う、正確なシミュレーションが存在する可能性について話そうと思う。

それが証明されて、そのコンピュータのタイプが先に説明したようなものであるなら、必然的に、有限の大きさの時空の中で起きる全てのものは、有限な数の論理的な操作で正確に分析可能でなければならないことになるだろう。

現在の物理学が、そうした考え方をしていないのは、明らかである。

現代の物理学は、空間が無限小の距離まで小さくなることを許すし、また、波長が無限大まで大きくなることを許す。また、計算項は、無限の次元まで和をとることが許される。等々。

それゆえ、先の有限の仮定が正しいとすれば、物理法則が間違っていることになる。

物理をコンピュータでシミレートする

1. はじめに
2. 時間をシミレートする
3. 確率をシミレートする
4. 量子コンピュータ – 万能量子シミレータ
5. 量子システムは、古典的なコンピュータで確率論的にシミレートされることができるか？
6. 負の確率
7. 光子の偏極 – 二つの状態を取るシステム
8. 二つの光子の相関実験

量子コンピュータ
-- 万能量子シミュレーター

量子の世界のシミレートは、新しいタイプのコンピューター、量子コンピューター？で可能になるだろう。

私が理解する限りでは、それは量子論的なシステムによって、量子コンピュータの要素によって、シミュレート出来るようになることは、いまや、明らかになった。

それはチューリング・マシンではない。別のタイプのマシンである。

物理をコンピュータでシミレートする

1. はじめに
2. 時間をシミレートする
3. 確率をシミレートする
4. 量子コンピュータ – 万能量子シミレータ
5. 量子システムは、古典的なコンピュータで確率論的にシミレートされることができるか？
6. 負の確率
7. 光子の偏極 – 二つの状態を取るシステム
8. 二つの光子の相関実験

量子論的システムは、古典的なコンピューターで確率論的にシミュレートされるか？

量子論的なシステムは、古典的な万能計算機で、確率論的にシミュレートされるだろうか？

別の言い方をすれば、コンピューターは、量子論的なシステムが行うのと、同じ確率を与えるだろうか？　コンピューターを今まで述べてきたような古典的なものだとすれば（前節で述べたような量子論的なものではないとすれば）、また法則はすべて変更されないままで、ごまかしもないとすれば、

答えは明らかにノーである。

量子力学の結果を、古典的な万能デバイスで表現することは不可能である。

量子論の方程式をできる限り古典論の方程式に近い形にしてみると、その難しさと何が起きるのかがわかる。変数の数が多すぎるのだ。

……

……

……

物理をコンピュータでシミレートする

1. はじめに
2. 時間をシミレートする
3. 確率をシミレートする
4. 量子コンピュータ – 万能量子シミレータ
5. 量子システムは、古典的なコンピュータで確率論的にシミレートされることができるか？
6. 負の確率
7. 光子の偏極 – 二つの状態を取るシステム
8. 二つの光子の相関実験


ベルの定理とその実験での検証
-- 証明と実験--
量子計算の古典的検証
Classical Verification of Quantum Computation

“On the Einstein Pdolsky Rosen Paradox”

John Bell

1964年

Physics Vol. 1, No. 3, pp. 195–200, 1964 Physics Publishing Co. Printed in the United States

ON THE EINSTEIN PODOLSKY ROSEN PARADOX*

J. S. BELL†
Department of Physics, University of Wisconsin, Madison, Wisconsin

(*Received* 4 *November* 1964)

I. Introduction

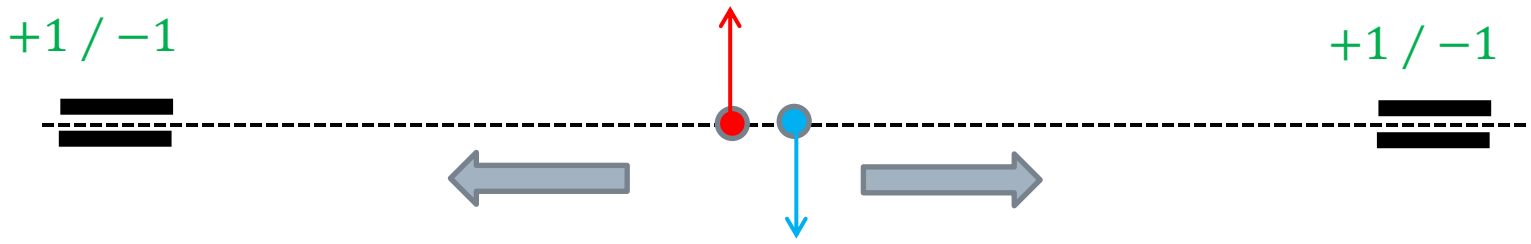
THE paradox of Einstein, Podolsky and Rosen [1] was advanced as an argument that quantum mechanics could not be a complete theory but should be supplemented by additional variables. These additional variables were to restore to the theory causality and locality [2]. In this note that idea will be formulated mathematically and shown to be incompatible with the statistical predictions of quantum mechanics. It is the requirement of locality, or more precisely that the result of a measurement on one system be unaffected by operations on a distant system with which it has interacted in the past, that creates the essential difficulty. There have been attempts [3] to show that even without such a separability or locality requirement no "hidden variable" interpretation of quantum mechanics is possible. These attempts have been examined elsewhere [4] and found wanting. Moreover, a hidden variable interpretation of elementary quantum theory [5] has been explicitly constructed. That particular interpretation has indeed a grossly nonlocal structure. This is characteristic, according to the result to be proved here, of any such theory which reproduces exactly the quantum mechanical predictions.

ベルの思考実験

実験の想定

上下反対のスピンを持つ量子を対発生させ、それぞれの粒子を別々に反対方向に打ち出す。それを、離れた場所にある観測機器で独立に観測する。

それぞれの観測での値+1（上向き）あるいは －1（下向き）を記録する。

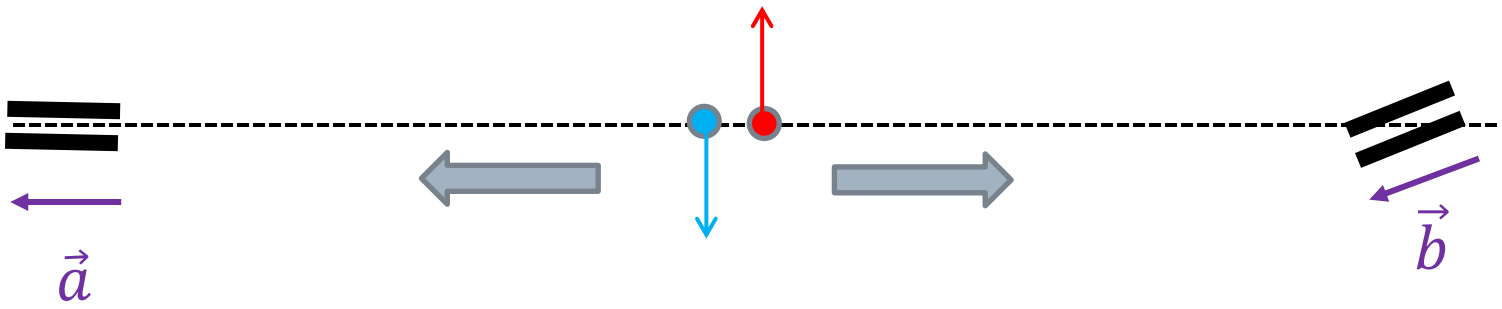


観測機器の配置

観測機器は、別々の方向を向いているとする。単位ベクトル$\vec{a}, \vec{b}$を取って、観測機器の配置を表すことにしよう。この時、設定$\vec{a}, \vec{b}$を取ったときの二つの観測機器間の出力の相関$P(\vec{a}, \vec{b})$は、次のように表すことができる。

$$P(\vec{a}, \vec{b}) = -\vec{a} \cdot \vec{b} = -\cos\theta$$

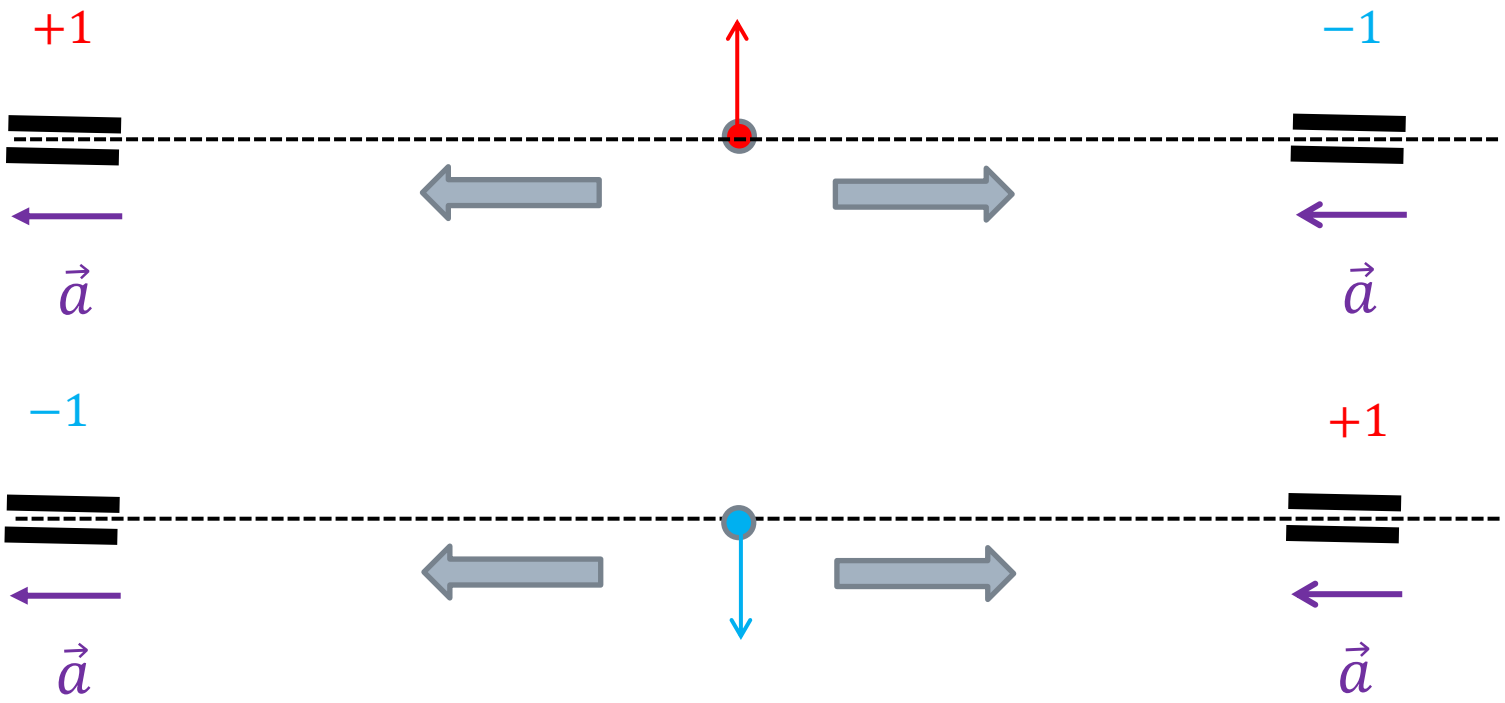
ここに θは、ベクトル$\vec{a}$と$\vec{b}$のなす角である



観測機器が平行に配置された場合

二つの機器が平行($\vec{a} = \vec{b}$)な場合には、それぞれの出力は他方の出力と反対のものになるので、その相関は

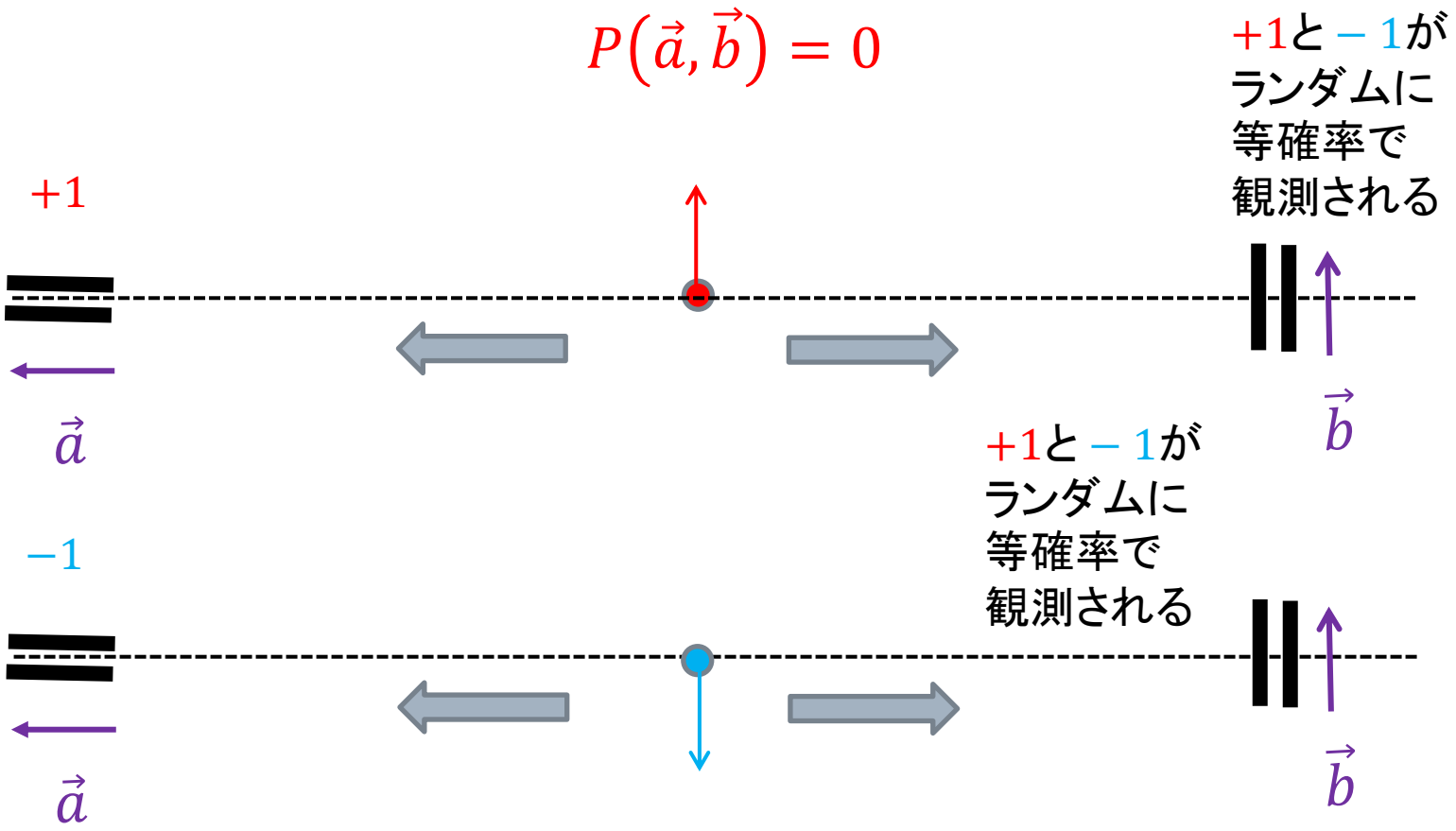
$$P(\vec{a}, \vec{a}) = -P(\vec{a}, -\vec{a}) = -1$$



観測機器が直交に配置された場合

二つの機器が直交する($\vec{a} \cdot \vec{b} = 0$)場合には、それぞれの出力には相関はない。

$$P(\vec{a}, \vec{b}) = 0$$



隠れた変数を仮定した場合の相関

隠れた変数をλとし、λの確率密度関数を$\rho(\lambda)$とする。

また、二つの観測機器A, Bのベクトル$\vec{a}, \vec{b}$と隠れた変数λのもとでの、観測機器の出力を、$A(\vec{a}, \lambda), B(\vec{b}, \lambda)$とする。

$$A(\vec{a}, \lambda) = \pm 1, B(\vec{b}, \lambda) = \pm 1$$

である。

この時、可能なすべてのλについて積分して、次の式が成り立つ。

$$P(\vec{a}, \vec{b}) = \int d\lambda \, \rho(\lambda) A(\vec{a}, \lambda) B(\vec{b}, \lambda)$$

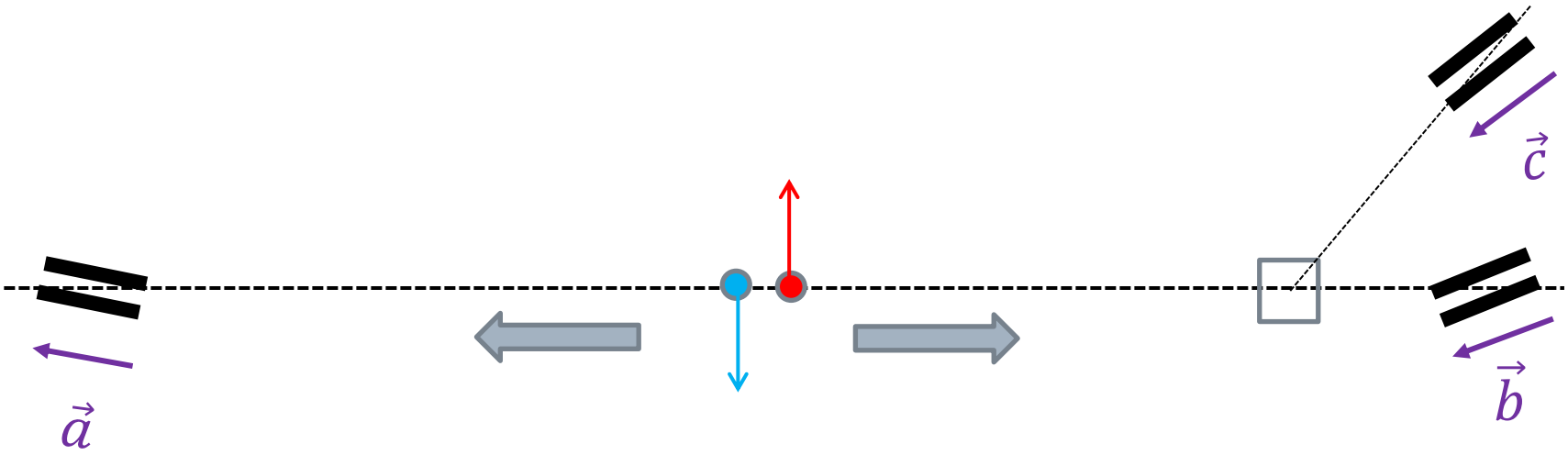
重要なことは、AとBは物理的に分離しているので、Aの出力は$\vec{b}$に依存せず、同様に、Bの出力は$\vec{a}$に依存しない。

隠れた変数の仮定のもとでの不等式
ベルの不等式

第二の観測機器の配置が$\vec{b}$でなく$\vec{c}$である実験も考えよう。

この時、Bellは、古典論の隠れた変数の仮定のもとで、次の不等式が成り立つことを示す。

$$\left|P(\vec{a},\vec{b})-P(\vec{a},\vec{c})\right| \leq 1+P(\vec{b},\vec{c})$$



量子論のもとでの相関

しかし、量子論のもとでは、上の不等式は成り立たない。

例えば、$\vec{a}$と$\vec{b}$は直交し、$\vec{c}$は$\vec{a}$と$\vec{b}$いずれとも、45度の角度をなしているとしよう。

この時、

$$P(\vec{a},\vec{b}) = 0$$

$$P(\vec{a},\vec{c}) = P(\vec{b},\vec{c}) = -\frac{\sqrt{2}}{2}$$

である。

この値を、先の式

$$\left|P(\vec{a},\vec{b}) - P(\vec{a},\vec{c})\right| \leq 1 + P(\vec{b},\vec{c})$$

に代入してみる。

古典論の不等式を量子論は破る

$$\left|0+\frac{\sqrt{2}}{2}\right| \leq 1-\frac{\sqrt{2}}{2}$$

$$\frac{\sqrt{2}}{2} \leq 1-\frac{\sqrt{2}}{2}$$

これは、矛盾している。

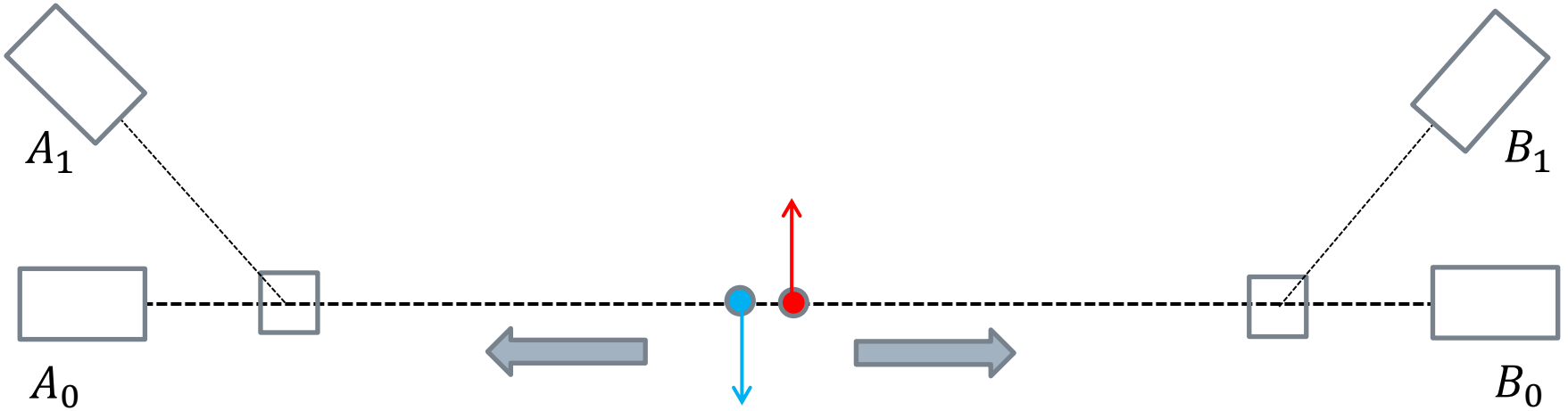
観測機器の可能なすべての配置 $\vec{a}$と$\vec{b}$と$\vec{c}$ の量子論的相関の中には、古典論的相関を破るものが存在する。

アスペの実験

-- CHSHの定式化に基づく --

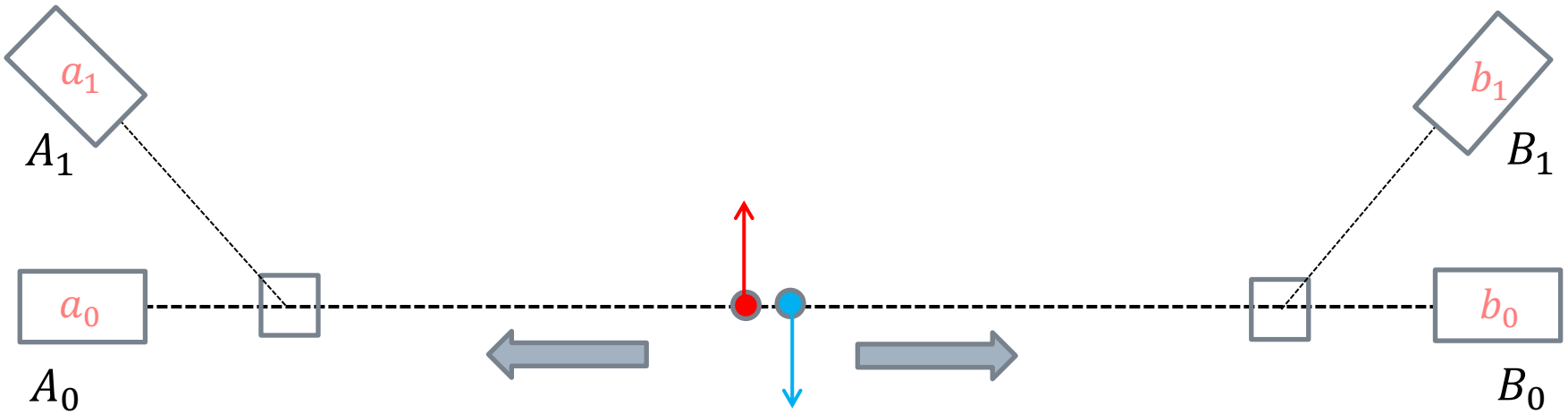
Aspectの実験

Aspectは、次のように観測機器 A_0, A_1, B_0, B_1を配置する。
この実験のもとになったのは、CHSHによる、Bellの定理の定式化である。



Aspectの実験

それぞれの観測機器が観測した値を $a_0, a_1, b_0. b_1$とする。
$a_0 = \pm 1, a_1 = \pm 1,\ b_0 = \pm 1. b_1 = \pm 1$ である。



CHSHによるBellの不等式

先の観測値 $a_0, a_1, b_0. b_1$ について、次の式Cを考える。

$$C = a_0 b_0 + a_0 b_1 + a_1 b_0 - a_1 b_1$$

この式を、次のように変形する。

$$C = (a_0 + a_1) b_0 + (a_0 - a_1) b_1$$

$a_0 = \pm 1, a_1 = \pm 1$から、$a_0 = a_1$ あるいは $a_0 = -a_1$ である。

もし、$a_0 = a_1$ なら、$(a_0 - a_1) b_1 = 0$ となり、

$$C = (a_0 + a_1) b_0 + 0 = 2 a_0 b_0 = \pm 2$$

もし、$a_0 = -a_1$ なら、$(a_0 + a_1) b_0 = 0$ となり、

$$C = 0 + (a_0 - a_1) b_1 = 2 a_0 b_1 = \pm 2$$

観測機器A_i, B_jで繰り返し観測した結果の平均を$< A_i B_j >$で表すと

$$< A_0 B_0 > + < A_0 B_1 > + < A_1 B_0 > - < A_1 B_1 > \leq 2$$

これをBell不等式、あるいは CHSH不等式という。

ここでの観測の前提
局所実在論

ここでの観測には、次のような前提がある。

- 【実在論】観測された物理的性質 $a_0, a_1, b_0. b_1$ は、観測とは独立に存在している。

- 【局所性】Aでの観測はBの観測に影響を与えないし、Bでの観測はAでの観測に影響を与えない。

量子論での計算

量子論での観測

上下のスピンを持つエンタングルメント状態は、

$$|\psi_{\pm}> = \frac{1}{\sqrt{2}}(|01> \pm |10>)$$

と表すことができる。

|00>,|01>,|10>,|11>を基底として

$$|\psi_{-}> = \frac{1}{\sqrt{2}}\begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

としよう。

A,BのObservable

パウリのスピン行列σ_x, σ_zは、次のもの。

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Aの側は、$A_0 = \sigma_z, A_1 = \sigma_x$ をobservableとし観測する。

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, A_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

これらのobservableの固有ベクトルは|0>.|1>なので、標準基底で観測するということ。

Bの側は、$B_0 = -\frac{\sigma_x+\sigma_z}{\sqrt{2}}, B_1 = \frac{\sigma_x-\sigma_z}{\sqrt{2}}$ をobservableとし観測する。

$$B_0 = -\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, B_1 = -\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix}$$

これらのobservableの固有ベクトルは|+>.|->なので、アダマール基底で観測するということ。

Observableの観測の平均値

Observable Oが与えられた時、状態$|\phi>$の観測値の平均は、

$$<\phi|O|\phi>$$

で与えられる。

先の$|\psi>, A_0. A_1, B_0, B_1$について、次の値を計算してみよう。

$$<\psi|A_0 \otimes B_0|\psi>$$
$$<\psi|A_0 \otimes B_1|\psi>$$
$$<\psi|A_1 \otimes B_0|\psi>$$
$$<\psi|A_1 \otimes B_1|\psi>$$

$A_0 \otimes B_0$

$$A_0 \otimes B_0 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 1\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 0\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\ 0\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & -1\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

$< \psi | A_0 \otimes B_0 | \psi >$

$$< \psi | A_0 \otimes B_0 | \psi >$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= \frac{1}{2} \cdot -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= -\frac{1}{2\sqrt{2}} (0 + (-1) + (-1) + 0) = \frac{1}{\sqrt{2}} = \frac{\sqrt{2}}{2}$$

$A_0 \otimes B_1$

$$A_0 \otimes B_1 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 1\begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} & 0\begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} \\ 0\begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} & -1\begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

$< \psi | A_0 \otimes B_1 | \psi >$

$$< \psi | A_0 \otimes B_1 | \psi >$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= \frac{1}{2} \cdot -\frac{1}{\sqrt{2}} \begin{pmatrix} -1 & -1 & 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= -\frac{1}{2\sqrt{2}} (0 - 1 - 1 + 0) = \frac{1}{\sqrt{2}} = \frac{\sqrt{2}}{2}$$

$A_1 \otimes B_0$

$$A_1 \otimes B_0 = -\frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 0\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 1\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\ 1\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 0\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \\ 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{pmatrix}$$

$< \psi | A_1 \otimes B_0 | \psi >$

$$< \psi | A_1 \otimes B_0 | \psi >$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \\ 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= -\frac{1}{2\sqrt{2}} \begin{pmatrix} -1 & -1 & 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= -\frac{1}{2\sqrt{2}} (0 + (-1) + (-1) + 0) = \frac{1}{\sqrt{2}} = \frac{\sqrt{2}}{2}$$

$A_1 \otimes B_1$

$$A_1 \otimes B_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 0 \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} & 1 \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} \\ 1 \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} & 0 \begin{pmatrix} 1 & -1 \\ -1 & -1 \end{pmatrix} \end{pmatrix}$$

$$= -\frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & -1 \\ 1 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 \end{pmatrix}$$

$< \psi | A_1 \otimes B_1 | \psi >$

$$< \psi | A_1 \otimes B_1 | \psi >$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & -1 \\ 1 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= -\frac{1}{2\sqrt{2}} \begin{pmatrix} -1 & 1 & -1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$= -\frac{1}{2\sqrt{2}} (0 + 1 + 1 + 0) = -\frac{1}{\sqrt{2}} = -\frac{\sqrt{2}}{2}$$

量子論での観測値

次の式で$< A_iB_j >$は$< A_i \otimes B_j >$を表しているとする。

$$< A_0B_0 > + < A_0B_1 > + < A_1B_0 > - < A_1B_1 >$$
$$= \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} = 2\sqrt{2}$$

量子論の観測では、

$$< A_0B_0 > + < A_0B_1 > + < A_1B_0 > - < A_1B_1 > = 2\sqrt{2}$$

これは、古典論での$Bell$の不等式

$$< A_0B_0 > + < A_0B_1 > + < A_1B_0 > - < A_1B_1 > \leq 2$$

を破っている。

ここでの$2\sqrt{2}$を$Tsirelson\ bound$と呼ぶ。


Interactive Proofと証明概念の転換
-- 証明と検証 --
量子計算の古典的検証
Classical Verification of Quantum Computation

Interactive Proofの登場

1990年代に入って、計算複雑性理論は、大きく発展する。

その転機となったのは、László Babai, Shafi Goldwasser, Silvio Micali, Shlomo Moran, Charles Rackoff らによるInteractive Proof Systemである。

Interactive Proofの登場は、計算複雑性理論を一変させた。

Interactive Proofの登場と発展

彼ら（彼女）らは、「Interactive Proof Systemの発明」によって、1993年、新設された「ゲーデル賞」の最初の受賞者となる。

また、2001年の「PCP定理」、2019年の「PCP定理の新証明」にも「ゲーデル賞」が与えられている。

Interactive Proofの登場と発展

彼ら（彼女）らは、「Interactive Proof Systemの発明」によって、1993年、新設された「ゲーデル賞」の最初の受賞者となる。

また、2001年の「PCP定理」、2019年の「PCP定理の新証明」にも「ゲーデル賞」が与えられている。

こうしたInteractive Proofに基づく計算複雑性研究の発展の、最も重要な達成が2020年の「MIP*＝RE定理」である。

証明概念を転換した
ゲーデル賞の女性たち

Gödel Prize　1993, 2001

Interactive Proof

Gödel Prize　2019

PCP定理の新証明

重要なことが一つある。

重要なことが一つある。

それは、Interactive Proofの登場によって、証明の概念が大きく変わったということである。

重要なことが一つある。

それは、Interactive Proofの登場によって、証明の概念が大きく変わったということである。

ここでは、そのことを見ておこう。

主要に、証明を行う側から考えた、「証明」観

これまでも、数学的証明の特質については、様々な議論があった。

証明が従うべき演繹規則、その出発点としての公理群、証明体系の無矛盾性 … 等々、

ただ、それは、主要に証明を行う側から証明を考えたものだった。

「証明者」と「検証者」の分離と両者の「対話」としての証明

Interactive Proofの特徴は、「証明者」と「検証者」を分離し、証明を「証明者」と「検証者」の両者の「対話」の過程として捉え返すことである。

証明を、検証の側から捉え直す

それはまた、証明を証明の世界に閉じたものと考えることをやめて、証明を検証との関係で、検証の側から捉え返すことでもあった。

証明は、なぜ正しいのか？

証明は、もともとが正しい推論によって導かれているから、正しいと検証されるのだと考えることと、正しいと検証されるから正しい証明だと考えることは別のことである。

検証されたものが、正しい証明である

証明は、もともとが正しい推論によって導かれているから、正しいと検証されるのだと考えることと、正しいと検証されるから正しい証明だと考えることは別のことである。

Interactive Proofのアプローチは、証明の正しさについて後者の考え方をとる。正しいと検証されたものが正しい証明なのである。

演繹のルールは、決定論的か？

「正しいものから正しいものを演繹する」という証明観では、証明の世界と確率の世界を結びつけるのは難しい。演繹のルールは確率論的性質を持たず、厳密に決定論的に振る舞うからである。

検証は、確率論的な性質を持つ

ただ、検証の世界は、確率論的な性質を持つ。

証明者が、正しい証明を持っていない場合には、検証者に返す答がランダムなものになるのは明らかだし、正しい証明を持っている場合には、証明者が提示するサンプルの全てを検証する必要はない。

「確率的に検証可能な証明」というコンセプト

ただ、検証の世界は、確率論的な性質を持つ。

証明者が、正しい証明を持っていない場合には、検証者に返す答がランダムなものになるのは明らかだし、正しい証明を持っている場合には、証明者が提示するサンプルの全てを検証する必要はない。

検証から証明を捉え返した時、証明の世界と確率の世界は結びつき、「確率的に検証可能な証明」という重要なコンセプトが現れてくる。

こうした、「証明観」の大きな転換は、
証明が可能とするものの領域を
大きく広げることとなった。


複雑性の新しいクラスBQPの発見
-- ショアのアルゴリズムと量子計算複雑性 --
量子計算の古典的検証
Classical Verification of Quantum Computation

計算複雑性の新しいクラス BQPの発見

Shorのアルゴリズムの発見を
Shor自身はどう考えていたか？

しかし、コンピュータサイエンスの歴史において、最も重要な問題は、多項式時間またはNP完全問題のいずれかであることがわかっています。したがって、量子コンピュータは、NP完全問題を解くことができない限り、おそらくそれほど広くは役に立たないでしょう。

Shorのアルゴリズムの発見を Shor自身はどう考えていたか？

NP完全問題を効率的に解くことは、理論的コンピュータ科学の「聖杯」です。それが、古典的なコンピュータ上で可能であると期待するのは、非常に少数の人々だけです。量子コンピュータ上で、これらの問題を解決するための多項式時間アルゴリズムを見つけることは、画期的な発見となるでしょう。

量子コンピュータはNP完全問題を解くのに十分に強力ではないといういくつかの弱い徴候があります[Bennett et al。1994]。

しかし、私はこの可能性がまだ排除されるべきであるとは思いません。

Shor論文のEnding Remark

BQPクラスの発見

Umesh Vazirani

“Quantum Complexity Theory”
Bernstein ,Vazirani
1993年
https://goo.gl/3y4RSf

新しい量子複雑性のクラスBQPの発見

本論文では、シミュレーションのオーバーヘッドが多項式で制限される万能量子チューリングマシンの存在を証明する。

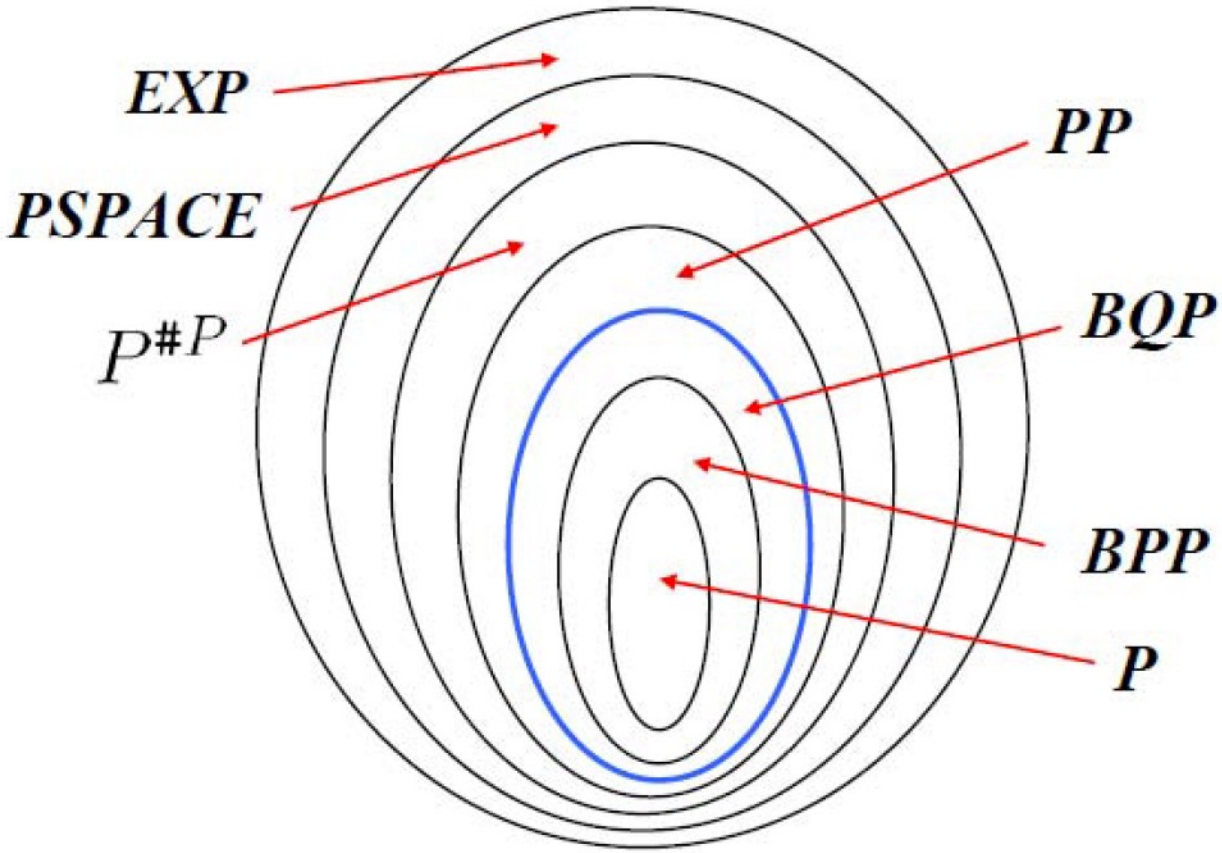
本論文では、量子チューリングマシンが古典的な確率的チューリングマシンよりも強力である可能性があるという最初の証拠を提示する。

P ：古典的コンピュータで、多項式時間で計算可能なクラス

BQP: 量子コンピュータで、多項式時間で計算可能なクラス

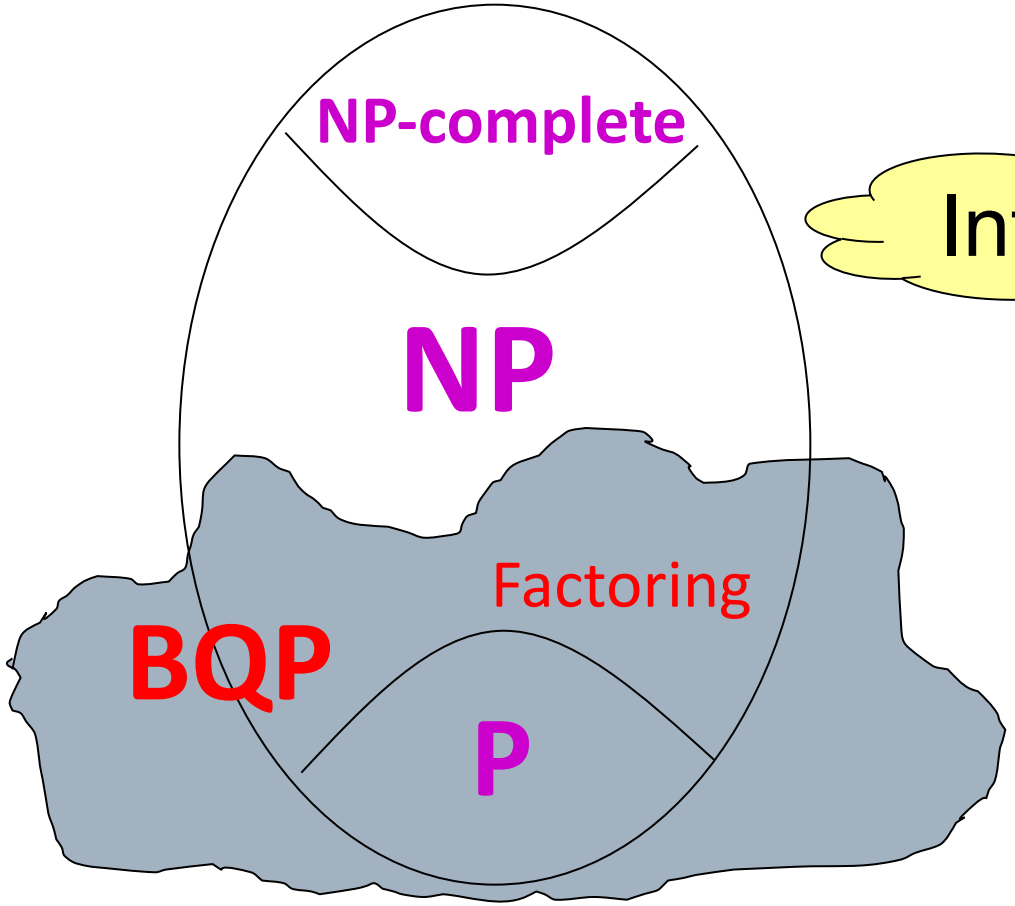
$$\mathbf{P} \subseteq \mathbf{BQP}$$

Basic properties of BQP



BQP (Bounded-Error Quantum Polynomial-Time): The class of problems solvable efficiently by a quantum computer, defined by Bernstein and Vazirani in 1993

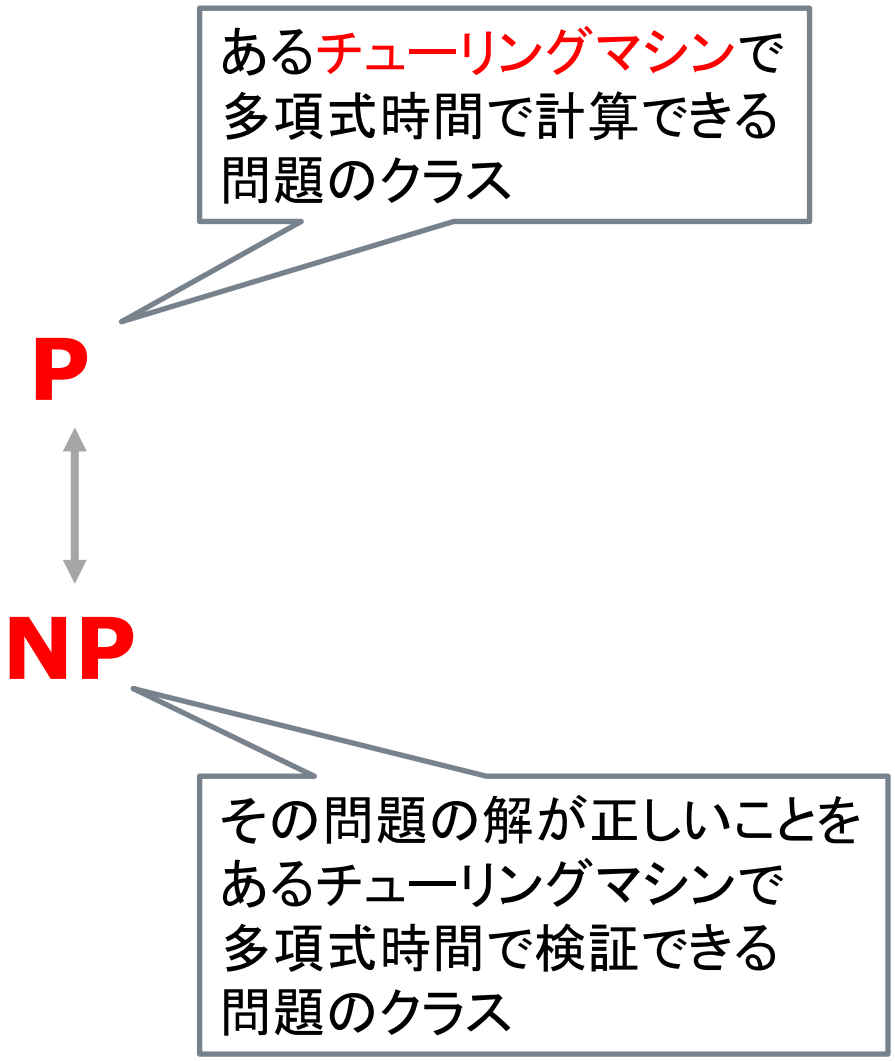
Shor 1994: Factoring integers is in **BQP**



基本的な複雑性のクラス

-- 古典複雑性と量子複雑性 --

古典複雑性：PクラスとNPクラス



量子複雑性：BQPクラスとQMAクラス

ある量子チューリングマシンで
多項式時間で計算できる問題
のクラス

BQP

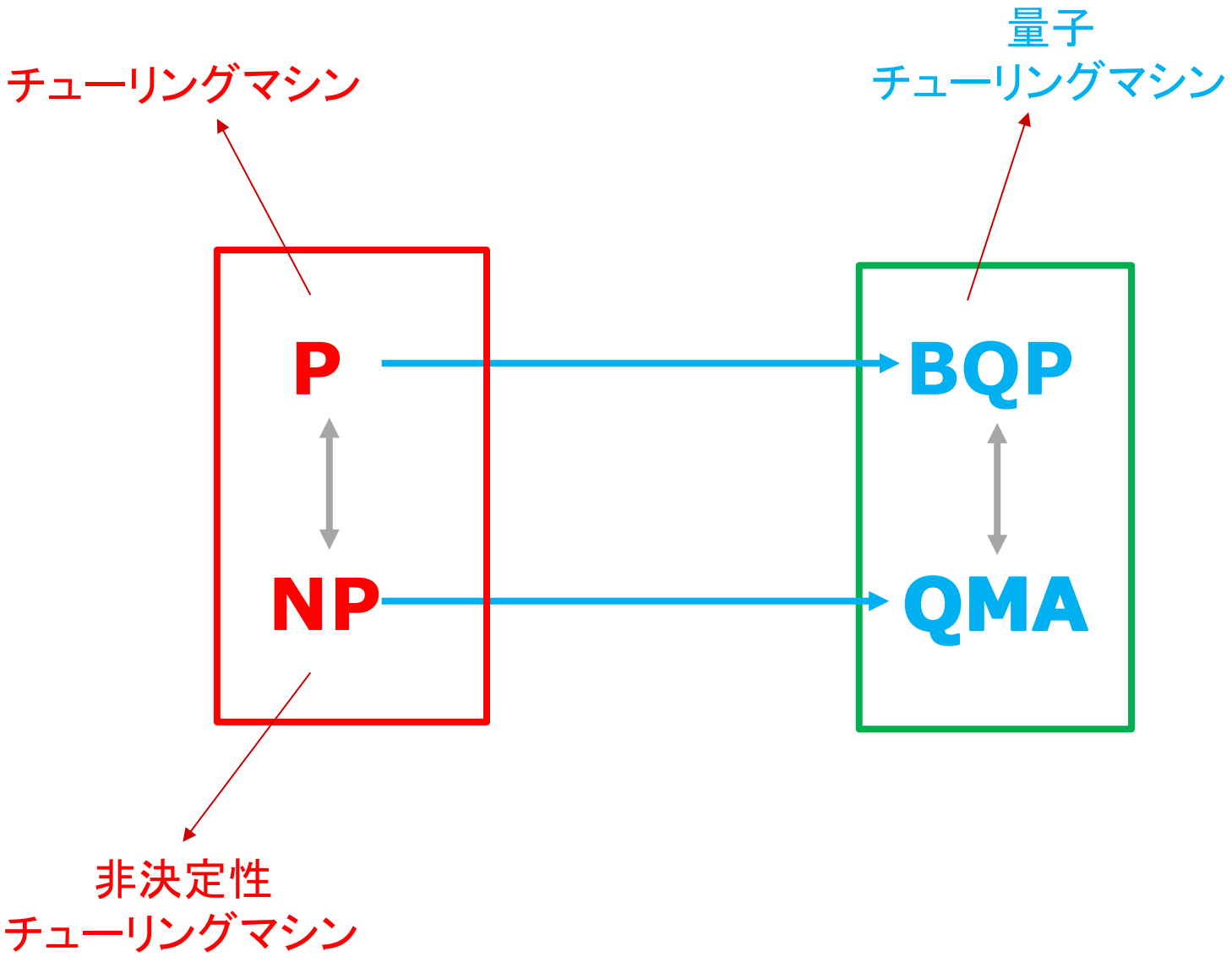
BQP：Bounded-Error Quantum
Polynomial-Time

QMA：Quantum MA

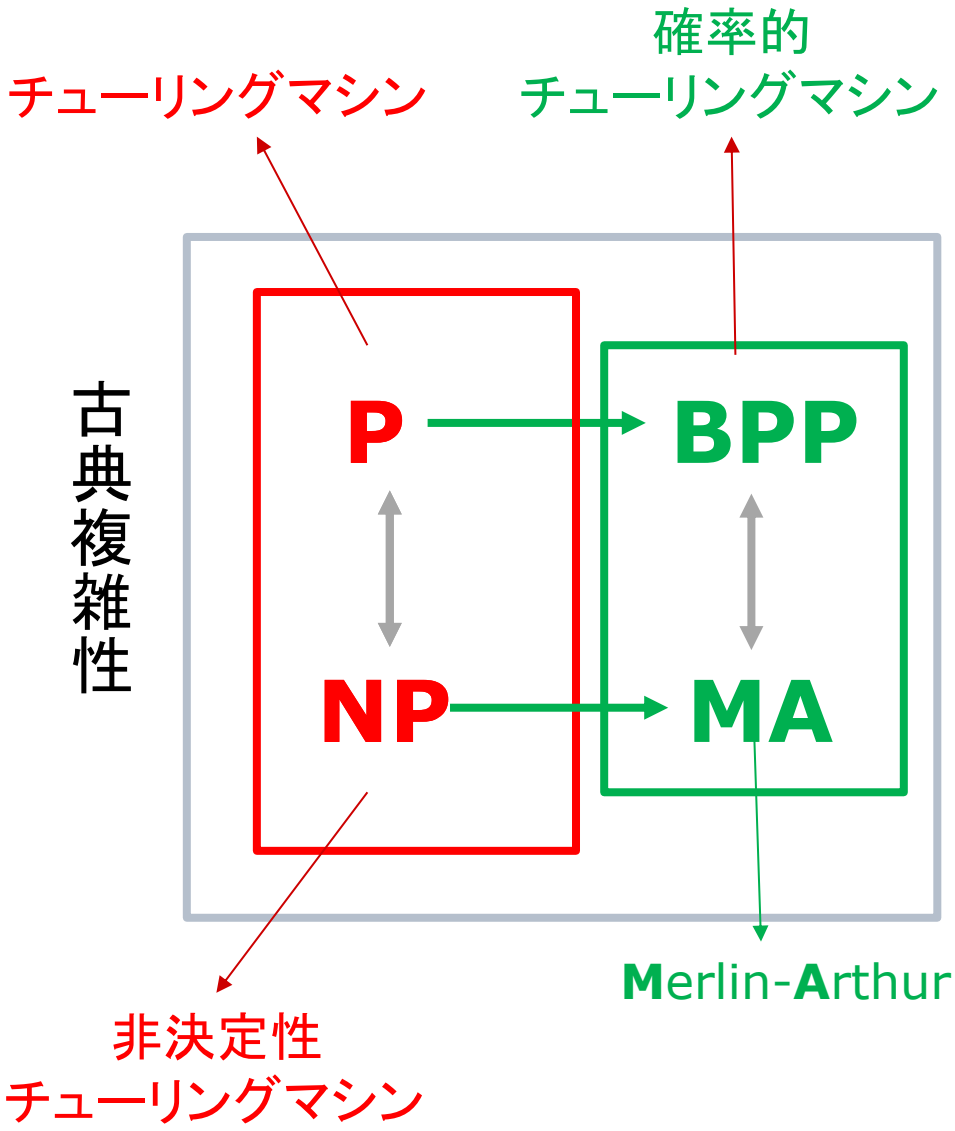
QMA

その問題の解が正しいことを
ある量子チューリングマシンで
多項式時間で検証できる
問題のクラス

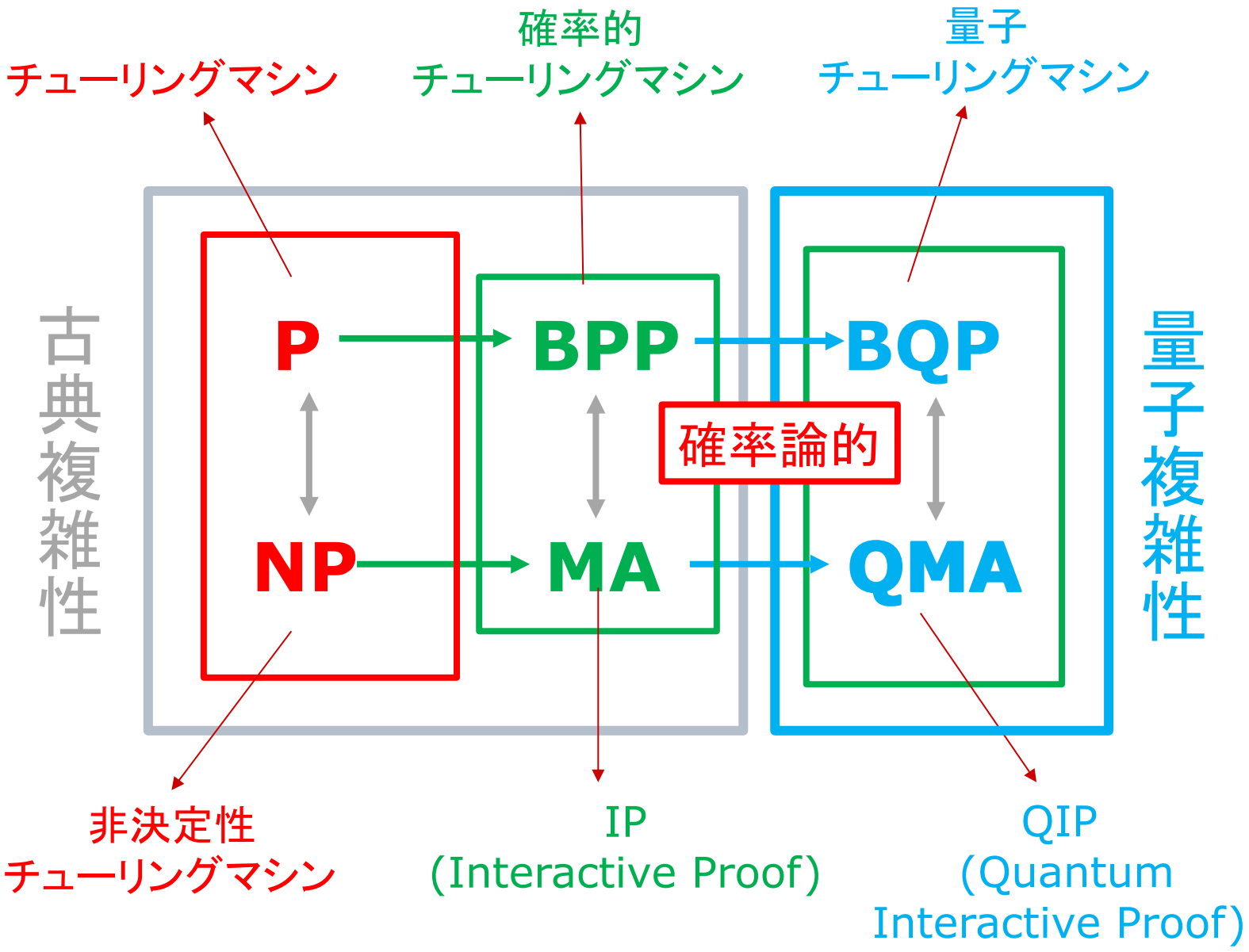
チューリングマシンと量子チューリングマシン



確率的古典複雑性


確率的
チューリングマシン
チューリングマシン
古典複雑性
P
BPP
NP
MA
Merlin-Arthur
非決定性
チューリングマシン

基本的な複雑性のクラス




Part III
準備

Agenda Part III

準備

- Learning with Errors
- Universal Quantum Circuit
- Trapdoor Claw-Free Functions


Learning with Errors
-- Post量子暗号 --
量子計算の古典的検証
Classical Verification of Quantum Computation

Oded Regev

"On lattices, learning with errors, random linear codes, and cryptography"

2005

https://dl.acm.org/doi/10.1145/1060590.1060603

Gödel Prize (2018)

https://en.wikipedia.org/wiki/Oded_Regev_(computer_scientist)

Alice：秘密キーsとエラーキーeの作成

- Aliceは、n個の$\mathbb{Z}_q$の要素をランダムに集めて、秘密キーsを作成する。

$$s \leftarrow \mathbb{Z}_q^n$$

- Aliceは、m個の$\mathbb{Z}_q$の小さな要素をランダムに集めて、エラー・キーeを作成する。

$$e \leftarrow \mathbb{Z}_q^m$$

Alice：公開キー A, B の作成

- Aliceは、$n \times m$個の$\mathbb{Z}_q$の要素をランダムに集めて、n行m列の行列Aを作成し、公開キーAとする。

$$A \leftarrow \mathbb{Z}_q^{n \times m}$$

- Aliceは、公開キーAと秘密キーsとエラー・キー eから、次の式で公開キーBを作成する。

$$B = A^T s + e$$

$$B^T = s^T A + e^T$$

Aliceの仕事

秘密キー

$s \leftarrow \mathbb{Z}_q^n$

$e \leftarrow \mathbb{Z}_q^m$

$A \leftarrow \mathbb{Z}_q^{n \times m}$

$B = A^T s + e$

公開キー

Alice

Bob

Bob：公開キーAからのサンプリング

- Bobは、サンプリング用のベクトル $x \in \{0,1\}^m$ を作成する。
  1が立っているところがピックアップされる。

$$x \leftarrow \{0,1\}^m$$

- Bobは、公開キーAとベクトルxを掛けて、サンプリング・データ u を作成する。

$$u = Ax$$

$u = Ax$

$$u = Ax$$

行列Aに列ベクトルx を掛けたものが、列ベクトルu。
$A \in \mathbb{Z}_q^{n \times m}$, $x \in \{0,1\}^m$ とすれば、次のような形の計算になる。

$$\mathrm{u} = \underbrace{\overbrace{A^{n \times m}}^{\mathrm{n}}}_{\mathrm{m}} \times \begin{pmatrix} 0/1_1 \\ 0/1_2 \\ \vdots \\ 0/1_m \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix}$$

$$u_i = \sum A_{ik}$$

Bob: $bit \in \{0,1\}$ のエンコードと エンコード・データ (u, v)の送信

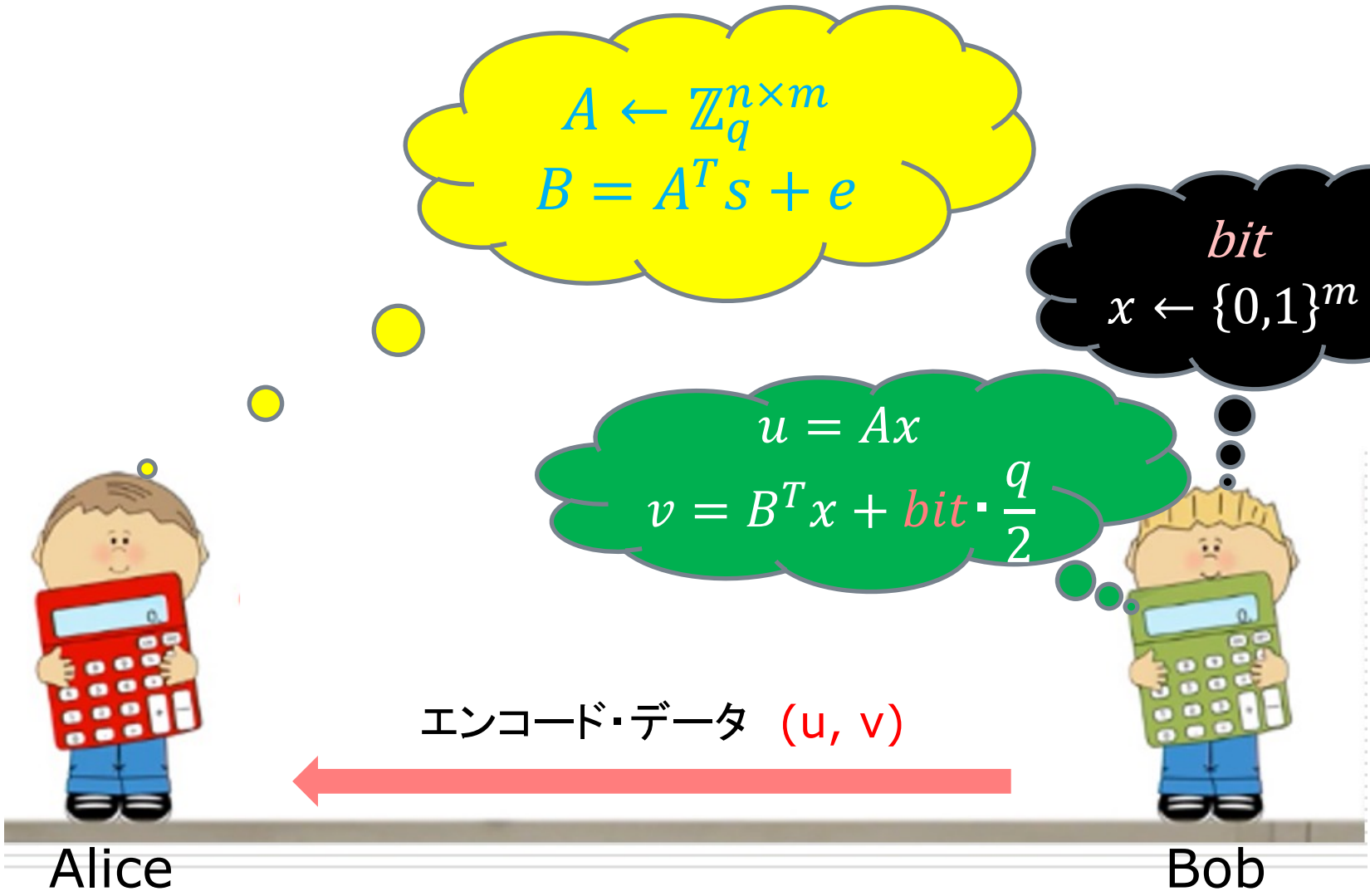
- Bobは、公開キーBとサンプリング・ベクターx から、$bit \in \{0,1\}$のエンコード・データ v を作成する。

$$v = B^T x + bit \cdot \frac{q}{2}$$

$$v = B^T x + bit \cdot \frac{q}{2}$$

- Bobは、エンコード・データ (u, v) を Aliceに送信する。

Bobの仕事


$A \leftarrow \mathbb{Z}_q^{n \times m}$
$B = A^T s + e$
bit
$x \leftarrow \{0,1\}^m$
$u = Ax$
$v = B^T x + bit \cdot \frac{q}{2}$
エンコード・データ (u, v)
Alice
Bob

Aliceの デコード

- Aliceは、受け取った(u,v)から、次の値を計算する

$$Dec = v - s^T u \ \ (mod\ q)$$

- Aliceは、Decから、次の式で bit をデコードする。

$$Dec < \frac{q}{2} \rightarrow \ \ bit = 0$$

$$Dec \geq \frac{q}{2} \rightarrow \ \ bit = 1$$

デコード・ルールとエラー項

デコードのルールとエラー項の役割

最後に見た 次のデコードのルールが分かりにくかったかと思う。

$$Dec < \frac{q}{2} \rightarrow \quad bit = 0$$
$$Dec \geq \frac{q}{2} \rightarrow \quad bit = 1$$

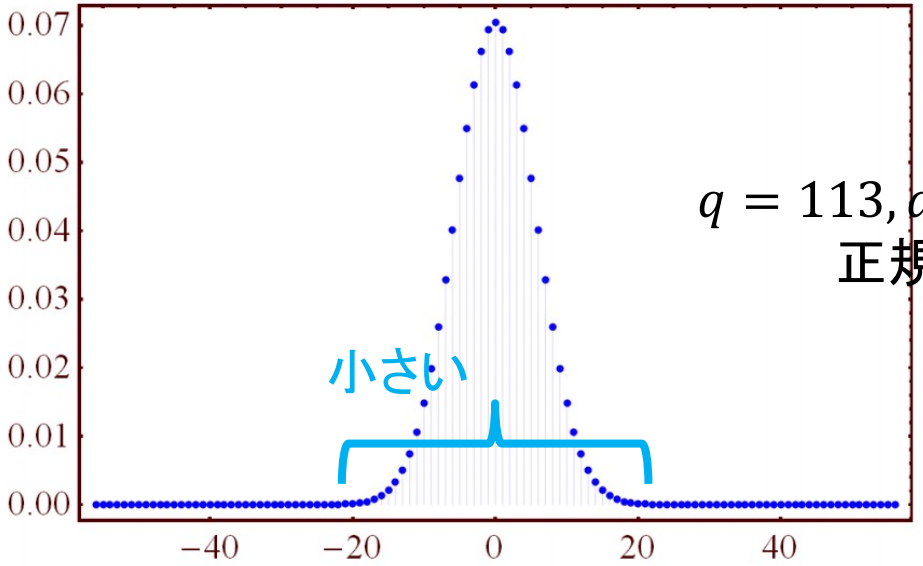
実は、これまできちんと説明してこなかったことがある。

それは、「小さな数をランダムに選ぶ」としてきたエラー項e がどのように選ばれ、LWEの中でどのような役割を果たしているかと言うことである。

エラー項は、正規分布に従うものとして選ばれる

エラー項eは、平均値 0 で標準偏差がαqである正規分布から、選ばれる。$e \in \mathbb{Z}_q^m$ であるので、正規分布と言ってもディスクリートな格子点上に選ばれる確率が定義されている。

エラー項としては、0が選ばれる確率が最も高い。平均値の0から左右に離れるとその値が選ばれる確率は急速に0に近づく。「小さな値」が選ばれると言うのは、そう言うことである。

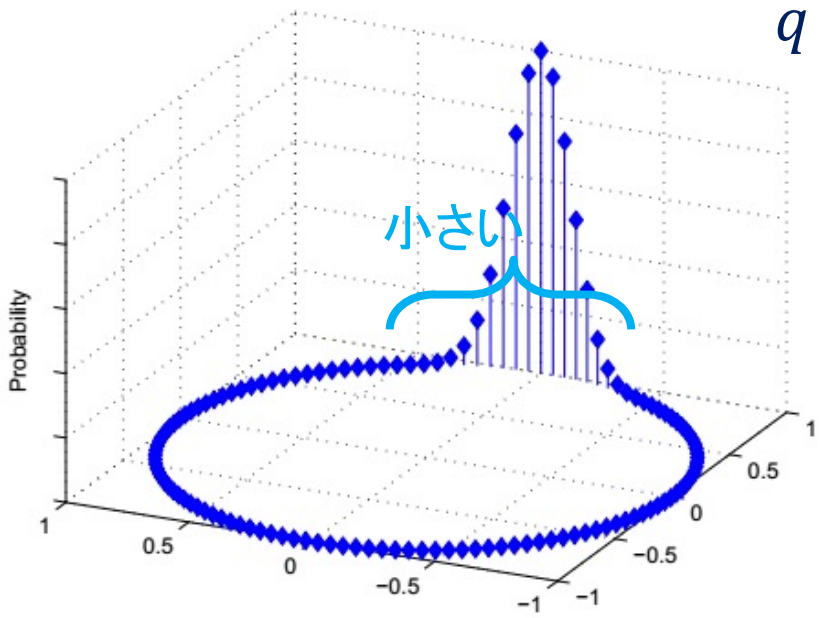


$q = 113, \alpha = 0.05$ の正規分布

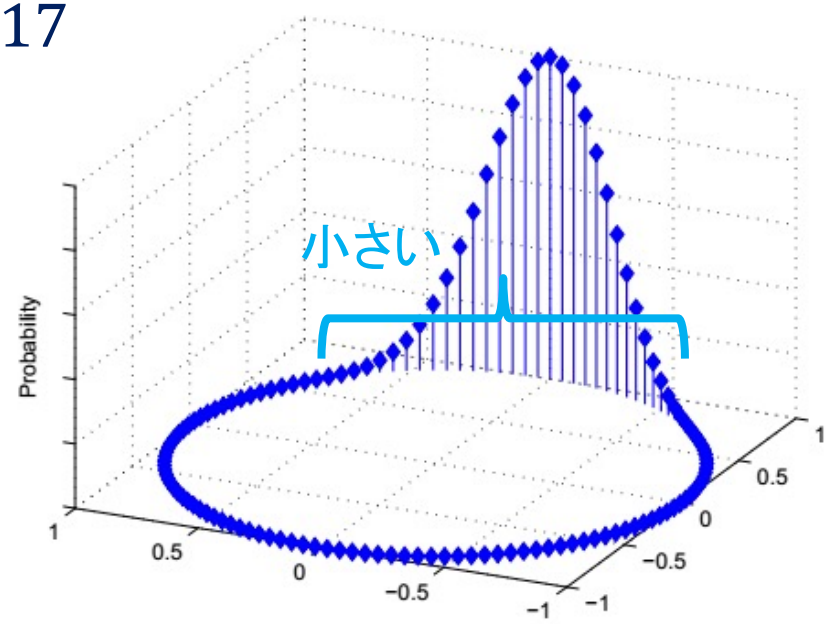
エラー項の分布

エラー項の要素は $\mathbb{Z}_q$なので巡回する円で表せる
その時のエラー項の正規分布は次のようになる

$q = 117$



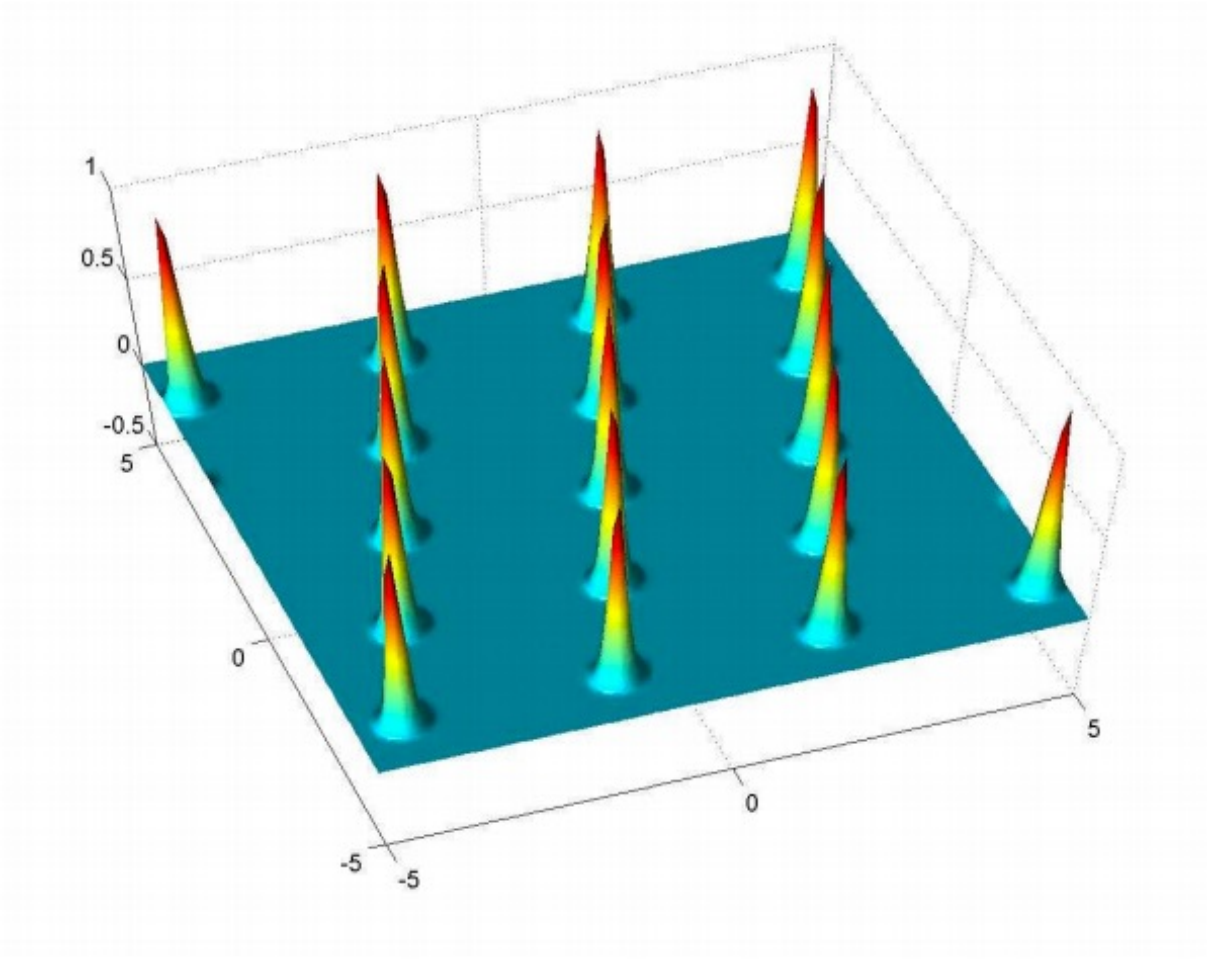
$\alpha = 0.05$



$\alpha = 0.1$

エラー項の分布
二次元の場合

二次元ラティス上の正規分布



エンコードされた(u, v)を振り返る

ベクトル $u \in \mathbb{Z}_q^n$ は、公開キー$A \in \mathbb{Z}_q^{n \times m}$と
サンプリング用のベクトル $x \in \{0,1\}^m$を掛けたもの

$$u = Ax$$

スカラー vは、公開キー $B \in \mathbb{Z}_q^{m \times 1}$のトランスポーズ $B^T \in \mathbb{Z}_q^{1 \times m}$と
サンプリング用のベクトル $x \in \{0,1\}^m$を掛けたものに
数 $bit \cdot \frac{q}{2}$ を加えたもの

$$v = B^T x + bit \cdot \frac{q}{2}$$

$Dec = v - s^T u$ を計算する

$B = A^T s + e$ から、$B^T = s^T A + e^T$

左からxを掛けて、$B^T \mathrm{x} = s^T A \mathrm{x} + e^T \mathrm{x}$ 、

$u = Ax$ だから次の式が成り立つ。

$$B^T \mathrm{x} - s^T u = e^T \mathrm{x}$$

$v = B^T x + bit \cdot \frac{q}{2}$ だから

$$Dec = v - s^T u = \left(B^T x + bit \cdot \frac{q}{2}\right) - s^T u = e^T x + bit \cdot \frac{q}{2}$$

$$(v - s^T u) - bit \cdot \frac{q}{2} = e^T x = \sum e_k \approx 0$$

$$(v - s^T u) \approx bit \cdot \frac{q}{2}$$

eからk個の要素をサンプリングして和をとったもの

デコードのルール

$Dec = v - s^T u$ を $v - < s, u >$と内積の形で書こう。どちらの項もスカラーであることがわかりやすい。

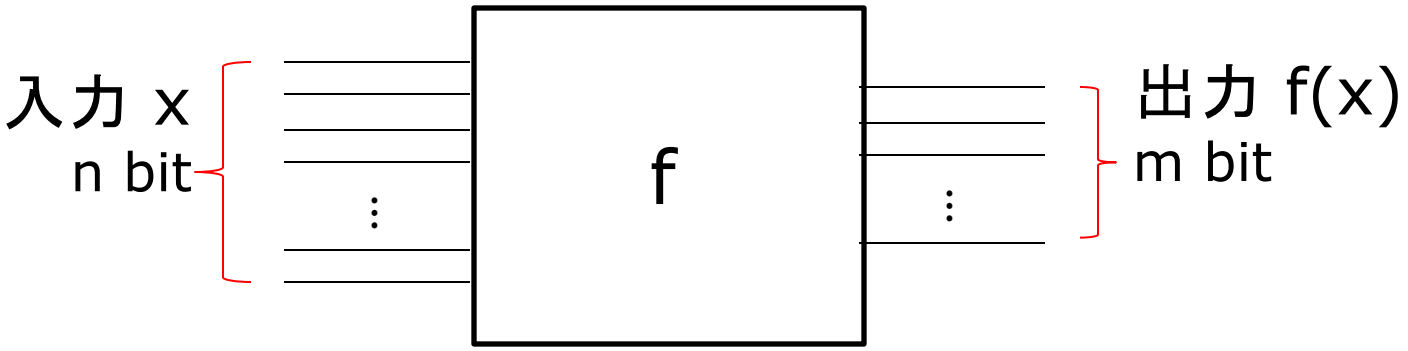
- $v - < s, u >$が $\left\lfloor \frac{q}{2} \right\rfloor$ より、0に近ければ、$bit = 0$
- そうでなければ、 $bit = 1$


Universal Quantum Circuit
-- 任意のf(x)を計算する量子回路を考える --
量子計算の古典的検証
Classical Verification of Quantum Computation

量子回路の一般的な形を考える

量子回路の一般的な形を考える

関数 f を計算する古典的な回路を考えよう。入力 x のビット数 n と、出力 f(x)のビット数 m とは一般に独立である。



例えば、XORゲートでは、入力のビット数 nは2で、出力のビット数 mは1である。

量子回路のユニタリ性

関数 f を計算する量子的なアルゴリズでも、入力 x のqubit数 n と、出力 f(x)のqubit数 m とは一般に独立である。

それでは、関数 f を計算する量子回路は、先の古典回路と同じような形を取るのだろうか？　そうはならない。

量子ゲートはユニタリ変換を行うゲートなのだが、量子ゲートから構成される量子回路も、それ自体がユニタリ性を持たねばならない。量子回路全体に対応するユニタリ行列は、巨大なものになる。

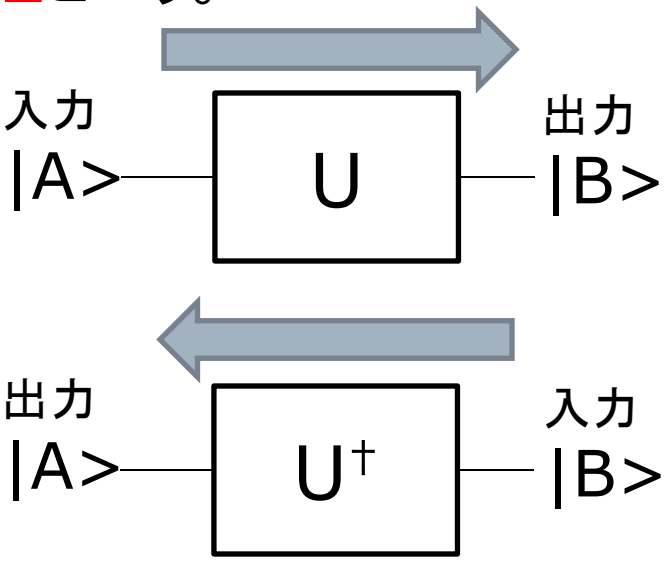
例えば、2-qubitの入力を受け取り、2-qubitを出力する量子ゲート（CNOTがそうである）は、4次元のベクトルを受け取り、4次元のベクトルに変換する 4x4の行列で表現される。

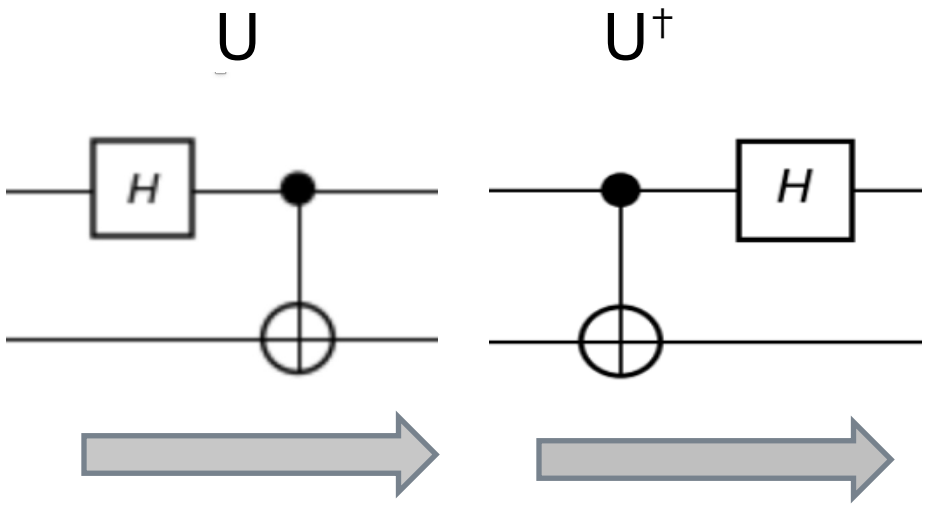
n-qubitの入力を受け取り、n-qubitを出力する量子回路は、2^n x 2^n のユニタリ行列で表現される。

量子回路の可逆性（reversibility）

回路全体を表現するユニタリ行列Uを書き下ろすことができないにしても、Uの性質 $UU^{\dagger}=U^{\dagger}U=I$ から次のことがわかる。

回路U（行列Uに対応する）の入力と出力を入れ替えても、すなわち、Uに対するある入力|A>の結果であるUの出力|B>を、Uの出力側に与えても、Uは|A>を出力する。これを、量子回路の可逆性という。



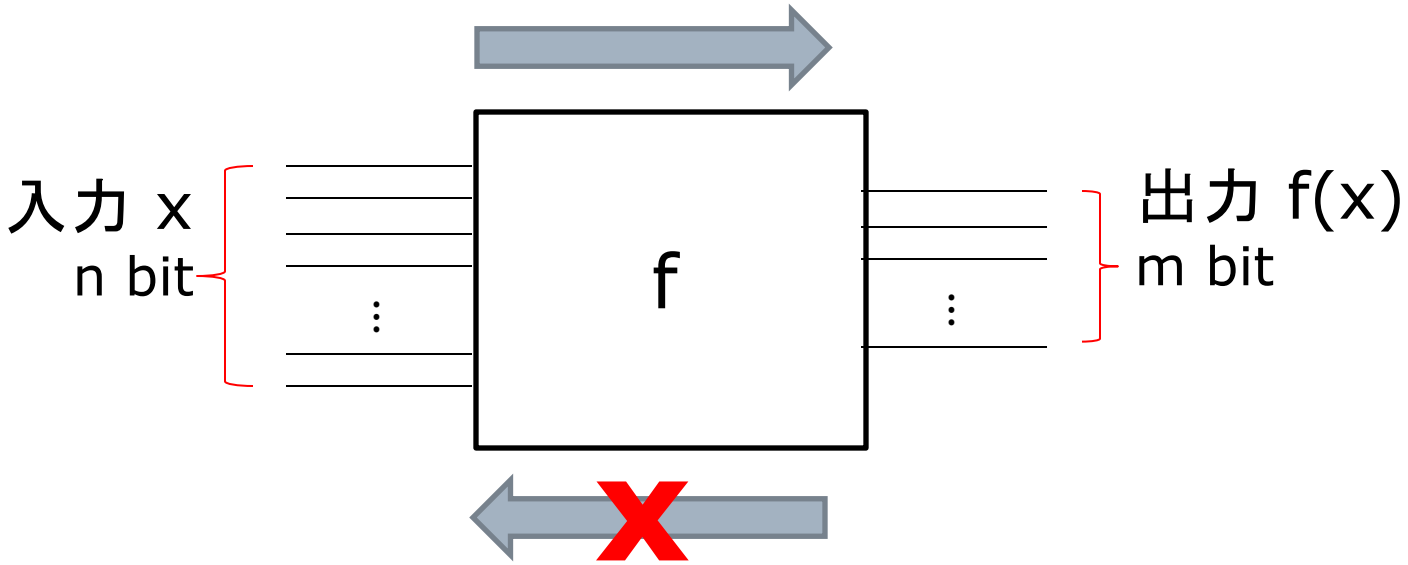


古典回路は非可逆である
任意の関数は、ユニタリでは表現できない

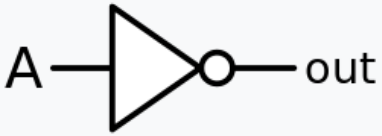
古典回路は、非可逆である。

また、任意の関数 f をユニタリ行列で表現できるわけではない。

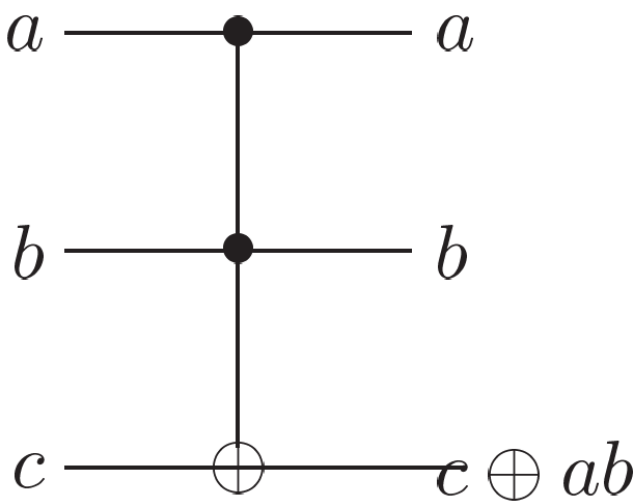
- 例えば、f(x)=z, f(y)=z という関数 f がユニタリ行列で表現できたとする。これは、U|x>=|z>, U|y>=|z>を意味するのだが、両式を引くと $U(|x\rangle-|y\rangle)=\vec{0}$ となるのだが、こうしたUはユニタリではない。



古典論理ゲートは可逆か？

論理	論理式	回路記号（MIL記号）	
NOT	$\overline{A}$		○
OR	$A + B$		×
AND	$A \cdot B$		×
XOR	$A \oplus B$		×

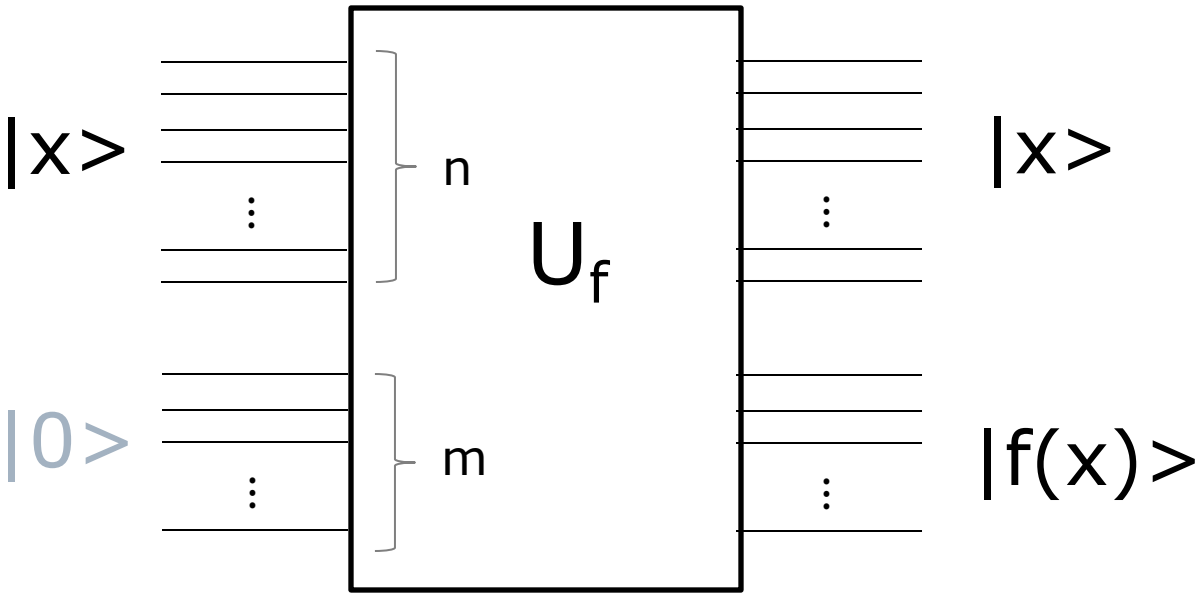
実は、次のような
ゲートを使えば、
古典回路も可逆に
できる



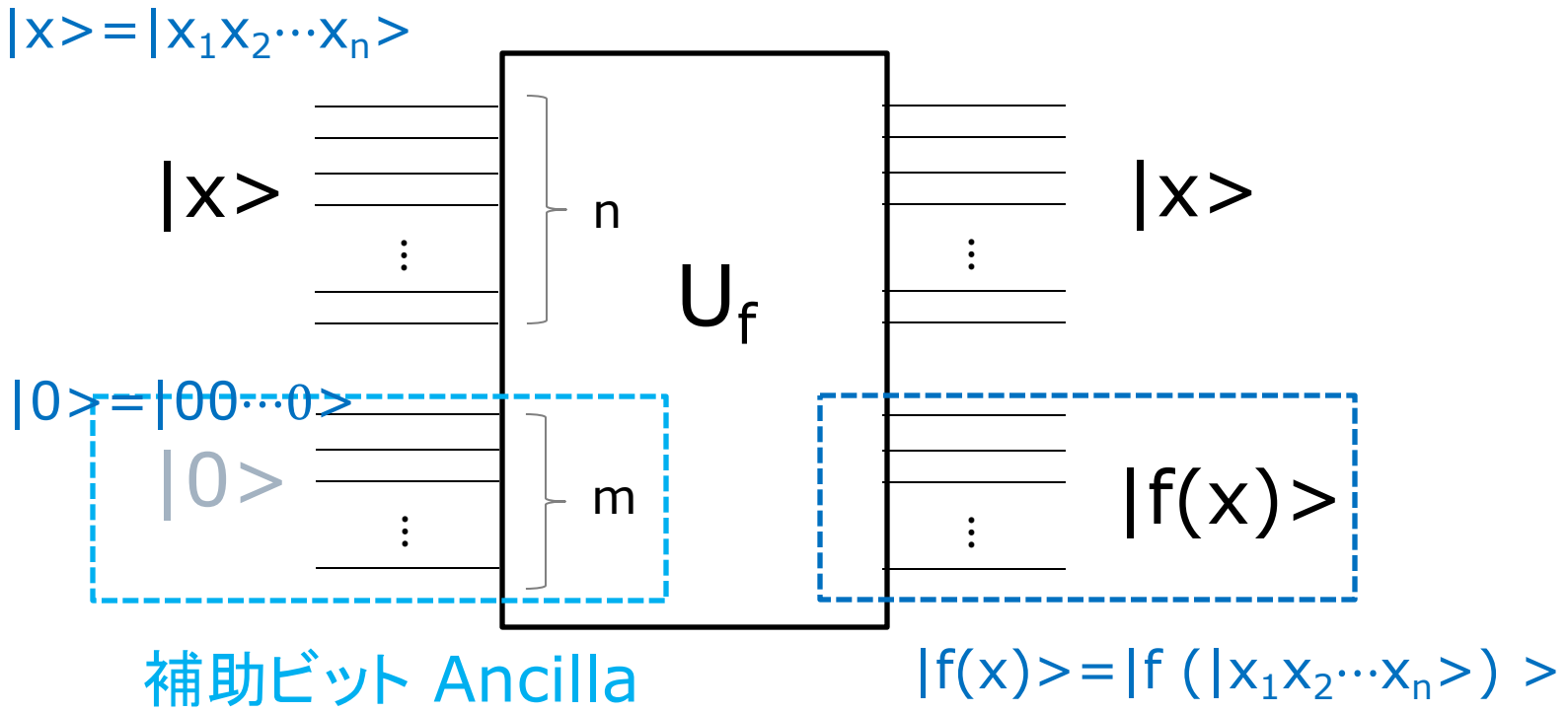
Toffoli ゲート

任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形

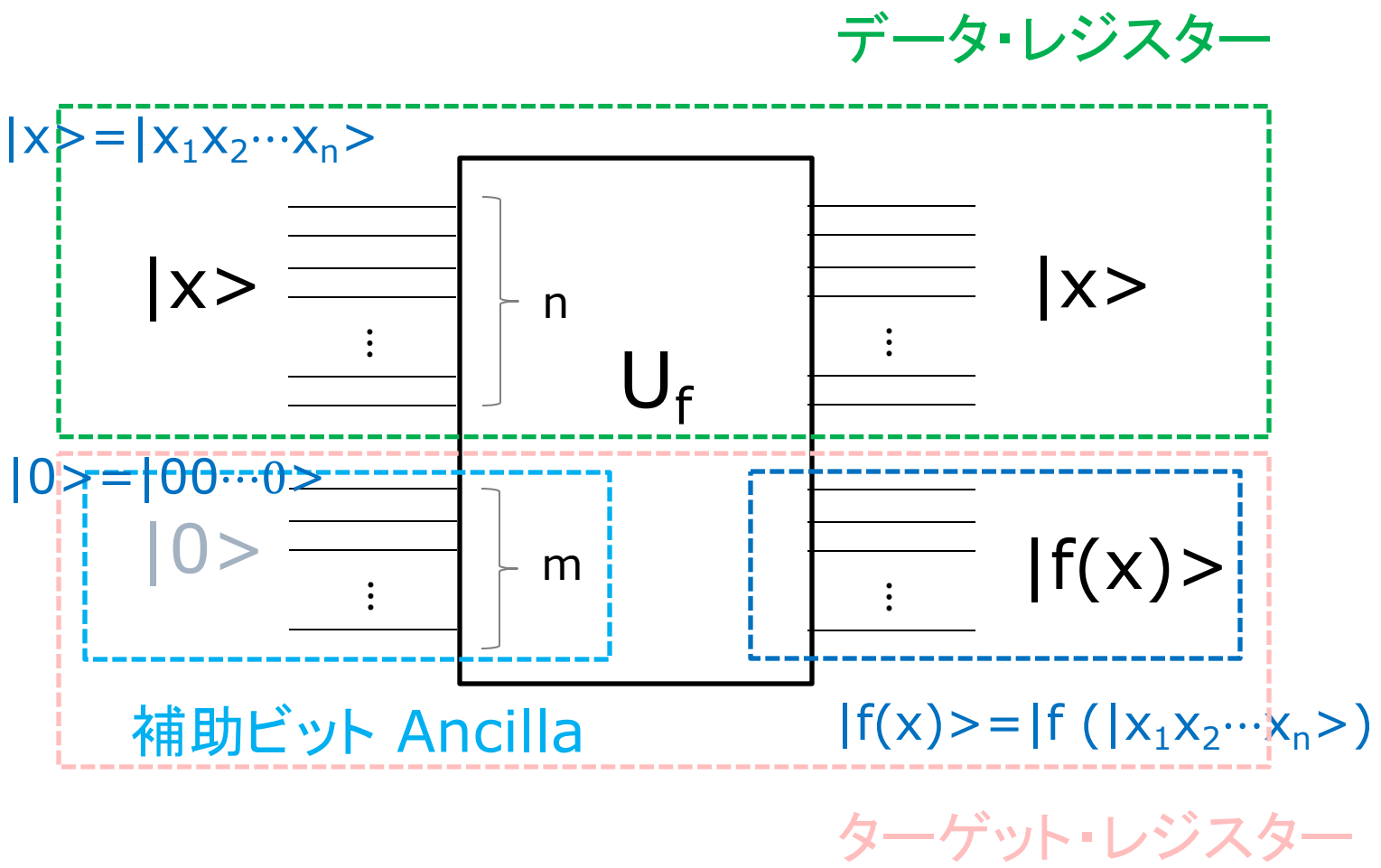
f(x)の具体的な実装を与えているわけではない。
ブラックボックスとかOracleと言ったりする



任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形

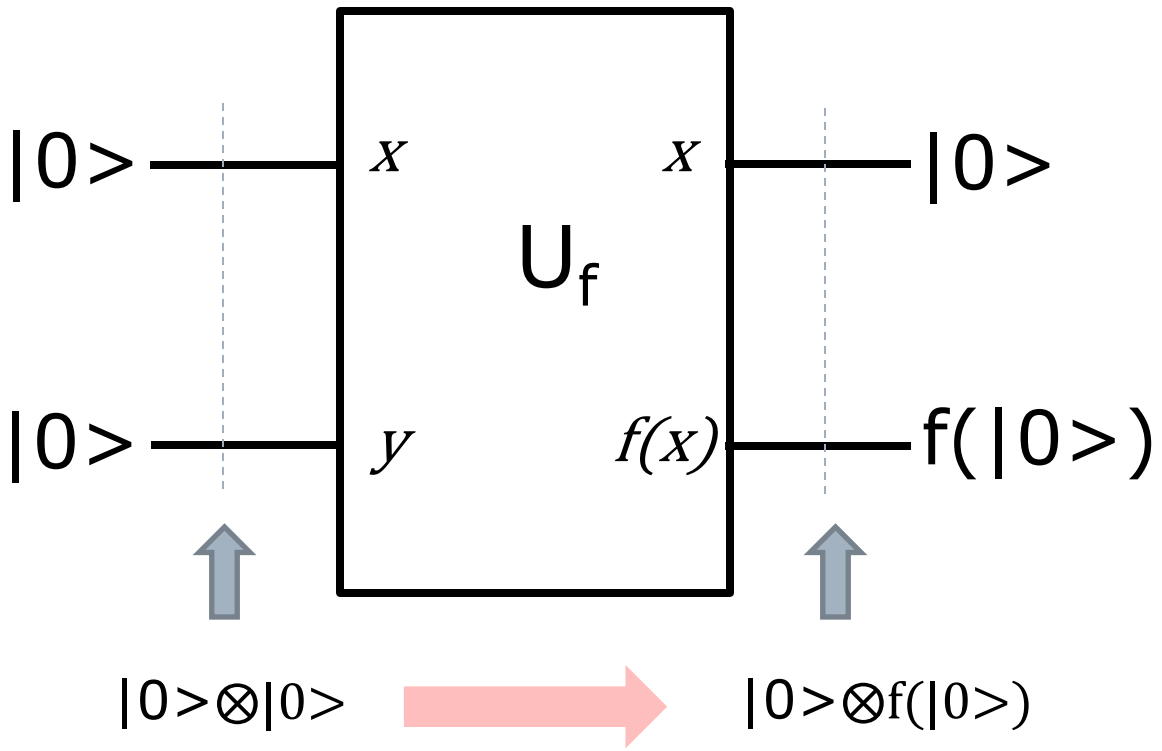


任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形



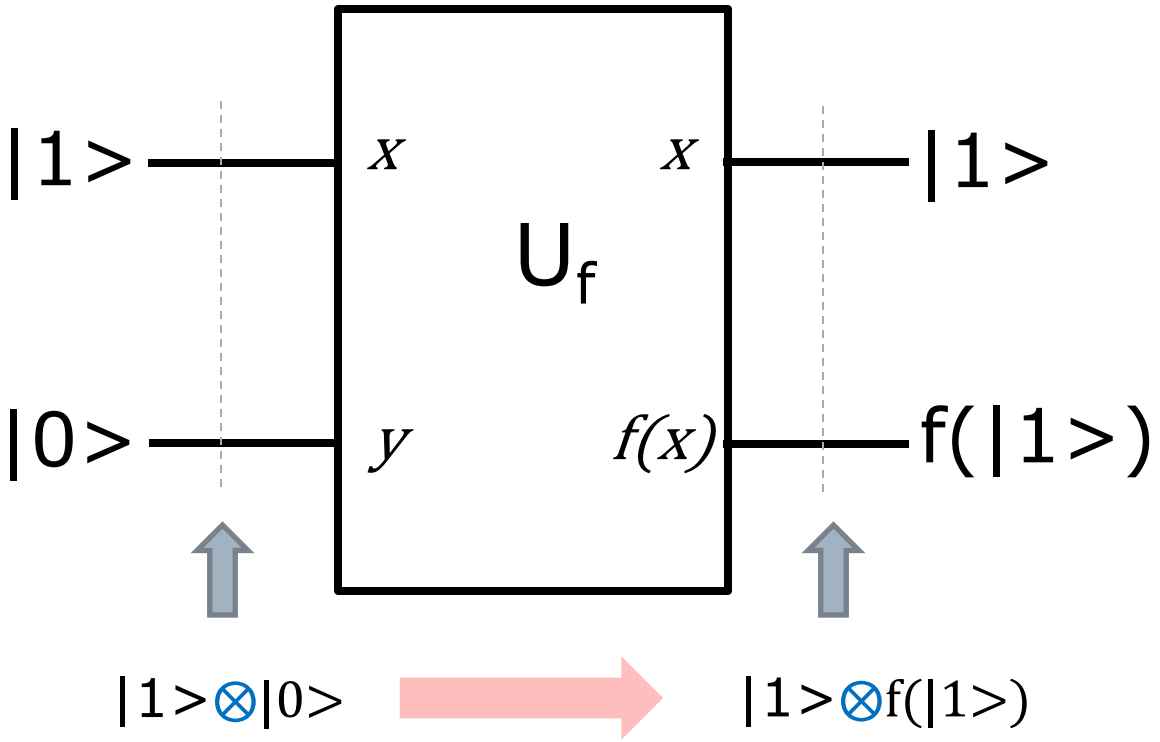
2-qubit(入力　1-qubit, 出力 1-qubit)の単純な回路U_fを考える

入力 x = |0> の時



2-qubit(fの入力 1-qubit, fの出力 1-qubit)の単純な回路U_fを考える

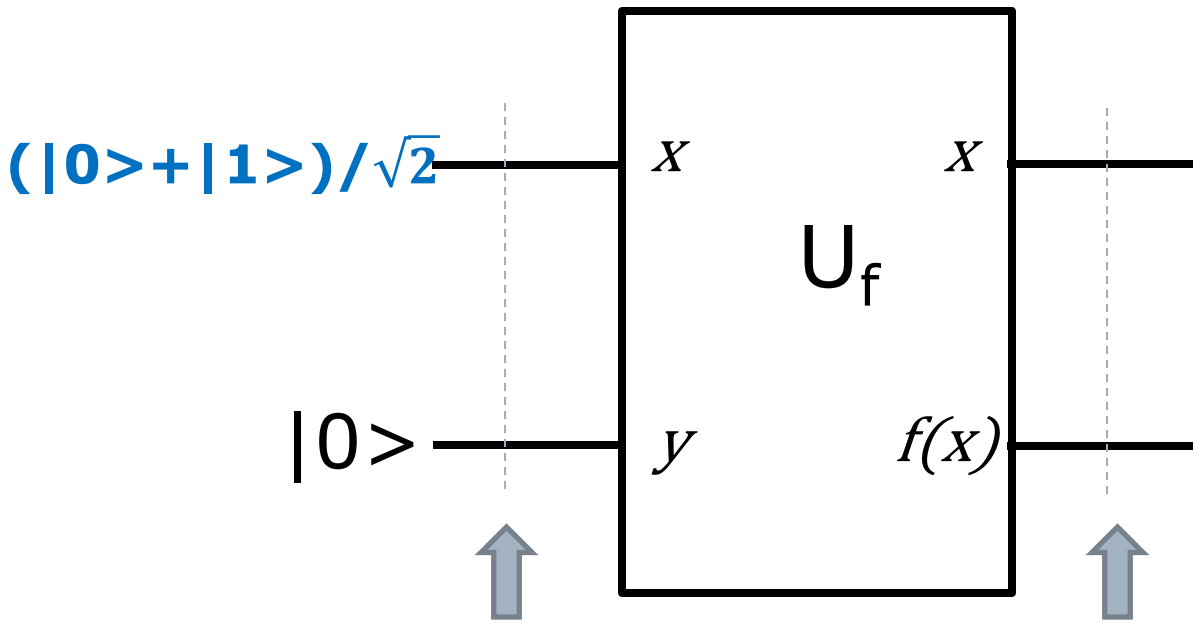
入力 x = |1> の時



意味が明確ならテンソル記号を省略してもいい

2-qubit(fの入力 1-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 $x = (|0>+|1>)/\sqrt{2}$ の時



回路の内部で
entangleが起きると
ライン毎に分離できない
可能性がある

ただ、Ufの出力全体
の状態は計算できる

$|(|0>+|1>)/\sqrt{2} \otimes |0>$ → $|0>f(|0>)/\sqrt{2} + |1>f(|1>)/\sqrt{2}$

テンソル記号を省略した

なぜ？
$|0\rangle f(|0\rangle)/\sqrt{2} + |1\rangle f(|1\rangle)/\sqrt{2}$

関数 f は線形であるとは限らないが、U_fはユニタリであり線形である。Mが線形であるとは、

$M(a|A\rangle+b|B\rangle)=aM|A\rangle+bM|B\rangle$が成り立つことをいう。

$x = |0\rangle$ の時と $x = |1\rangle$ の時の例から、U_fは次の式を満たすことがわかる。

$U_f(|0\rangle\otimes|0\rangle)= |0\rangle\otimes f(|0\rangle)$

$U_f(|1\rangle\otimes|0\rangle)= |1\rangle\otimes f(|1\rangle)$

この時、U_fは線形であるので、

$U_f((|0\rangle+|1\rangle)/\sqrt{2}\otimes|0\rangle)$

$= U_f(|0\rangle/\sqrt{2}\otimes|0\rangle+|1\rangle/\sqrt{2}\otimes|0\rangle)$

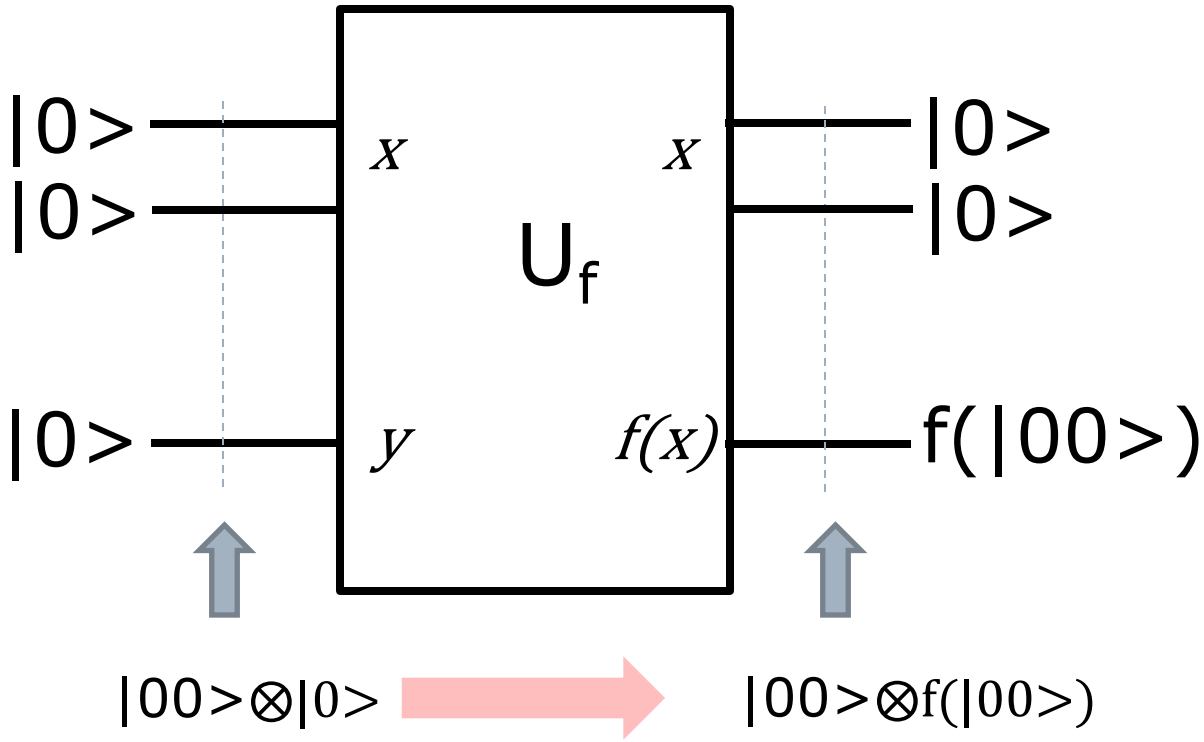
$= U_f(|0\rangle\otimes|0\rangle)/\sqrt{2} + U_f(|1\rangle\otimes|0\rangle)/\sqrt{2}$

$= |0\rangle\otimes f(|0\rangle)/\sqrt{2} + |1\rangle\otimes f(|1\rangle)/\sqrt{2}$

$= |0\rangle f(|0\rangle)/\sqrt{2} + |1\rangle f(|1\rangle)/\sqrt{2}$

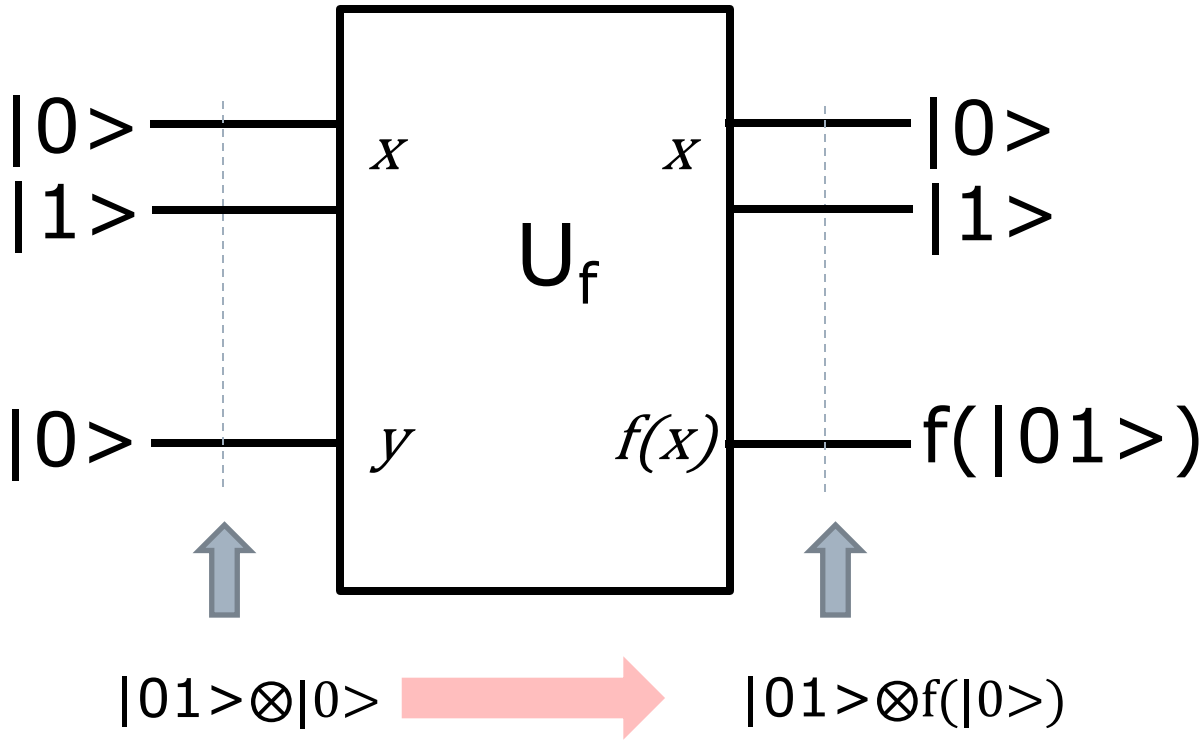
3-qubit(fの入力 2-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 x = |0>⊗|0> = |00> の時



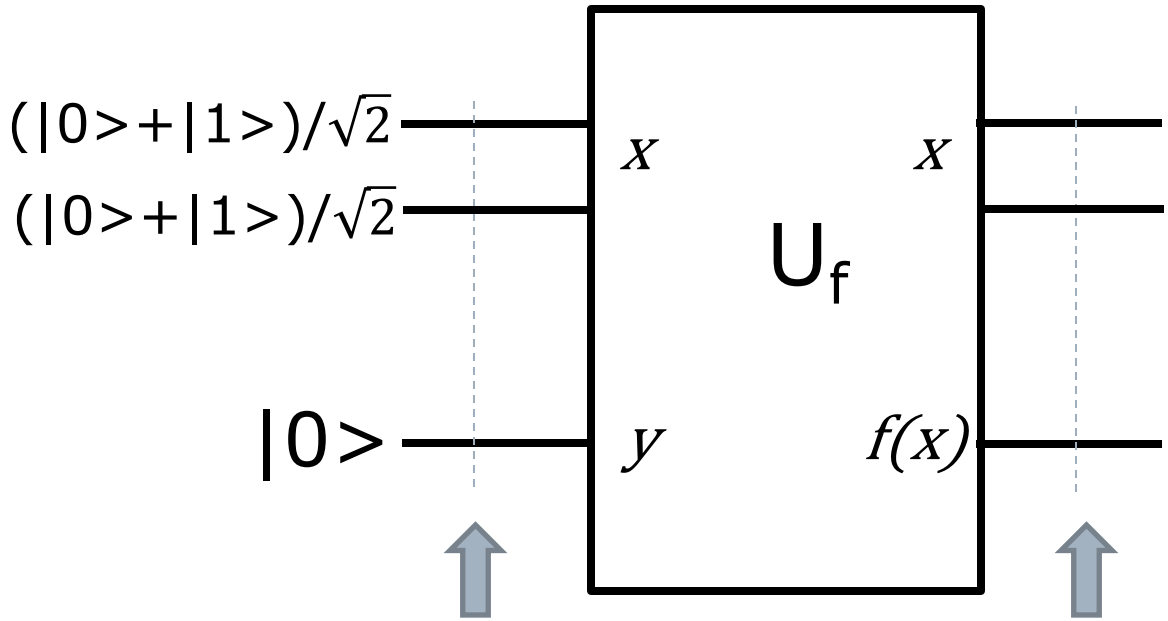
3-qubit(fの入力 2-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 x = |0>⊗|1> = |01> の時



3-qubit(fの入力 2-qubit, fの出力 1-qubit)の単純な回路U_fを考える

入力 $x = (|0>+|1>)/\sqrt{2})\otimes(|0>+|1>)/\sqrt{2}$ の時



回路の内部で
entangleが起きると
ライン毎に分離できない
可能性がある

ただ、Ufの出力全体
の状態は計算できる

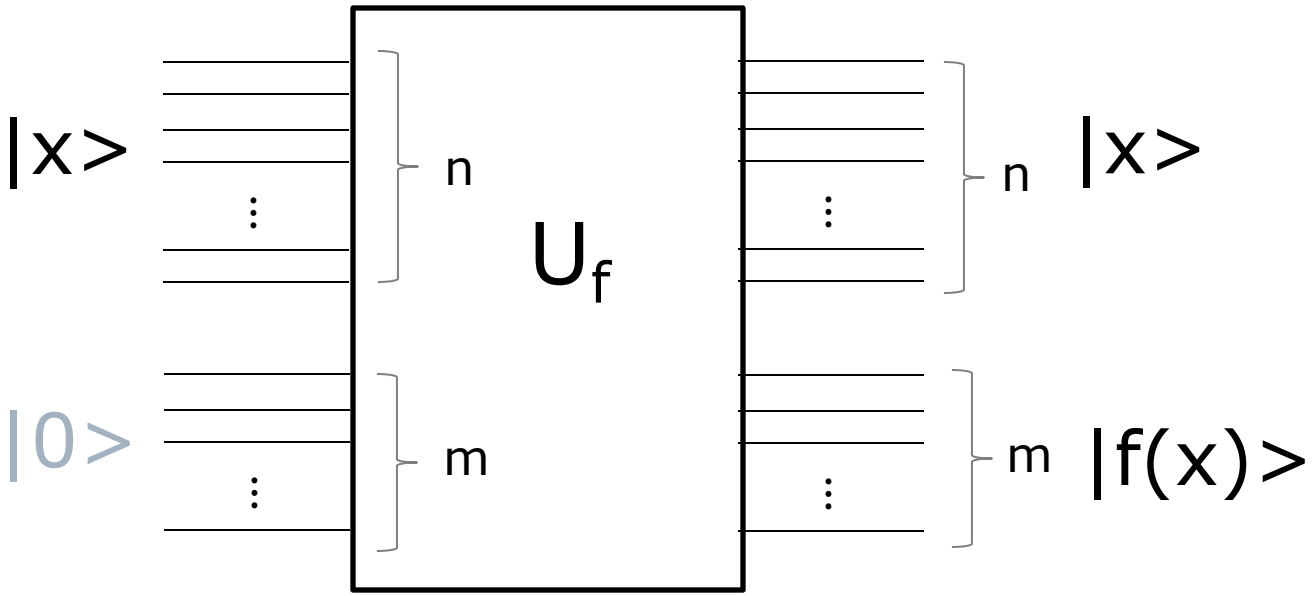
|x>⊗|0> → (|00>f(|00>) + |01>f(|01>) +
|10>f(|10>) + |11>f(|11>))/2

なぜなら

$x = (|0\rangle+|1\rangle)/\sqrt{2})\otimes(|0\rangle+|1\rangle)/\sqrt{2}$ とする。
$x = (|00\rangle+|01\rangle+|10\rangle+|11\rangle)/2$ である。

この時、
$U_f(|x\rangle\otimes|0\rangle)$
$= U_f((|00\rangle+|01\rangle+|10\rangle+|11\rangle)\otimes|0\rangle)/2$
$= (U_f(|00\rangle\otimes|0\rangle)+U_f(|01\rangle\otimes|0\rangle)+$
$U_f(|10\rangle\otimes|0\rangle)+U_f(|11\rangle\otimes|0\rangle))/2$
$= (|00\rangle\otimes f(|00\rangle)+|01\rangle\otimes f(|01\rangle)+$
$|10\rangle\otimes f(|10\rangle)+|00\rangle\otimes f(|11\rangle)/2$
$=(|00\rangle f(|00\rangle) + |01\rangle f(|01\rangle) +$
$|10\rangle f(|10\rangle) + |11\rangle f(|11\rangle))/2$

ここまでは、f(x)が1-qubitで表現される例をみてきた。このことは、関数 f(x)が、0または1の値をとることを意味する。
もし、関数f(x)が m-qubitで表現されるのなら、そのことは関数 fが、0～2^{m-1}の範囲の値を取ることを意味する。それは、古典ビットでも同様である。



$\{0,1\}^n \ni x$ で、

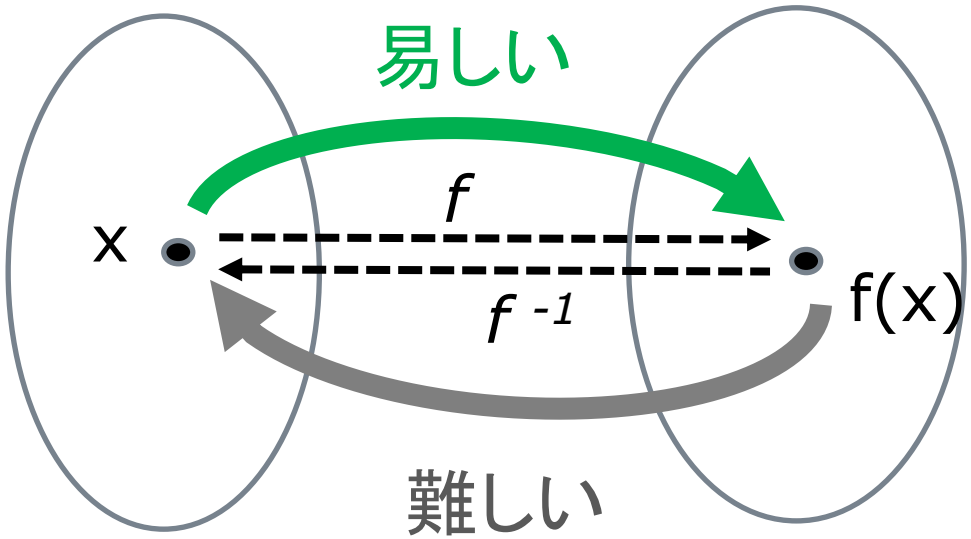
$$U_f\left(\sum |x> |0>^{\otimes m}\right) = \sum U_f\left(|x> |0>^{\otimes m}\right) = \sum |x> |f(x)>$$


Trapdoor Claw-Free Functions
-- Clawとは何か --
量子計算の古典的検証
Classical Verification of Quantum Computation

一方向関数

一方向関数　One Way Function

関数 y=f(x)は、xからf(x)を計算するのは「易しい」が、逆の計算 yから y=f(x)を満たすxを求めるのが「難しい」時、「一方向関数」という。



p,q が与えられた時
N=pqを計算するのは
易しい

Nが与えられた時
N=pqなる素因数
p,qを見つけるのは
難しい

一方向関数　One Way Function の定義

一方向関数 f の計算の「易しさ」の定義は、易しい。
f の計算が、「多項式時間で可能」であるとすれば良い。

ただ、f^{-1}の計算の「難しさ」の定義は、難しい。

- f^{-1}の計算に「指数関数的時間がかかる」ということが言えればいいように思うのだが、それを証明するのは簡単ではない。
- 「NP-完全」のクラスを利用すればいいように思うかもしれないが、暗号として使うには、「最悪の場合」に計算が難しいだけでなく、「平均して」、その計算が難しいものでなければならない。

一方向関数の存在は、P≠NPを含意する

実は、「一方向関数が存在する」という命題は、P≠NPを含意する。

だから、「一方向関数が存在する」ことは、数学的には、まだ、証明されていないのだ。

ただ、ほとんどの数学者は、P≠NPだと信じているので、一方向関数が存在すると信じている。

一方向関数のこれまでの候補

素因数分解：

N=pqなる素数p,qを求めることのの難しさを利用する。
これが、RSA暗号の基礎である。

離散対数：

実数e,x,y が $y=e^x$ を満たす時、
$x=\log_e y$と書き、xをyの（底eについての）対数という。
整数e,x,y,p が $y \equiv e^x \pmod p$を満たす時、
$x \equiv \log_e y \pmod p$と書き、xをyの離散対数という。
一般に、e,y,pが与えられた時、その離散対数 $\log_e y \pmod p$ を求めるのは難しい。これが、楕円曲線暗号の基礎である。

ただ、この二つとも、「ショアのアルゴリズム」で多項式時間で破られる。

Trapdoor Claw-free Functions

Trapdoor Claw-Free Function

λをセキュリティ・パラメタとして、λに依存する有限集合を$\mathcal{X}, \mathcal{Y}$とする。我々の目的にとって理想的な関数の族$\mathcal{F}$は次のような性質を持っている。

- それぞれの公開キーkに関して、次の二つの関数が存在する。

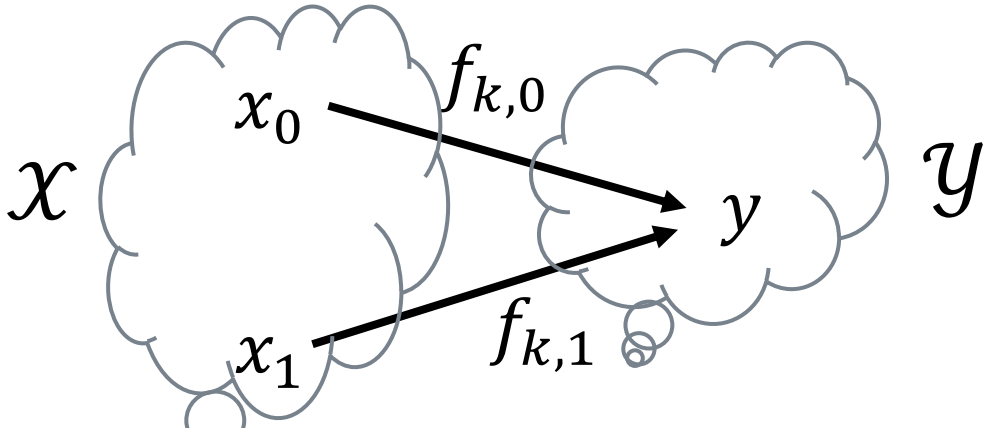
$$\{ f_{k,b} : \mathcal{X} \rightarrow \mathcal{Y} \}_{b \in \{0,1\}}$$

- 二つの関数はいずれも単射で同じ値域を持つ。すなわち、

$$(b, x) \mapsto f_{k,b}(x)$$

  は、2対1である。

Trapdoor Claw-Free Function
続き



$$f_{k,0}(x_0) = f_{k,1}(x_1) = y$$

- これらの関数は、適切なtrapdoor t_kが与えられれば、f^{-1}を計算できる。すなわち t_kは、与えられた bと $y = f_{k,b}(x)$から xを計算するのに利用される。

- さらに、これら二つの関数のペアは、claw-free である。

claw-free とはどういう性質か

上のような関係が成り立つとき、(x_0, x_1)の組をclaw と呼ぶ

- claw-free とは、どのような攻撃者にとっても、yから$f_{k,0}(x_0) = f_{k,1}(x_1) = y$となる、その二つの関数の原像の組であるclaw (x_0, x_1)を求めることが難しいことを言う。
  claw-free は、clawが存在しないと言う意味である。

- claw-free とは、関数の組 $(f_{k,0}, f_{k,1})$が、秘密キー t_kを知らない限り「一方向関数」であるということである。

Hardcore bit properties

Mahadevのアプローチでは、先の条件に加えて、関数の族 $\mathcal{F}$ は、次の性質を持つことが求められている。これを、Hardcore bit properties と呼ぶ。

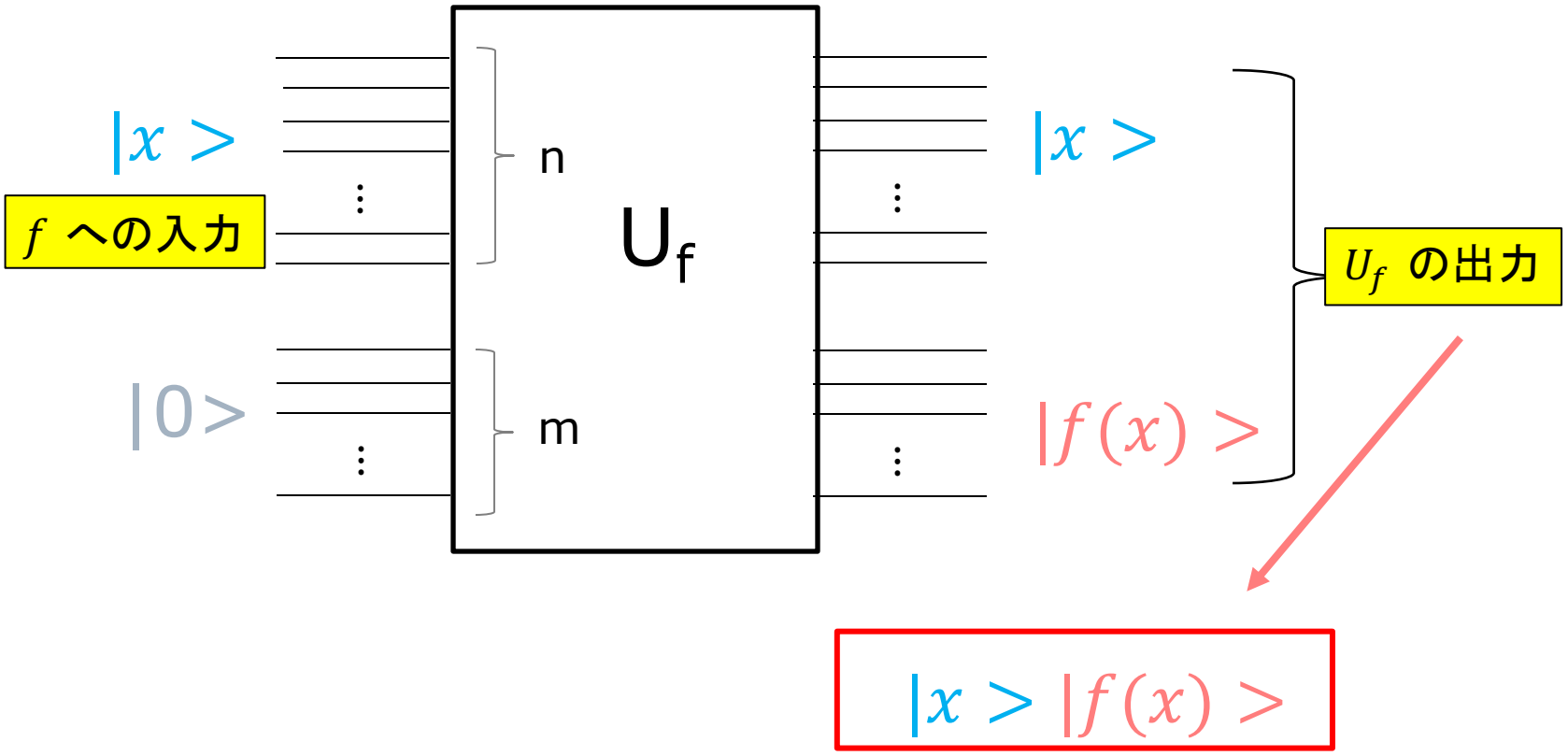
- すべてのclaw (x_0, x_1)に対して、$\mathrm{d} \neq 0$で$d \cdot (x_0 \oplus x_1) = c_k$ であるベクトルdとビットc_kが存在するが、量子コンピュータは、このd とc_kを見つけることができない。

このビットc_kをhardcore bit と呼ぶ。

Clawを含む状態を作る

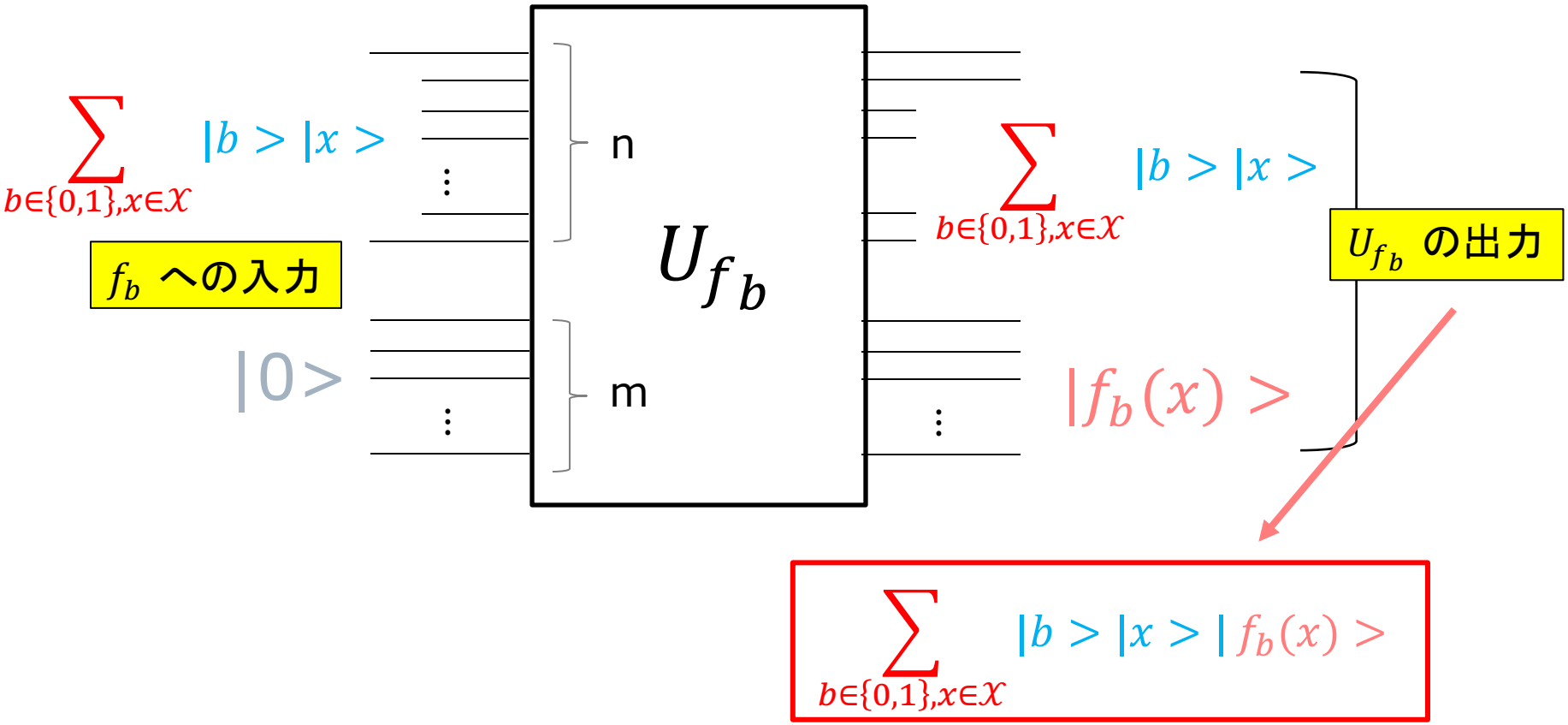
任意のf(x)を計算し、かつユニタリな量子回路U_fの一般的な形

f(x)の具体的な実装を与えているわけではない。
ブラックボックスとかOracleと言ったりする



入力が $\sum_{b\in\{0,1\},\, x\in\mathcal{X}} |b>|x>$ の場合

$$U_f\left(\Sigma|x>|0>^{\otimes m}\right) = \Sigma\, U_f\left(|x>|0>^{\otimes m}\right) = \Sigma\,|x>|f(x)>$$



量子コンピュータは、f_0, f_1 が与えられれば、内部に次のような状態を作ることができる。

$$\sum_{b\in\{0,1\},x\in X} |b>|x>|f_b(x)>$$

最後のレジスター$|f_b(x)>$を観測すると、$f_0(x_0)=f_1(x_1)=y$を得ることができる。このとき、内部の状態は、次のように変わる。

$$\sum_{b\in\{0,1\},x\in\{x:f_b(x)=y\}} |b>|x>$$

これは、次の状態である。

$$\frac{1}{\sqrt{2}}|0>|x_0>+|1>|x_1>$$


Noisy Trapdoor Claw-Free Functions using LWE
-- LWE暗号の利用 --
量子計算の古典的検証
Classical Verification of Quantum Computation

A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device

Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, Thomas Vidick

2021/05/04 ver.4 <- 2018/09/12 ver.2
https://arxiv.org/pdf/1804.00640.pdf

二つの基本的問題の解決

この論文では、純粋に古典的な検証者が、多項式時間に制限された一つの量子マシンと相互作用する新しいモデルを考察する。

量子デバイスの計算を多項式時間に制限することで、検証者は、ポスト量子暗号、すなわち、古典コンピュータ上で効率的に実装可能であるが、量子コンピュータでも解読できない暗号の基本技術を活用することができる。

このモデルでは、「信頼できない量子デバイスが〝本当に〞量子的であることを効率的に検証する方法」と「信頼できない量子デバイスから検証可能な乱数文字列を生成する方法」という2つの基本的問題に対する解決策を提案する。

第一の課題 : 量子優越性の証明問題

一つ目の課題は「量子優越性の証明問題」とも呼ばれ、既存のプロトコル[AA11, BIS+16, AC17, BFNV19, AAB+19]では、古典的スーパーコンピュータを用いて指数関数的な時間をかけて検証する必要がある。

それに対し、本論文のqubit検証テストは、古典的検証者が多項式時間で検証可能な量子性の証明を提供する。

第二の課題 :量子デバイスからの検証可能な乱数文字列の生成

また，量子デバイスからの検証可能な乱数文字列の展開に関する研究もかなり行われており[Col06, PAM+10, VV11, MS16, AFDF+18]、実験による実証も行われている[PAM+10, BKG+18]。

しかし、この課題でのすべての先行研究は、エンタングルメントを共有する複数の量子デバイスが存在し、乱数性の確認がベル不等式の違反に依存するという設定に焦点を合わせている。

量子デバイスからの
検証可能な乱数文字列の生成

また，量子デバイスからの検証可能な乱数文字列の展開に関する研究もかなり行われており[Col06, PAM+10, VV11, MS16, AFDF+18]、実験による実証も行われている[PAM+10, BKG+18]。

しかし、この課題でのすべての先行研究は、エンタングルメントを共有する複数の量子デバイスが存在し、乱数性の確認がベル不等式の違反に依存するという設定に焦点を合わせている。

問題のむずかしさ

信頼できない量子デバイスを扱う難しさの中核には、デバイスの操作を通じて、量子デバイスの内部にあるqubitの構造を持たせることを強制することにある。

すなわち、量子デバイスに実際にこの量子ビットを保持させ、また、検証者の要求に応えて、qubitの計測を実行することの難しさにある。

Noisy Trapdoor Claw-Free Functions using LWE

Learning with Errors ふりかえり

Learning with Errors 問題は、n'個の変数に対するm'個の連立一次方程式から始まる。$m' > n'$ である。

一様ランダムな行列A, sを $A \in \mathbb{Z}_q^{n \times m}$, $s \in \mathbb{Z}_q^n$ とする。
t＝Asとすると、(A, t)は簡単に解ける連立方程式になる。

問題を難しくするために、ノイズベクトル $e \in \mathbb{Z}_q^m$を追加して、

$$t = As + e$$

とする。

Learning with Errorsの分布(A,t)は、一様ランダムな分布(A,u)と区別できない

ノイズベクトルeの分布は（適切なガウス分布から）選ばれる。
sは（A, t）により一意に決まるが、それを回復するのは計算上困難であるようにする。

Learning with Errors の仮定は、一様ランダムな$u \in \mathbb{Z}_q^m$に対して、（A, t）の分布は（A, u）の分布と計算上区別できないというものである。言い換えれば、ノイズeの追加はsを計算上隠してしまうということである。

LWEからTFCファミリーを定義する

LWEサンプル $(A, t = As + e)$ が与えられた時、

$$f_0(x) = Ax + e_0$$
$$f_1(x) = Ax + e_0 + t$$

で、TCFファミリーを定義する。

e_0 は、ランダムに選ばれるので、この関数の出力は、ある分布からのランダムなサンプリングになる。

$t = As + e$ を代入すると、

$$f_1(x) = A(x + s) + e_0 + e$$

もし、$e = 0$なら、$f_1(x) = f_0(x + s)$ となる。すなわち、二つの分布は等しい。

eより広い正規分布からe_0をサンプリングすれば、$\mathrm{f}_1(x)$と$f_0(x + s)$は、統計的には近いことを保証できる。こうして、$f_1(x) = f_0(x + s)$を、効果的に保証できる。

noisy trapdoor claw-free function pair (NTCF)

こうした関数の組を、noisy trapdoor claw-free function pair (NTCF)とよぶ。

先に定義したNTCFペアのclaw (x_0, x_1, y)は、次のような性質を持つ。

- 関数ペアの全てのclaw (x_0, x_1, y) について、$x_1 = x_0 - s$

この関数ペアのclaw-freeな性質は、 x_0, x_1 の両方を知ることは、直ちにLWEの秘密キーを知ることになるので明らかである。

NTCF clawの上の重ね合わせ

量子デバイスは、NTCFを使ってclawの上に重ね合わせを設定することができる。

$$\frac{1}{\sqrt{2}}(|0>|x_0>+|1>|x_1>)=\frac{1}{\sqrt{2}}(|0,x_0>+|1,x_0-s>)$$

次のようにする。

$$\sum_b \sum_x \sum_{e_0} |b>|x>|Ax+e_0+bt> \text{をつくる}$$

$|f_b(x)>$

最後のレジスターを測定してyを得る

最初の2つのレジスタの望ましい重ね合わせを作成する


Part IV
Mahadev


Mahadevのプロトコル (1)
-- 量子性検証プロトコル --
量子計算の古典的検証
Classical Verification of Quantum Computation

Mahadevの
プロトコル (1)

量子性検証プロトコル

"Classical Verification of Quantum Computations"
Urmila Mahadev

Proverができること

- Proverは、自分で任意のqubitを一つ用意する。

$$|\psi> = \alpha_0|0> + \alpha_1|1>$$

- Proverは次のような計算が可能である。
  - 関数fの定義域$\mathcal{X}$の要素全ての重ね合せの状態をつくる

$$\frac{1}{\sqrt{|\mathcal{X}|}}\sum_{x\in\mathcal{X}}|x>$$

  - 重ね合わせの状態に関数 fを適用する

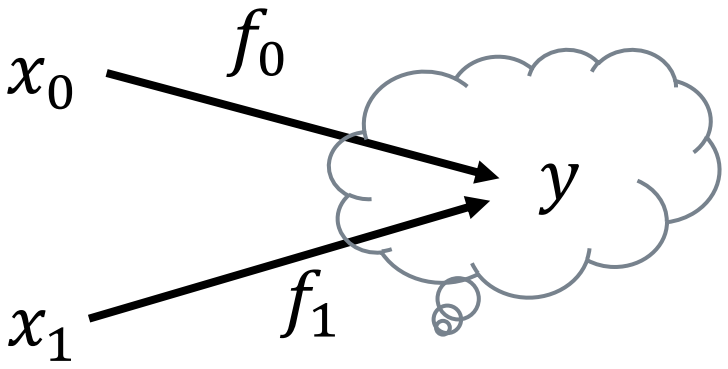
$$\frac{1}{\sqrt{|\mathcal{X}|}}\sum_{x\in\mathcal{X}}|x>|f(x)>$$

  - 最後のレジスターを測定して、$y = f(x)$を得る

Verifierができること

次の性質を持つ、Trapdoor claw-free function (TCF) ペア f_0, f_1 を生成できる

- 単射で、同一の像をもつ
- キーを知っていれば、容易に逆関数を求めることができる: yが与えられた時、$f_0(x_0) = f_1(x_1) = y$ となる x_0, x_1 を出力できる
- claw (x_0, x_1): $f_0(x_0) = f_1(x_1)$ を見つけるのは困難である



Verifierは、TCFペア f_0, f_1 を Proverに送る

- Verifierは、TCFペア(f_0, f_1)を生成し、それをProverに送り、計算を指示し、結果を返すように求める。
- Verifierは、トラップドアを知っているので、$f_0(x_0) = f_1(x_1) = y$ である claw (x_0, x_1)を知っている。

(f_0, f_1)

Verifier

Prover

Proverのペア(f_0, f_1)の計算

- Proverは次の状態を用意する。

$$\sum_{x\in\mathcal{X}} \alpha_0|0>|x> + \sum_{x\in\mathcal{X}} \alpha_1|1>|x>$$

$$= \alpha_0|0> \sum_{x\in\mathcal{X}} |x> + \alpha_1 |1> \sum_{x\in\mathcal{X}} |x>$$

- f_0, f_1を計算すると、内部の状態は次のように変わる。

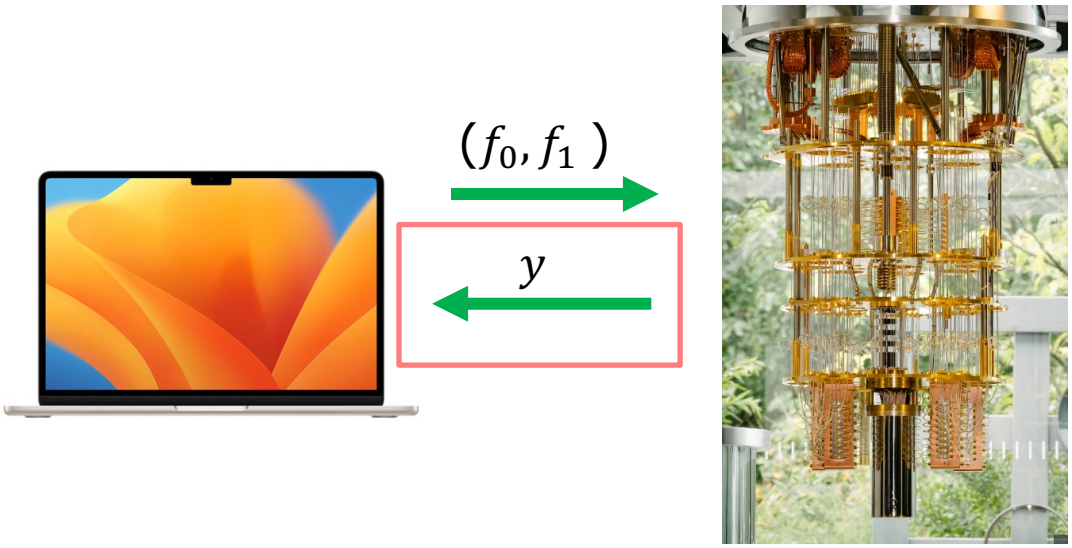
$$\alpha_0|0> \sum_{x\in\mathcal{X}} |x>|f_0(x)> \; + \alpha_1 |1> \sum_{x\in\mathcal{X}} |x>|f_1(x)>$$

- これは、次のように表現できる。

$$\sum_{b\in\{0,1\},x\in\mathcal{X}} \alpha_b|b>|x> \xrightarrow{(f_0,f_1)} \sum_{b\in\{0,1\},x\in\mathcal{X}} \alpha_b|b>|x>|f_b(x)>$$

Proverは、f_0, f_1 の計算結果yを Verifierに送る

- Proverは、f_0, f_1 の計算結果yを Verifierに送らねばならないのだが、そのためには、少し作業が必要になる。
- このyの評価・送出によって、Proverの内部状態は変化する。



yを求める

- yの評価以前の状態

$$\sum_{b\in\{0,1\},x\in X} \alpha_b |b>|x>|f_b(x)>$$

- $y = f_0(x_0) = f_1(x_1)$ である。
  yを取り出すには、この状態の第三レジスターを測定すればいい。

- 測定後、システムの状態は次のようになる。

$$\sum_{b\in\{0,1\}} \alpha_b |b>|x_b>$$

yを求めることで起きる変化

- yの評価以前のProverの状態

$$\sum_{b\in\{0,1\},x\in X} \alpha_b|b>|x>|f_b(x)>$$

- yの評価以後のProverの状態

$$\sum_{b\in\{0,1\}} \alpha_b|b>|x_b>$$

$$=\boxed{\alpha_0|0>|x_0>+\alpha_1|1>|x_1>}$$

この状態は、最初にprover が持っていたqubitの状態 $|\psi>=\alpha_0|0>+\alpha_1|1>$と claw (x_0,x_1)の重ね合わせの状態である。

Proverの内部の状態の変化

最初の状態

$$\sum_{b\in\{0,1\}} \alpha_b |b>$$

(f_0, f_1)を適用した状態

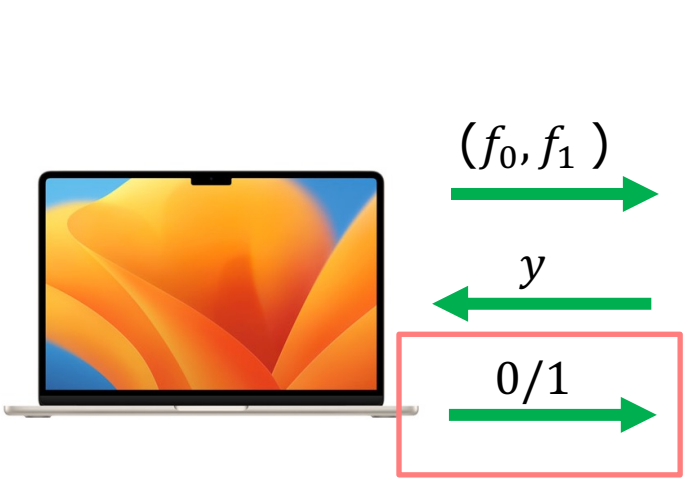
$$\sum_{x\in X}\sum_{b\in\{0,1\}} \alpha_b |b>|x>|f_b(x)>$$

$f_b = y$ を測定した状態

$$\sum_{b\in\{0,1\}} \alpha_b |b>|x_b>$$

VerifierはProverに、内部状態の測定を求める

- VerifierはProverに、標準基底あるいはHadamard基底での測定を求める。
- 標準基底での測定を求めるなら、1を送る。
  Hadamard基底での測定を求めるなら、0を送る。

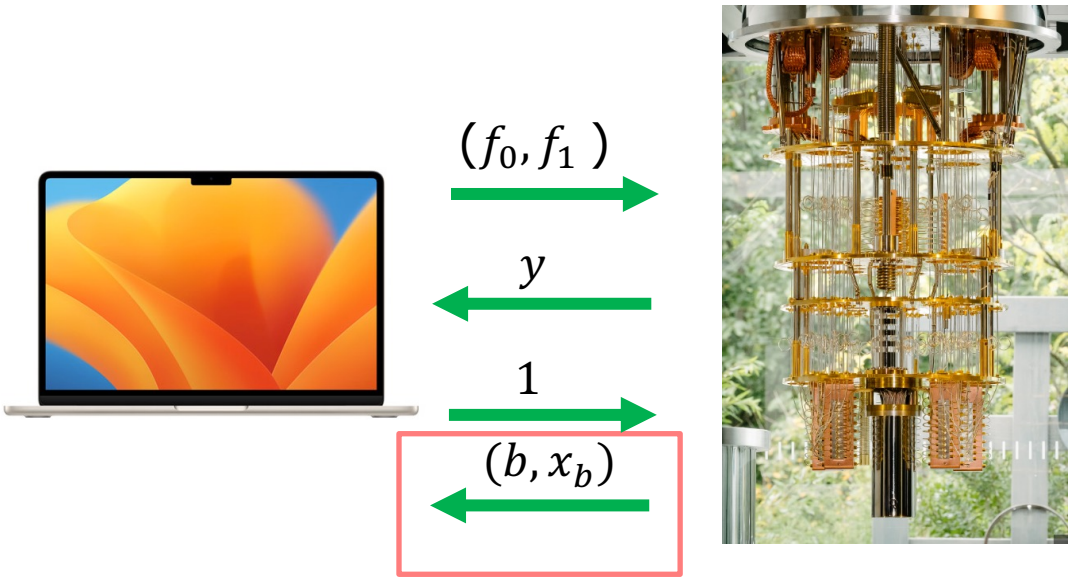


Verifier

Prover

ProverはVerifierに、測定結果を返す
標準基底の場合

- 標準基底の場合ProverはVerifierに、(b, x_b) の形で測定結果を返す。



標準基底での測定

$\alpha_0|0>|x_0>+\alpha_1|1>|x_1>$

を標準基底で測定する

確率$|\alpha_0|^2$ で、$|0>|x_0>$を観測し

確率$|\alpha_1|^2$ で、$|1>|x_1>$を観測する。

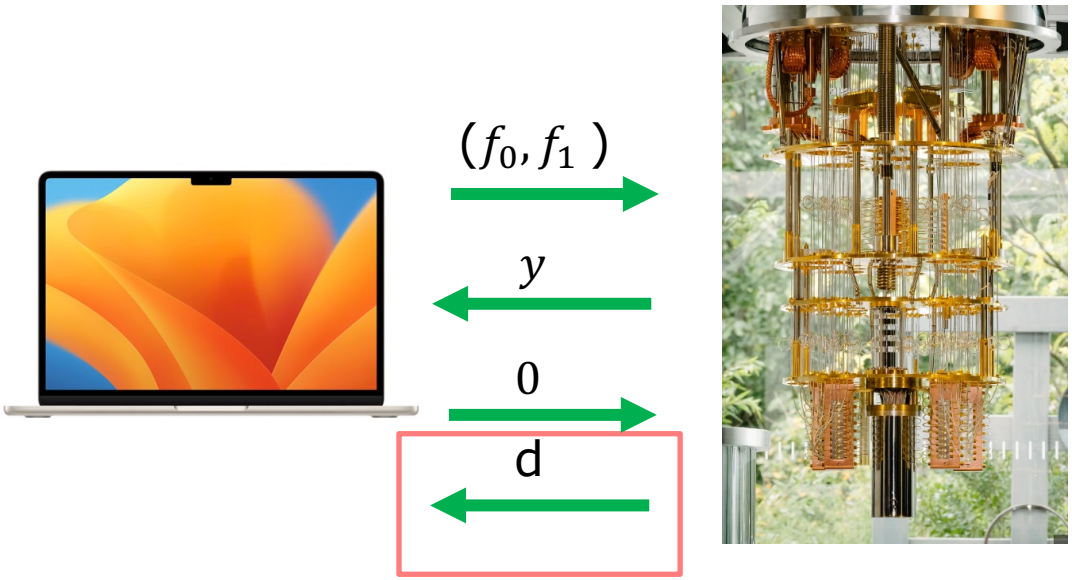
確率$|\alpha_0|^2$ で、$(0,x_0)$を送り

確率$|\alpha_1|^2$ で、$(1,x_1)$を送る。

ランダムにyの原像をを返すことになる。

ProverはVerifierに、測定結果を返す
Hadamard基底の場合

- Hadamard基底の場合ProverはVerifierに、$d \cdot (x_0 \oplus x_1) = 0$ をみたすdを返す。



Hadamardでの測定

$\alpha_0|0>|x_0>+\alpha_1|1>|x_1>$
をHadamard基底で測定する

これは、少し面倒である。
基本的なことを確認しておこう。

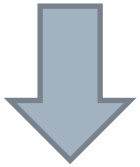
$\alpha_0|0> + \alpha_1|1>$ の測定

$|\psi> = \alpha_0|0> + \alpha_1|1>$ の測定

$$H|\psi> = \frac{1}{\sqrt{2}}(\alpha_0 + \alpha_1)|0> + \frac{1}{\sqrt{2}}(\alpha_0 - \alpha_1)|1>$$

0を観測する確率は、$\left|\frac{1}{\sqrt{2}}(\alpha_0 + \alpha_1)\right|^2$

1を観測する確率は、$\left|\frac{1}{\sqrt{2}}(\alpha_0 - \alpha_1)\right|^2$



bを観測する確率は、$\frac{1}{2}\left|\alpha_0 + (-1)^b \alpha_1\right|^2$

いままで見てきたように、量子デバイスはclawを計算することはできないが、それでも、すべての入力に対して一様な重ね合わせでfを評価する。それは、yだけでなく，yの二つの原像に対する次のような重ね合わせ

$$\alpha_0(|0>|x_0>+\alpha_1|1>|x_1>)$$

を同時に保持することができる。

二つの原像x_0, x_1が、重ね合わせで保存されていることを利用するために、状態の最初の量子ビットを除いてHadamard基底で測定を行い、文字列 $d \in \{0,1\}^n$)を生成することができる。

$$\frac{1}{\sqrt{2}}(|x_0>+|x_1>)$$

$\frac{1}{\sqrt{2}}(|x_0> + |x_1>)$の測定

結果だけを示す。

claw (x_0, x_1)の重ね合わせ$\frac{1}{\sqrt{2}}(|x_0> + |x_1>)$を、Hadamard変換して次の形を得る。

$$\frac{1}{\sqrt{x}}\sum_d \left((-1)^{d\cdot x_0} + (-1)^{d\cdot x_1}\right)|d>$$

これを測定すると、次の条件を満たす dを得る。

$$d \cdot (x_0 \oplus x_1) = 0$$

Hadamard基底での測定に成功するのは、量子デバイスだけである

Proverは、標準基底での測定でも、Hadamard基底の測定でも、正しい答えを返すことができる。

答えが正しいかどうかは、Verifierは、yに対する原像 x_0, x_1 の値をあらかじめ知っているので、いずれの測定の場合でもProverの答えが正しいかはすぐにチェックできる。

一方、claw-freeという性質は、x_0, x_1を同時に知ることは計算上困難であることを意味する。古典デバイスでは、$d \cdot (x_0 \oplus x_1) = 0$ という性質を持つdを求めることはできない。

Hadamard基底での測定は、量子デバイスと古典デバイスを明確に区別することになる。

量子性検証プロトコル

1. 検証者はトラップドアとともにTCFペアを生成し、関数ペアを証明者に送信する。

2. 証明者は、TCFペアのイメージyを返す。

3. 検証者は、ランダムに標準基底かHadamard基底での測定を証明者に求めて、yの原像か、ビットcと、$d \cdot (x_0 \oplus x_1) = c$となるnビット文字列dが返るのを待つ。

4. 証明者は要求された出力を返し、検証者はトラップドアのキーを使用してyの2つの原像x_0, x_1を計算して妥当性をチェックする。