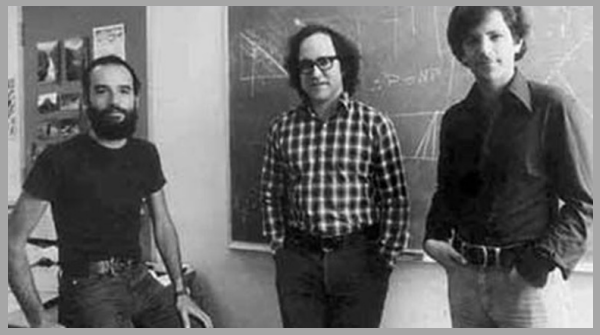
暗号技術の現在



Public Key/ RSA

Shor’s Algorithm

Post-Quantum Cryptography

暗号技術の現在 **Agenda**

概説

第一部　現代暗号技術の成立

- 現代暗号技術成立以前
- 現代暗号技術の成立
- 暗号と計算複雑性理論

第二部　ショアのアルゴリズムの発見

- Shorのアルゴリズムのインパクト
- 確率的素数判定
- 量子回路で「周期」を発見する
- 計算複雑性の新しいクラス BQP

暗号技術の現在 **Agenda**

第三部 ラティス暗号の時代の始まり

- 概説 －－ 主要なプレーヤー達
- ラティス入門
  - ラティスとは何か -- 格子点を表現する
  - 基底でラティスを定義する
  - ラティス問題
  - 同じラティスを生成する複数の基底

暗号技術の現在

概説

-- エピソードで振り返る暗号史 --

Turingの「名誉回復」

かつて（といっても、そう遠くない過去には）、暗号技術は「国家機密」だったことは、多くの人は知っていると思う。第二次大戦中のドイツの暗号機 Enigma をめぐる攻防は、有名である。

Enigma暗号の解読に大きな役割を果たしたのはTuring だった。ただ、Turingのこの分野での功績は、生前、公式には認められることは一度もなかった。誰が暗号解読にたずさわり、どんな仕事をしたかは「国家機密」だったからだ。

1952年、Turingは「同性愛法」で訴えられる。当時のイギリスでは、同性愛は犯罪だった。彼は、投獄か薬物的去勢かの選択をせまられる。

英国のブラウン首相は2009年、エリザベス女王は2013年、戦時中のTuringの貢献を高く評価し、公式に謝罪し、Turingの「名誉回復」がなされる。ただ、それは Turingが自殺してから、50年以上たってからだった。

Nashの慧眼

現代の暗号化技術の大きな特徴は、その基礎を、コンピュータはある種の数学的問題を、「効率的」には解くことができないという（経験的）事実においていることである。その理論的基礎は、数学的には、「計算複雑性の理論」である。

そのことに最初に気づいたのは、Nashだった。

1955年、数学者のJohn Nashは、NSAに対して「鍵の計算に指数関数的時間のかかる」方法を使えば、「誰にも破れない暗号を簡単に作れるようになる」という手紙を送っている。

暗号には「秘密」がつきものであった。Nashの発見は、機密とされた。

1970年代の初めには、イギリスの情報機関GCHQの科学者たちが、今日の公開キー暗号と同じものを作り上げていた。それはNSAも知っていた。ただ、それらは全て「機密」とされ、世に知られることはなかった。

Security = National Security

なぜ、「機密」にされたかといえば、securityの問題は、第一義的に”National Security”の問題だからということだ。

ただ、なぜそうした技術が、広く実用化されなかったという点に関して言えば、別の問題もあったことに気づく。それは、当時のコンピュータには、少なくとも経済的には、こうした複雑な計算を実行する計算能力が十分ではなかったのである。

国家機密から標準技術へ

現在の暗号化技術の基本が出来上がるのは、1976/1977年のことである。

その技術の真価は、80年代のコンピュータのコンシューマライゼーション（PC-AT, Windows, ...）を経て、90年代半ばのインターネットのグローバルな拡大、経済活動のグローバル・ネットワーク化、個人のネットワークへの登場を通じて、全面的に開花する。この技術なしでは、今日のネットワークの成功はなかったろう。

暗号技術は変化する

このことは、同時に、暗号化技術が、もはや、もっぱら ”National Security”のみにかかわる技術ではなくなったことを意味する。今では、誰もが暗号化技術を必要とし、誰もがそれを、オープンな「標準技術」として利用できる。それは、大きな変化である。

暗号化技術は、その時代で利用可能なコンピュータの計算能力、その時代で利用可能な通信基盤に大きく依存している。それは歴史的に変化するものだと僕は考えている。

Shorの発見
過大評価と過小評価

現代の暗号技術の基礎が確立して約20年後の1994年、ベル研究所のPeter Shorは、量子コンピュータを使えば、RSA暗号の基礎である素因数分解問題を、極めて「高速に」解くことができるアルゴリズムを発見する。楕円曲線暗号の基礎である離散対数問題も、このアルゴリズムで「高速に」解ける。

この発見は、当時、一大センセーションを巻き起こした。しかし、その熱もやがて冷める。というのも、そのアルゴリズムを実現するための量子回路を構築することが、当時の技術では「事実上不可能」であることが、次第に明らかになったからだ。

こうして、まだ存在しない量子コンピュータは、現在の暗号化技術に対する「当面の脅威ではない」という認識が一般には広く共有されることになる。

それもある意味では当然である。先にも触れたように、暗号技術は「その時代で利用可能なコンピュータの計算能力、その時代で利用可能な通信基盤に大きく依存している」からである。

ただ、繰り返すが、現代の暗号化技術が、「計算複雑性理論」に依存している。

現在のコンピュータでは効率的には計算できないが、量子コンピュータでは効率的に計算可能な複雑性のクラスが存在することを具体的に示した、Shorの発見は、研究者の強い関心を途切れることなく引き続けた。

筆者は、Shorのアルゴリズムの発見は、暗号技術の一つのステップとしてだけでなく、21世紀が、量子コンピューターや量子通信技術が花開く「量子の世紀」になることの、歴史的に重要な「先駆け」だったと考えている。

NSA

21世紀に入って、量子コンピュータの研究はさらに広がり、Google, IBM, Microsoft, Intelといったベンダーがこの領域に参入するとともに、Shorのアルゴリズムの「実現可能性」についての議論が広く行われるようになった。

このような状況を受けて、2015年、NSAは次のような重要な決定を公表した。 Shorの発見から、約20年後のことであった。
「我々は、来るべき量子耐性アルゴリズムへの移行について、早いうちから計画づくりとコミュニケーションを開始することを決定した。我々の最終的な目標は、量子コンピュータの潜在的な能力に対して、コスト効率の良いセキュリティを提供することである。」

NIST レポート

NSAの決定を受けて、NISTは2016年から “Post-Quantum Cryptography”の標準化の策定の作業を開始した。

その計画によれば、早ければ2022年、遅くとも2024年までには、新しい「ポスト量子暗号技術」の標準を決定するという。

先月7月、米NIST (National Institute of Standards and Technology)は、「ポスト量子暗号技術の標準化」についての文書 NIST IR 8413 を公開した。

Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process
https://doi.org/10.6028/NIST.IR.8413

暗号技術の現在について、最も重要な動きは、このNISTのプロジェクトであると、僕は考えている。

ラティス暗号の時代の始まり

暗号技術は、今、「ポスト量子暗号」への移行という大きな転換点を迎えようとしている。

この「ポスト量子暗号」の技術的中核は、「ラティス暗号」である。

現在は、ラティス暗号の時代の始まりと捉えることが出来る。

未来展望

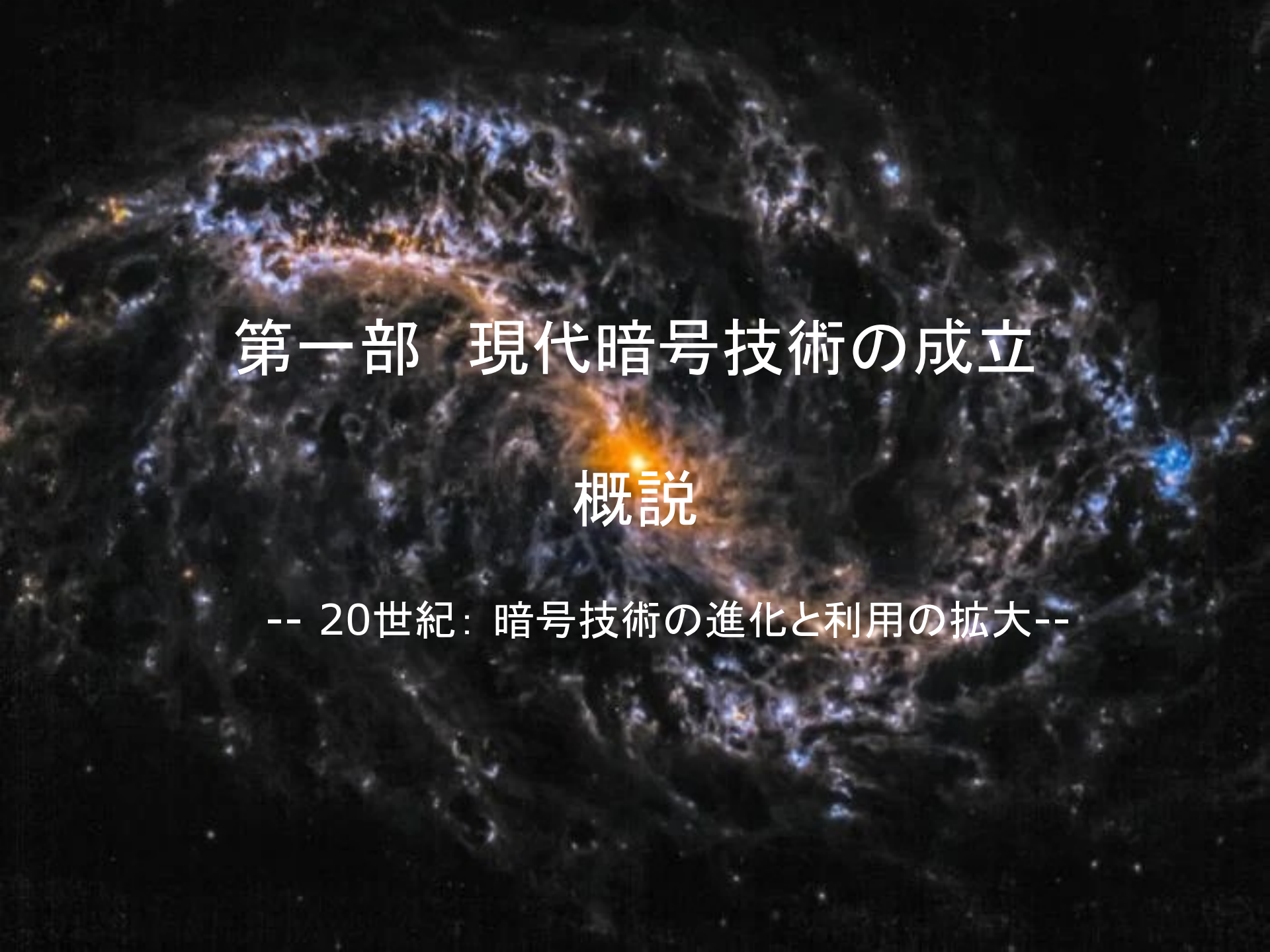
暗号技術は、時代とともに技術の到達点とともに、変化するものだ。暗号技術の歴史を見れば、ほぼ20年ごとに、大きな変化が起きているのがわかる。

一つの問題は、IT技術者にとって、20年とか40年のスパンで技術の未来を考えることが、なかなか難しいことだ。それは誰にとっても同じだと思う。しかし、未来を予測することなしに、未来の技術の準備はできない。

現在のコンピュータと量子コンピュータのハイブリッドの時代、また、現在の通信技術と量子通信のハイブリッドの時代は、新しい暗号技術を生み出すだろう。


© NASA/ESA/CSA/Judy Schmidt

第一部　現代暗号技術の成立



第一部　現代暗号技術の成立

概説

-- 20世紀：暗号技術の進化と利用の拡大--

20世紀：暗号技術の進化と利用の拡大

第二次大戦期にはプリミティブなものだった暗号技術は、1970代の後半には、以前の暗号技術とは全く異なる相貌を持つ現在の暗号技術が成立する。

20世紀後半には、インターネットの爆発的拡大と共に暗号技術の利用も爆発的に拡大する。それは現代の社会・経済活動を支える最も重要な技術の一つとなった。

現代暗号技術の成立とは独立に、ほぼ同時期に、「計算複雑性理論」が登場し、その抽象的理論が実践的な暗号技術の理論的基礎として応用できるという認識が生まれる。

コンピューターと暗号技術

すでに、大戦期の暗号解読で、BombeやColossusといった機械の利用が試みられていた。これらは、現代のコンピューター技術の起源の一つと考えられる。

20世紀半ばに登場したコンピューターが、計算能力を向上させ、低価格化し小型化するのには、それほど多くの時間を必要としなかった。

コンピュータの「コンシューマライゼーション」の進行は、暗号技術の利用の拡大の前提条件を形成した。

通信の世界の変化と暗号技術

現代の暗号技術の利用の拡大の決定的な要因となったのは、通信の世界の変化である。

コンピューターの世界と同様に、通信の世界も絶えざる変化を続けてきた。電信から電話へ、ラジオからテレビへ。20世紀後半に成立するインターネットとその拡大は、通信の世界を劇的に変えた。

こうして、誰もが暗号技術を利用する世界が生まれた。

現代の暗号技術の特徴

現代の暗号技術は、基本的には次のような技術要素から構成されている。

- コンピューターを使って攻撃しても、キーなしでは解読が事実上困難な暗号化方式
- 暗号を解読するのに必要なキーの交換を、安全ではない通信路で行う必要を無くした「公開キー暗号」
- メッセージの真正性を保証する「電子署名」

計算複雑性理論と暗号技術

xの値が与えられた時、f(x)の計算は容易であるが、y=f(x)であるyの値が与えられた時、y=f(x)を満たすxを見つけるのが難しい時、fを「一方向関数」と呼ぶ。

現代の暗号技術は、暗号化に、この一方向関数の性質を利用する。

先の「f(x)の計算は容易である」ということを「y=f(x)が正しい時、そのチェックは多項式時間で可能である」と解釈すると、暗号を解くことは、計算複雑性理論のNP問題であることがわかる。

暗号技術の現在 **Agenda**

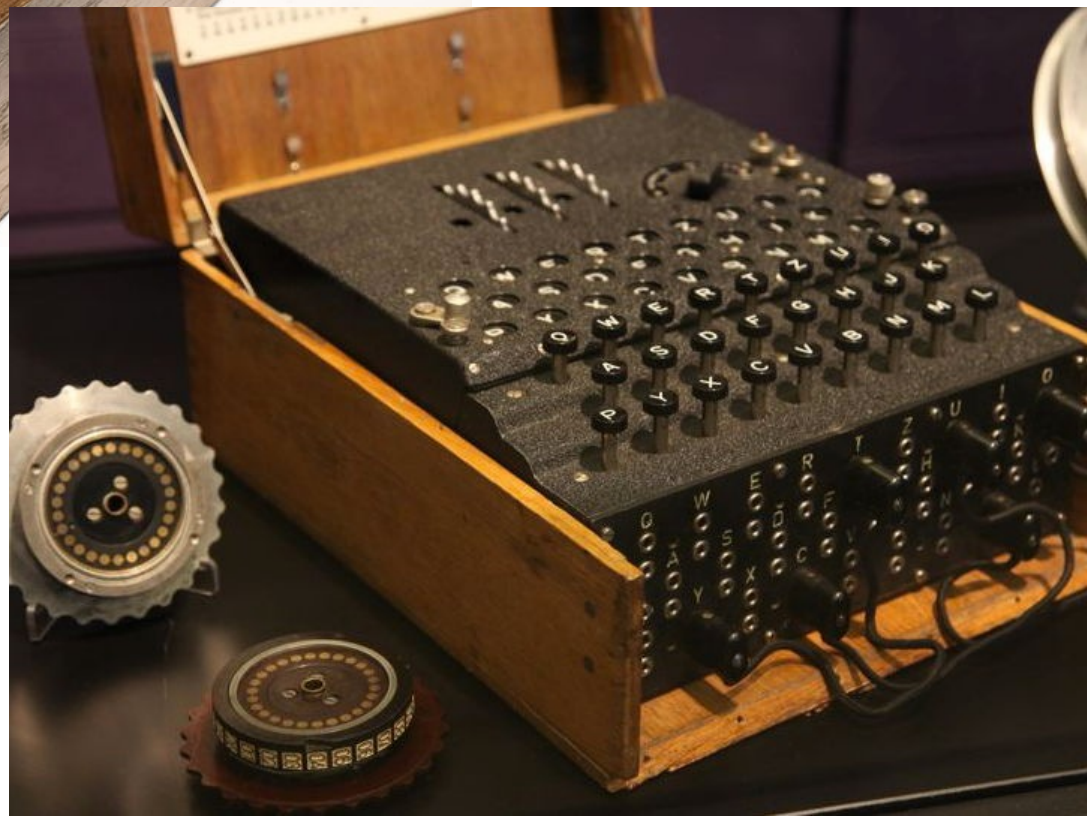
第一部 現代暗号技術の成立

- 現代暗号技術成立以前
- 現代暗号技術の成立
- 暗号と計算複雑性理論

現代暗号技術成立以前

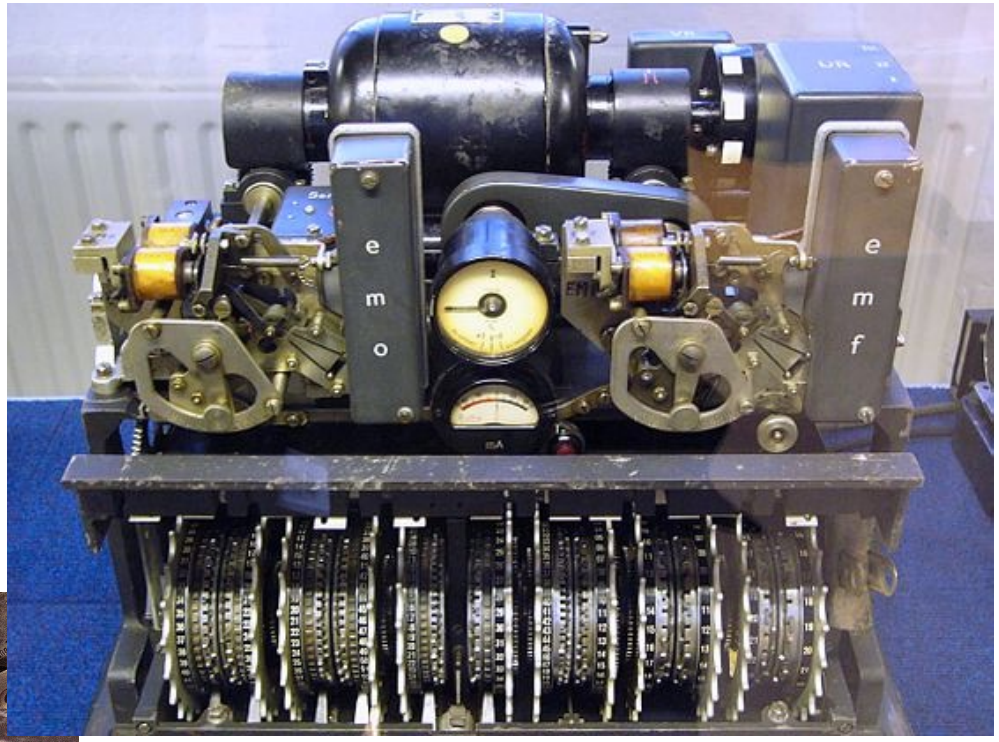
第二次大戦期の暗号技術

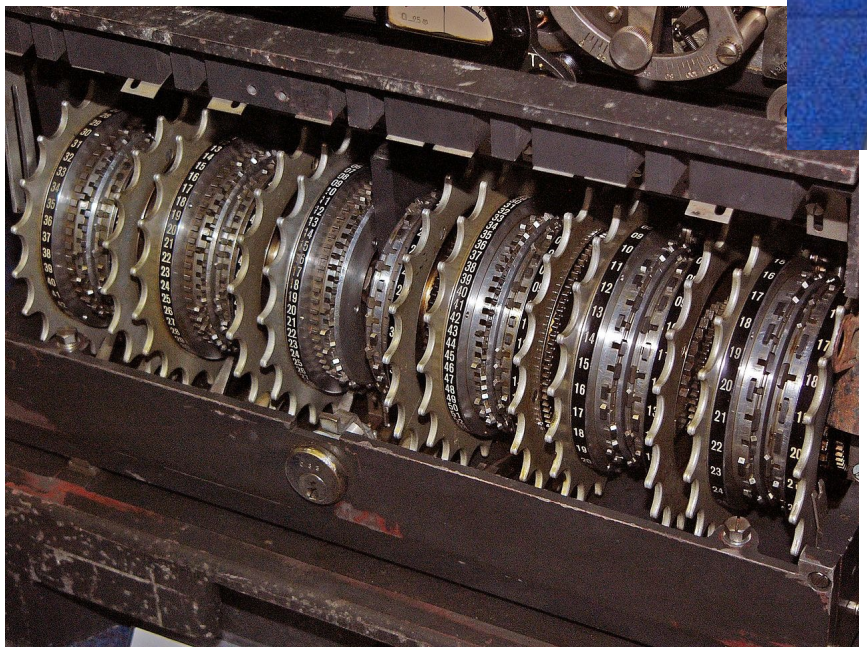
ドイツ
Enigma



プラグボード付き
Enigma

ドイツ
Lorenz SZ42

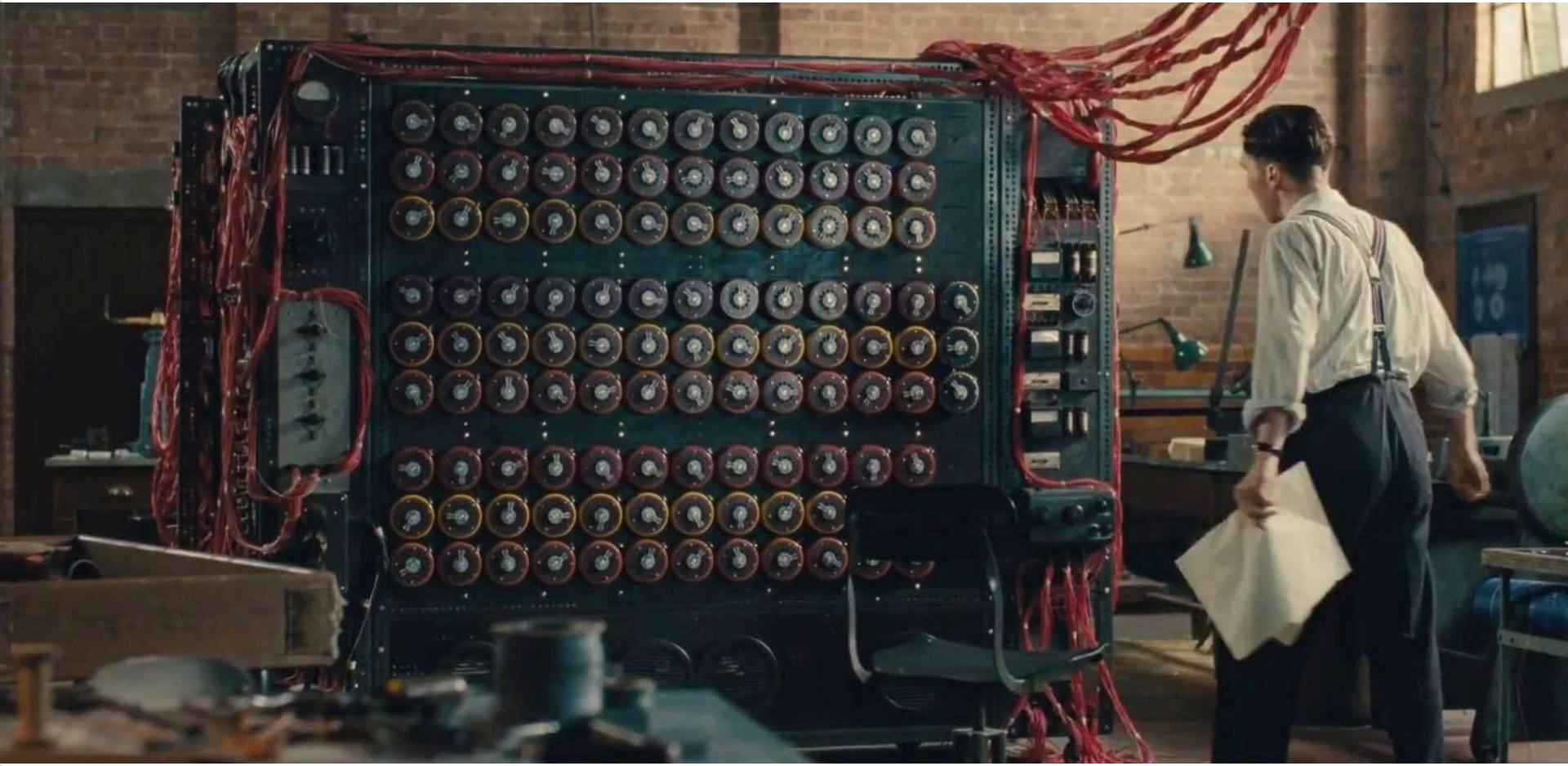




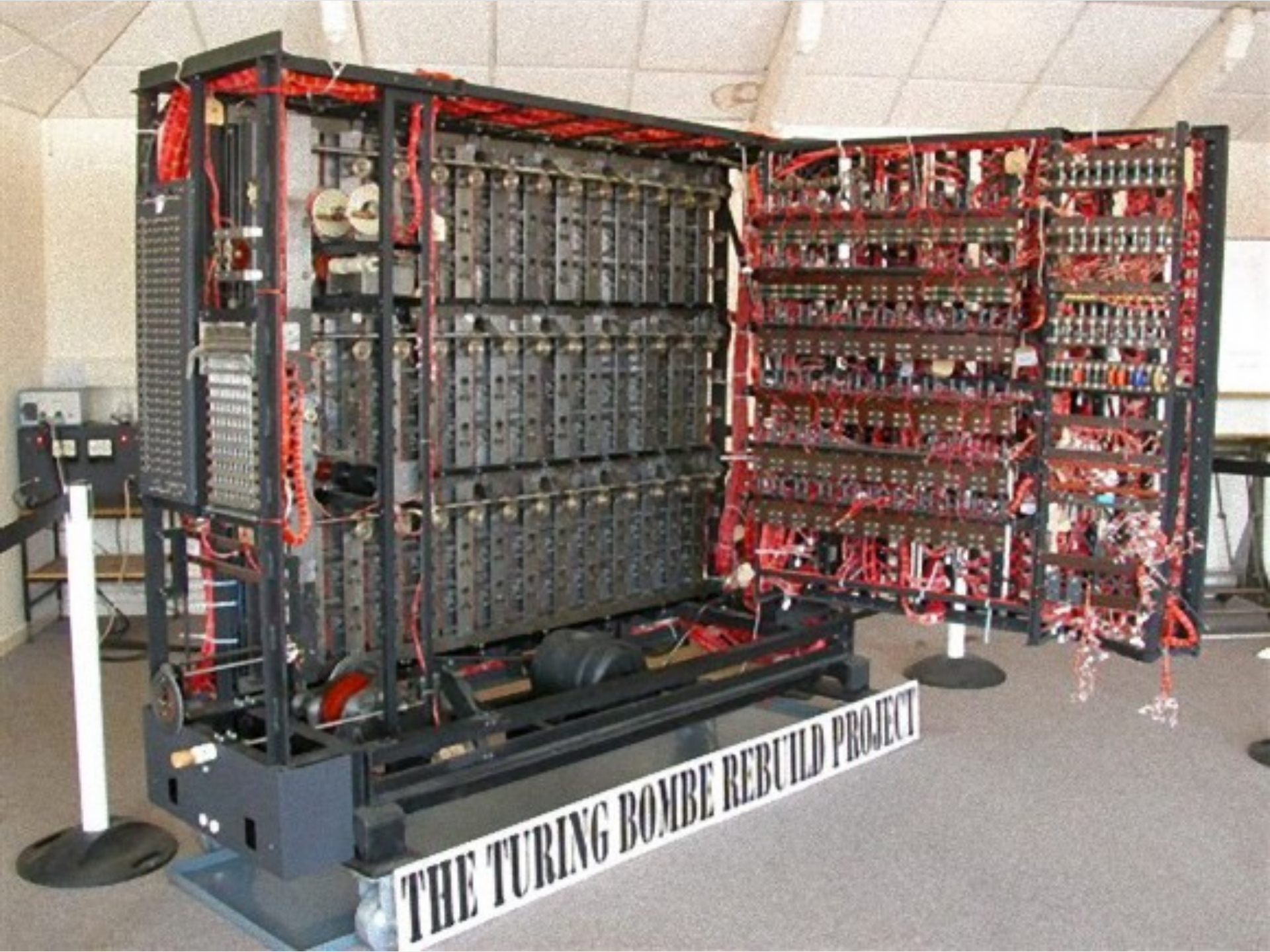
The Lorenz SZ machines had 12 wheels each with a different number of cams (or "pins").

https://en.wikipedia.org/wiki/Lorenz_cipher

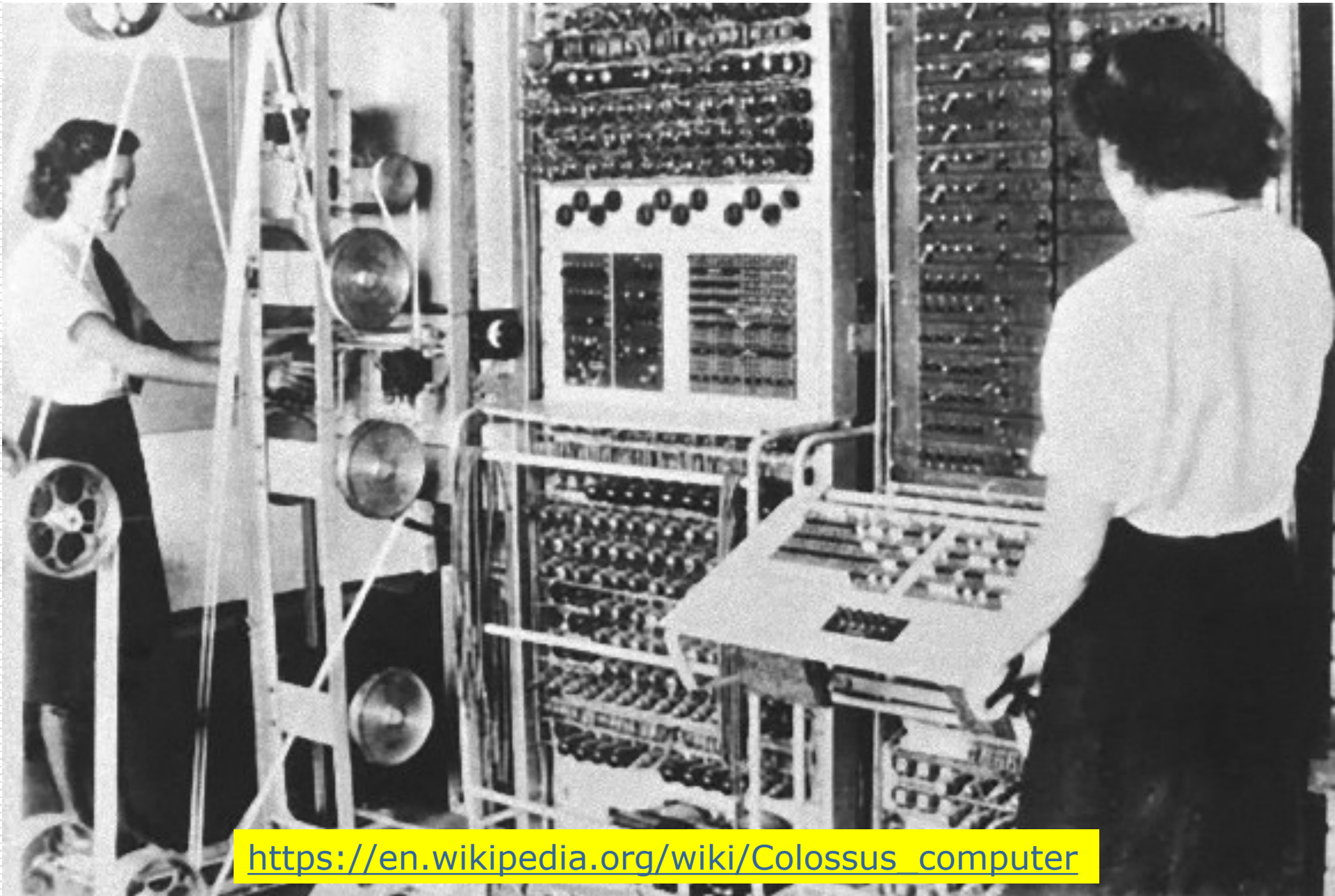
Turing Bombe (=>Enigma)



https://en.wikipedia.org/wiki/Bombe


THE TURING BOMBE REBUILD PROJECT

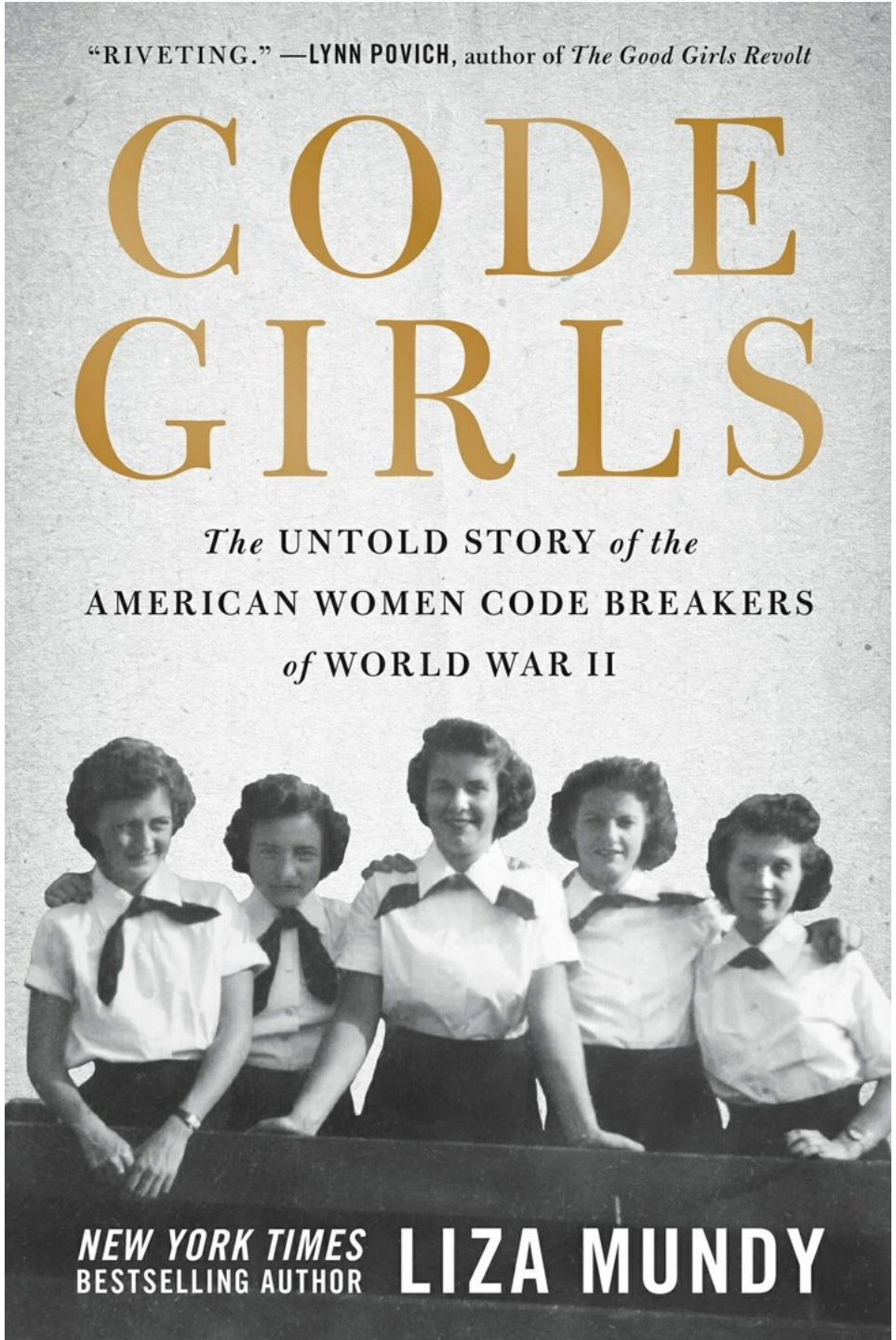
Colossus (=> Lorents SZ42)



日本　Purple暗号

Type B Cipher Machine", codenamed Purple by the United States, was a encryption machine used by the Japanese Foreign Office from February 1939 to the end of World War II.

https://en.wikipedia.org/wiki/Type_B_Cipher_Machine



Genevieve Grotjan

日本のPurple暗号を破る

https://amzn.to/2K1N23g

Code Talker

“コードトーカー(Code talker)とは、
国外にはその言葉を解するものがいない
固有の部族語をコード（暗号）として
前線での無線通信を行うため、アメリカ
軍が使用したアメリカインディアン部族
出身の暗号通信兵である。”

"Were it not for the Navajos, the Marines would never have taken Iwo Jima."

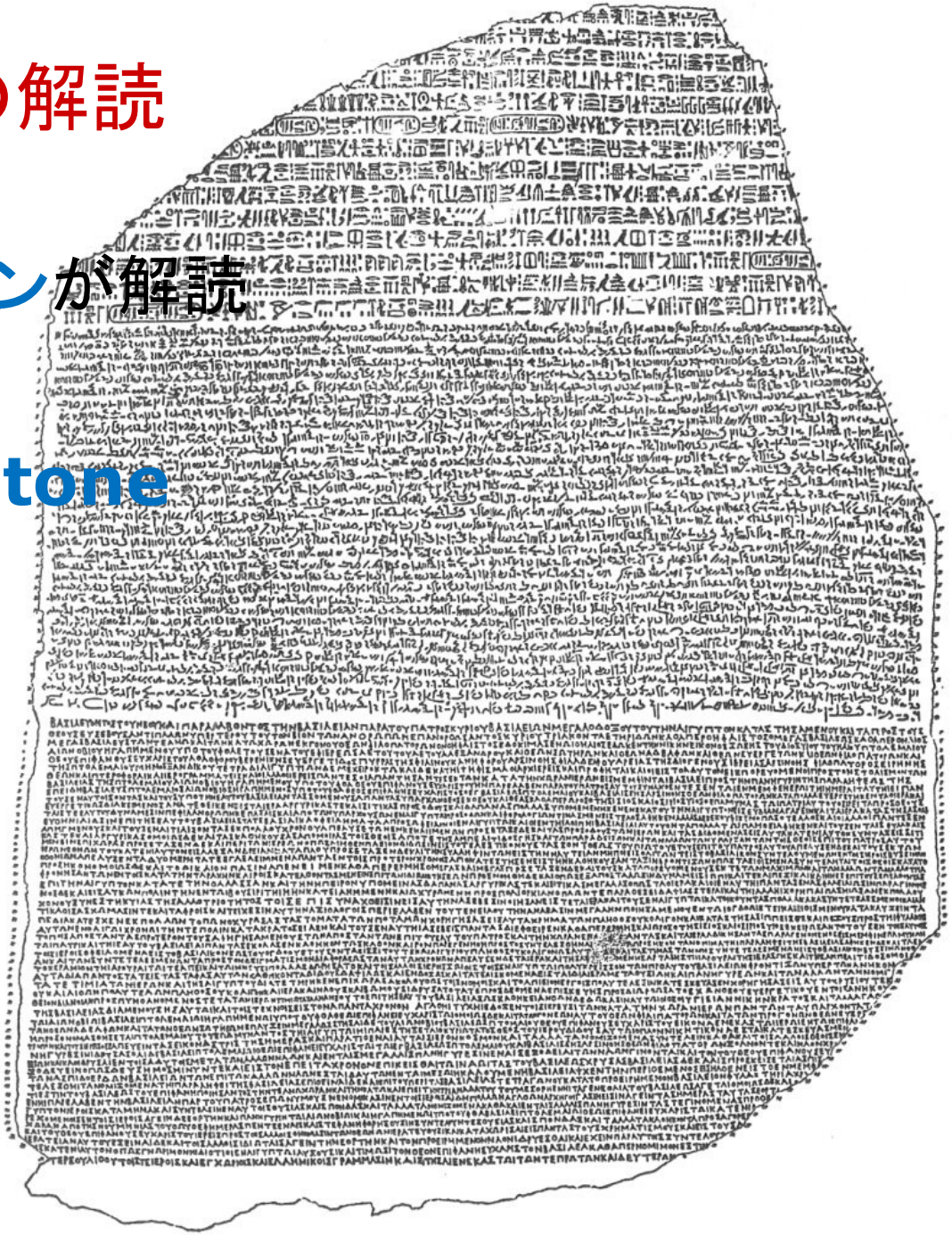
言語と暗号

未知の文字/言語の解読

象形文字の解読

1822年
シャンポリオンが解読

Rosetta Stone
BC 196年



神聖文字

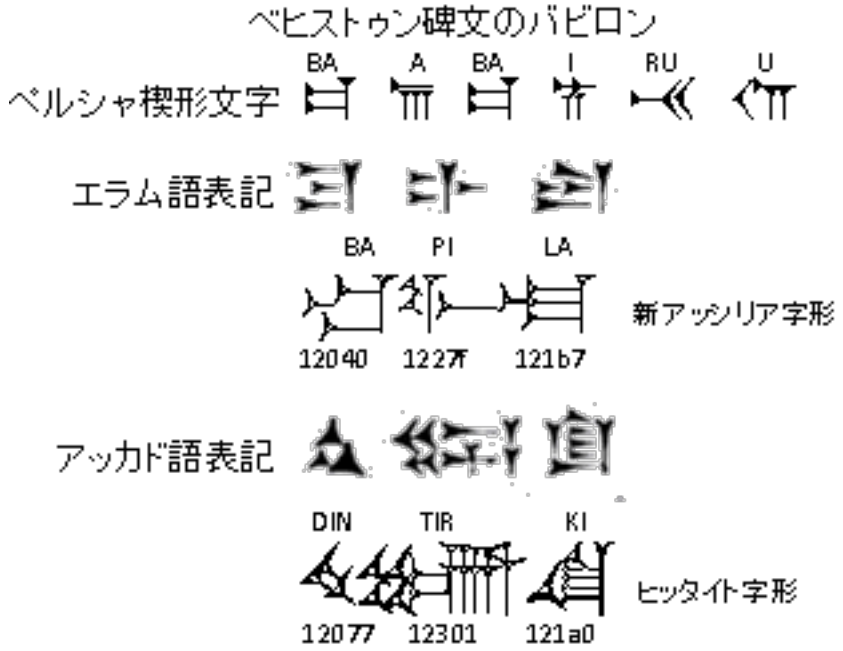
民衆文字

ギリシャ
文字

楔形文字の解読

ベヒストゥン碑文

- コーパス： ベヒストゥン碑文　BC522~
  （エラム語、古代ペルシア語、アッカド語の三つの言語で書かれている）
  解読者：ローリンソンとヒンクス
  1846-1851



線文字Bの解読

- 紀元前1450年から紀元前1375年頃までミュケナイ時代に、ギリシャ本土からエーゲ海諸島の王宮で用いられていた文字。
- 発見者であるイギリスの考古学者アーサー・エヴァンズにより線文字Bと命名された。
- 1953年、イギリスの建築家マイケル・ヴェントリスと言語学者ジョン・チャドウィックによりギリシア語として解読された。

https://goo.gl/4SZhkn

Phaistos diskの解読　2014年

- でも、それとは真逆の取り組みも存在する。クレタのGareth Owensは、たった一つの粘土板にきざまれた、45種類の「文字」で書かれた241文字の「文」の解読に成功したという。
  https://goo.gl/4Ye6Be

理論的に先行したもの

Nash:「NSAへの手紙」 1955年
Gödel:「フォン･ノイマンへの手紙」 1956年

1950年代に入ると、時代の先鋭な知性たちは、今日の複雑性理論の原型とも言える認識に到達し始める。

スコット・アーロンソンは、複雑性理論の到達点を概観した論文 "N = NP?" https://goo.gl/TZUkgh の冒頭に、ジョン・ナッシュの 1955年のNSA（スノーデンがいた、あのSNAだ！）宛の手紙をあげている。かつては「機密文書」とされていて、いつかの時点で情報公開されたものらしい。

誰にも破れない暗号を作るためには、指数関数的な計算が必要な暗号を作ればいいことを述べている。また、そうすれば、それまでの職人技的な暗号破り（多分、チューリングやファインマンがしていたこと）は「過去」のものになると、明確に自覚している！

ジョン・ナッシュ：1955年のNSA宛の手紙

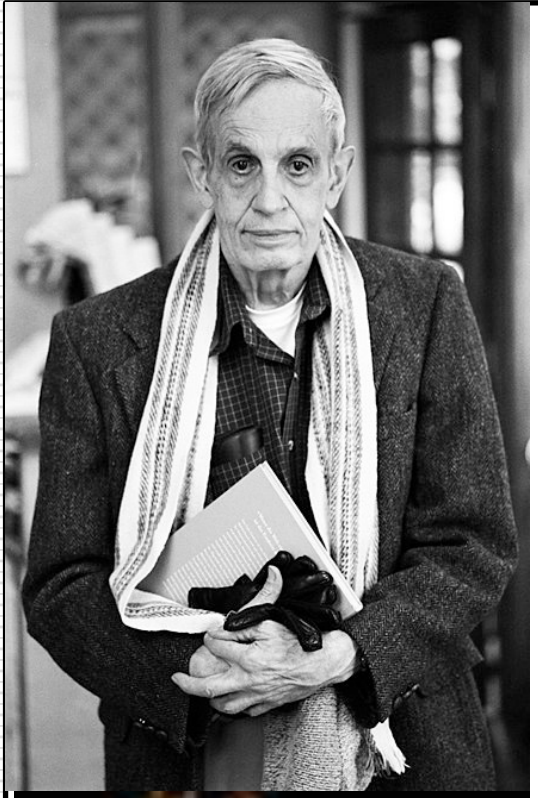
現時点での、私の一般的な予想は次のようなものです。:

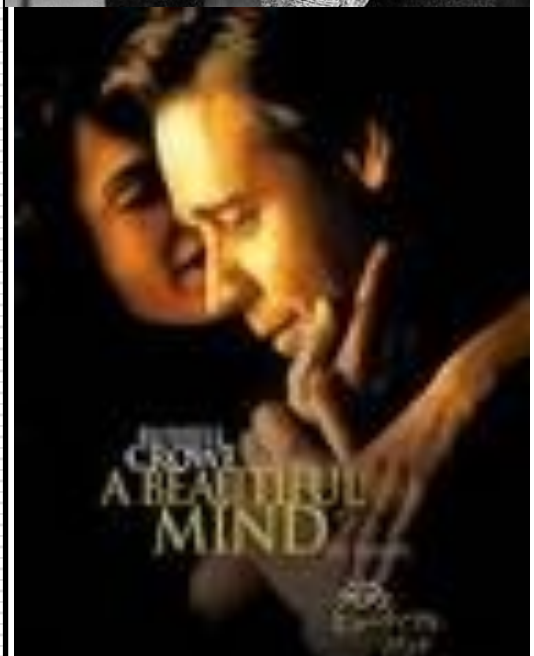
ほとんどすべての十分に複雑なタイプの暗号化にとって、特に、鍵の異なる部分によって与えられた命令が、相互に複雑に相互作用して、それが暗号化の最終的な決定において影響を与えている場合、鍵の計算の平均的な長さは、鍵の長さ、すなわち、鍵の持つ情報の内容に関して、指数関数的に増加します。

もし、この予想が正しいと仮定すれば、この一般的な予想の重要性を理解するのは容易です。それは、実際的には誰も破れない暗号を設計することを、極めて簡単に実行できることを意味します。暗号が洗練されたものになるにつれて、熟練したチームなどによる暗号破りのゲームは、「過去」のものになっていくはずです。

この予想の性質は、たとえ特別な種類の暗号をとっても、私にはそれを証明できないようなものです。また、私は、それが証明されることも期待していません。

https://www.nsa.gov/…/nash-l…/assets/files/nash_letters1.pdf





Now my general conjecture is as follows: For almost all sufficiently complex types of enciphering, especially where the ~~information~~ instructions given by different portions of the key interact complexly with each other in the determination of their ultimate effects on the ~~each~~ enciphering, the mean key computation length increases exponentially with the length of the key, or in other words, with the information content of the key.

The significance of this general conjecture, assuming its truth, is easy to see. It means that it is quite feasible to design ciphers that are effectively unbreakable. ~~and~~ As ciphers become more sophisticated the game of cipher breaking by skilled teams, etc., should become a thing of the past.

The nature of this conjecture is such that I cannot prove it, even for a special type of cipher. Nor do I expect it to be proven. But this

ナッシュの慧眼：計算の複雑性と暗号

「鍵の計算の平均的な長さは、鍵の長さ、すなわち、鍵の持つ情報の内容に関して、指数関数的に増加します。」

「それは、実際的には誰も破れない暗号を設計することを、極めて簡単に実行できることを意味します。」

暗号理論として：

鍵を破るのに、「指数関数的」時間がかかる暗号は、事実上、誰にも破れない。もちろん、正しい受け取り手が暗号を解読するのに「指数関数的時間」がかかってはいけない。

計算複雑性理論の萌芽として：

「指数関数的時間」とそうではない時間（「多項式時間」という）の区別

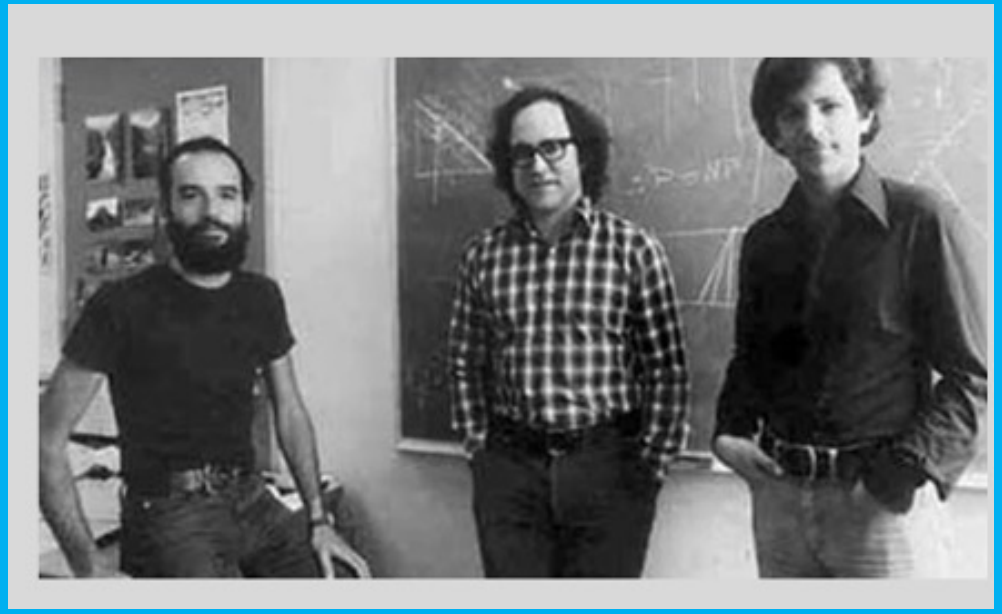
現代暗号技術の成立

公開キー暗号　1976年

Whitfield Diffie,
Martin Hellman

RSA暗号　1977年

Ron Rivest,
Adi Shamir,
Leonard Adleman



素因数分解の難しさと暗号への応用

ある数Nが、二つの素数p,qの積で表されるとしよう。

例えば、N=99400891の場合、Nは素数 9967と9973の積である。

難しい

N = p x q

99400891 = 9967 x 9973

易しい

この時、Nの素因数を求める計算は難しく、

p, qを掛けてNになることの検証はやさしい。

非常に大きな数Nを取ると、Nを公開したとしても、その素因数p, q を知ることは困難である。

例えば、次の220桁の数の素因数を求めるのは難しいRSA Factoring Challenge(2007年に終了)から

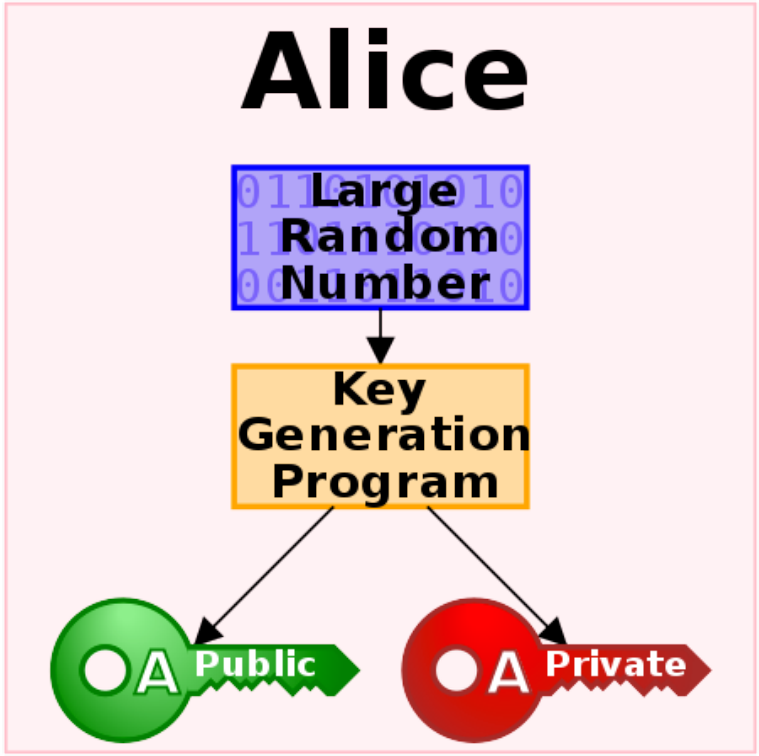
RSA-220 =
2260138526203405784941654048610197513508038915719776718321197768109445641817966676085931213065825772506315628866769704480700018111497118630021124879281994874820660701310665866460833279828035603792053919801399464969552 61


難しい


易しい

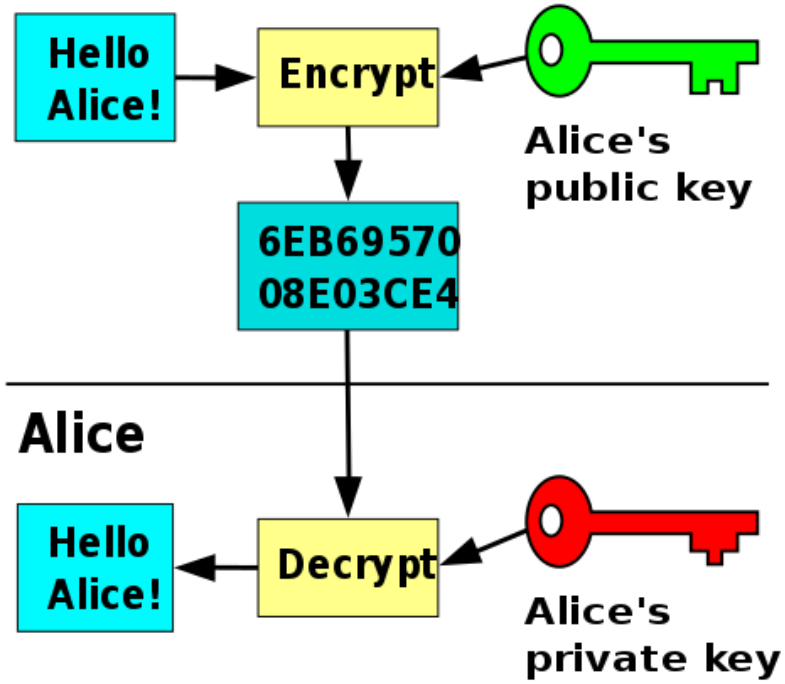
RSA-220 =
6863656412267566274382371499288437800130842239979164844621244993321541061441464266793821364420842019205499968 7 ×
3292907439486349812049301549212935291916455196536233952462686051169290349309465246333782486639073819176571260 3

公開キー暗号1976年
Whitfield Diffie, Martin Hellman



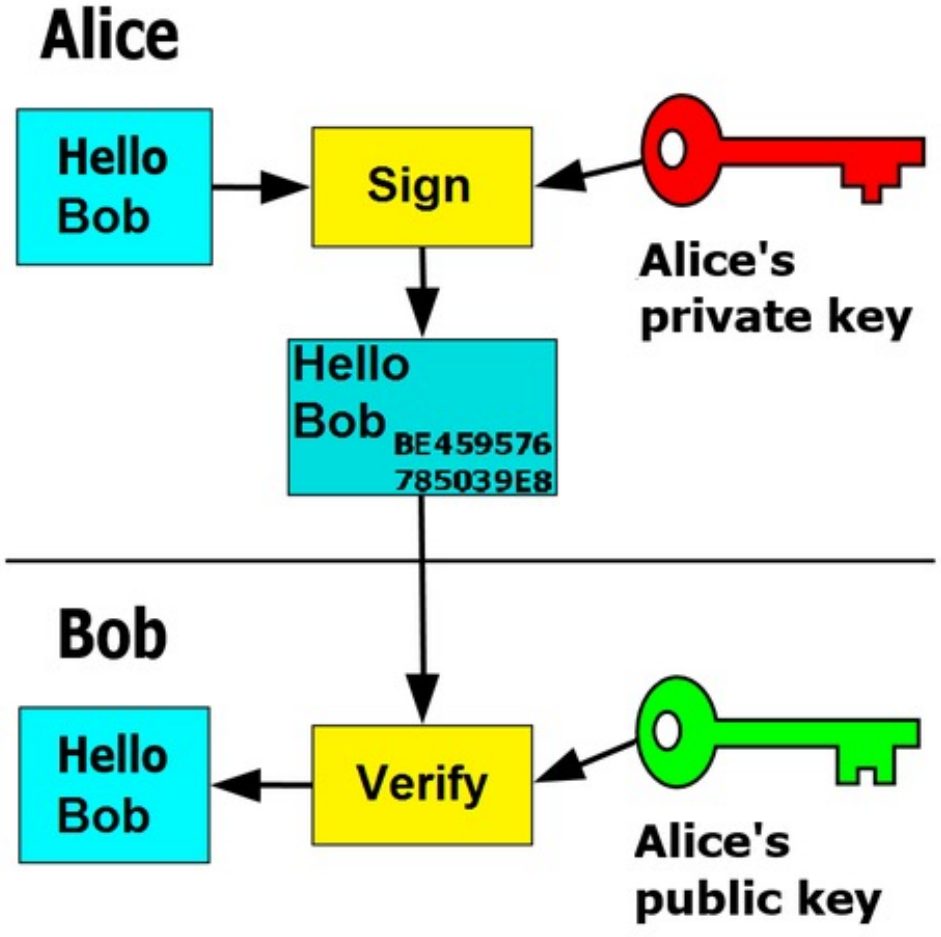
Aliceは、公開キーと秘密キーの二つのペアを生成する。

BobはAliceへのメッセージをAliceの公開キーを使って暗号化する。



Aliceは、Bobからのメッセージを自分の秘密キーを用いて解読する。

電子署名　Digital signatures



Aliceは、自分の送ったメッセージが真正のものであることを示す為に、自分の秘密キーを使って、署名を行う。ハッシュ関数。

Bobは、Aliceからのメッセージが真正であることを確かめる為に、Aliceの公開キーを用いて、署名を検証する

RSA暗号　1977年
Ron Rivest, Adi Shamir, Leonard Adleman

- 大きな桁数の合成数の素因数分解が困難なことを利用。
- まず、適当な正整数 e（通常は小さな数。65537 (= 2^{16} + 1) がよく使われる）を選択する。また、大きな2つの素数 $\{p, q\}$ を生成し、それらの積 n ($=pq$) を求めて、$\{e, n\}$ を平文の暗号化に使用する鍵（公開鍵）とする。
- 2つの素数 {p, q} は、暗号文の復号に使用する鍵（秘密鍵）d の生成にも使用し 秘密に保管する。

$$d = e^{-1} \pmod{(p-1)(q-1)}$$

機密にされた発見

RSA暗号/公開キー暗号には先行者がいた

1970-1974 Ellis-Cocks-Williamson

In 1970, James H. Ellis, a British cryptographer at the UK Government Communications Headquarters(GCHQ), conceived of the possibility of "non-secret encryption", (now called public key cryptography), but could see no way to implement it.[9] In 1973, his colleague Clifford Cocks implemented what has become known as the RSA encryption algorithm, giving a practical method of "non-secret encryption", and in 1974, another GCHQ mathematician and cryptographer, Malcolm J. Williamson, developed what is now known as Diffie–Hellman key exchange. The scheme was also passed to the USA's National Security Agency.[10] With a military focus and low computing power, the power of public key cryptography was unrealised in both organisations:

https://en.wikipedia.org/wiki/Public-key_cryptography

GCHQ pioneers on birth of public key crypto

Two men who played a role in developing public key cryptography, Clifford Cocks and Ralph Benjamin, talk to ZDNet UK about the invention of the important encryption technique

Public key cryptography is widely used to secure online transactions. The maths behind the technology was invented by UK Government Communications Headquarters scientists in the late 1960s and early 1970s.

The discovery was kept secret to avoid revealing how closely Government Communications Headquarters (GCHQ) was working with the US National Security Agency (NSA) at the time. The breakthrough by GCHQ scientists James Ellis, Clifford Cocks and Matthew Williamson only came to light in 1997, when their work was declassified.

https://zd.net/2wwmaAI

暗号と計算複雑性理論

「計算の難しさ」とは何か？

現代の暗号化技術の特徴

現代の暗号化技術の大きな特徴は、その基礎を、コンピュータはある種の数学的問題を、「効率的」には解くことができないという（経験的）事実においていることである。

RSA暗号は、素因数分解問題を、現在のコンピュータでは効率的に計算できないことを基礎にし、楕円曲線暗号は、離散対数問題を効率的に解けないことを基礎としている。

その理論的基礎は、数学的には、「計算複雑性の理論」である。

複雑性理論入門

複雑性理論

「計算の難しさ」を計算に要する「時間」と「メモリー」で評価する

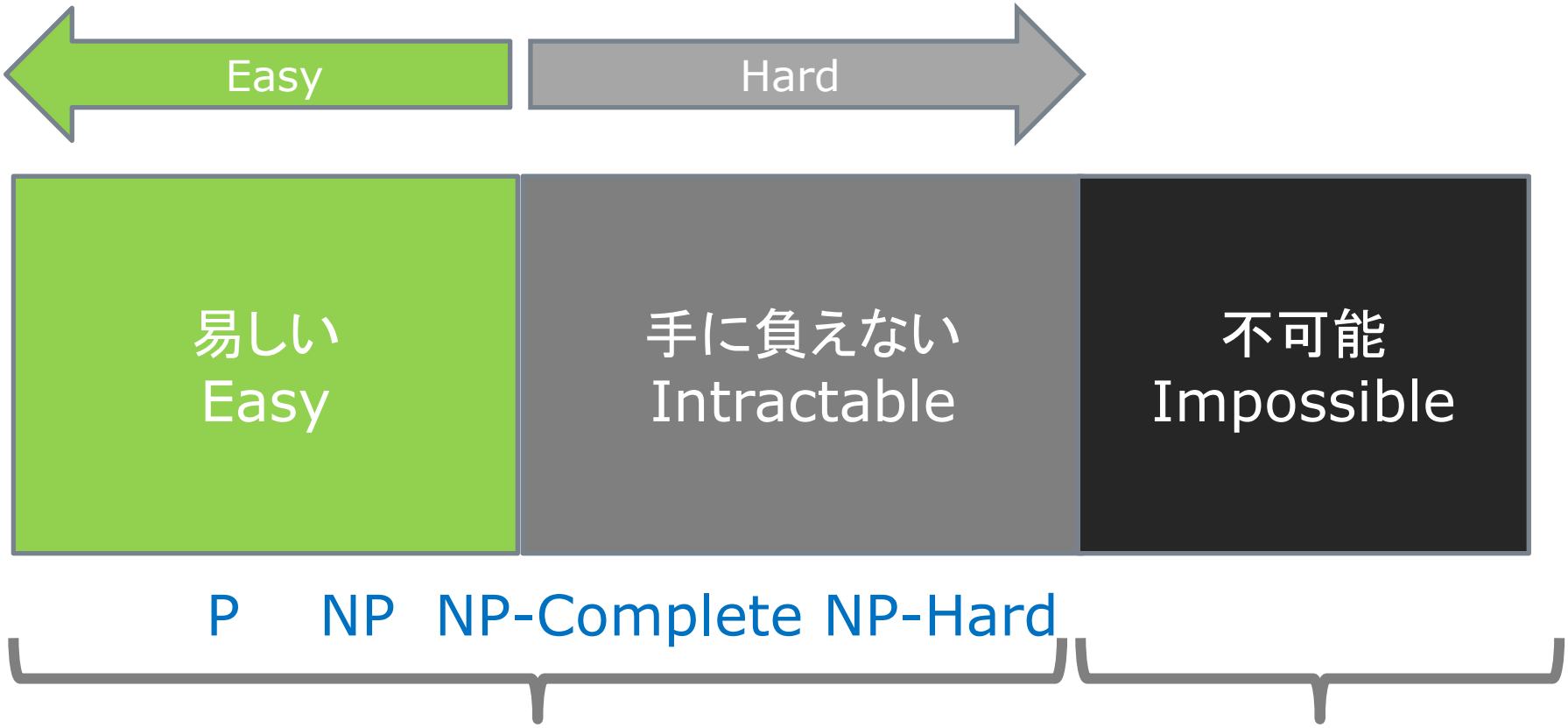
ゲーデル、チューリングらが明らかにしたように、「証明可能＝計算可能」なものには限界があることは、1950年代には、一般に認められるようになった。

ただ、それは、我々の認識の「限界」としては、非常に荒い、かつ、抽象的で原理的な限界を与えるものでしかなかった。

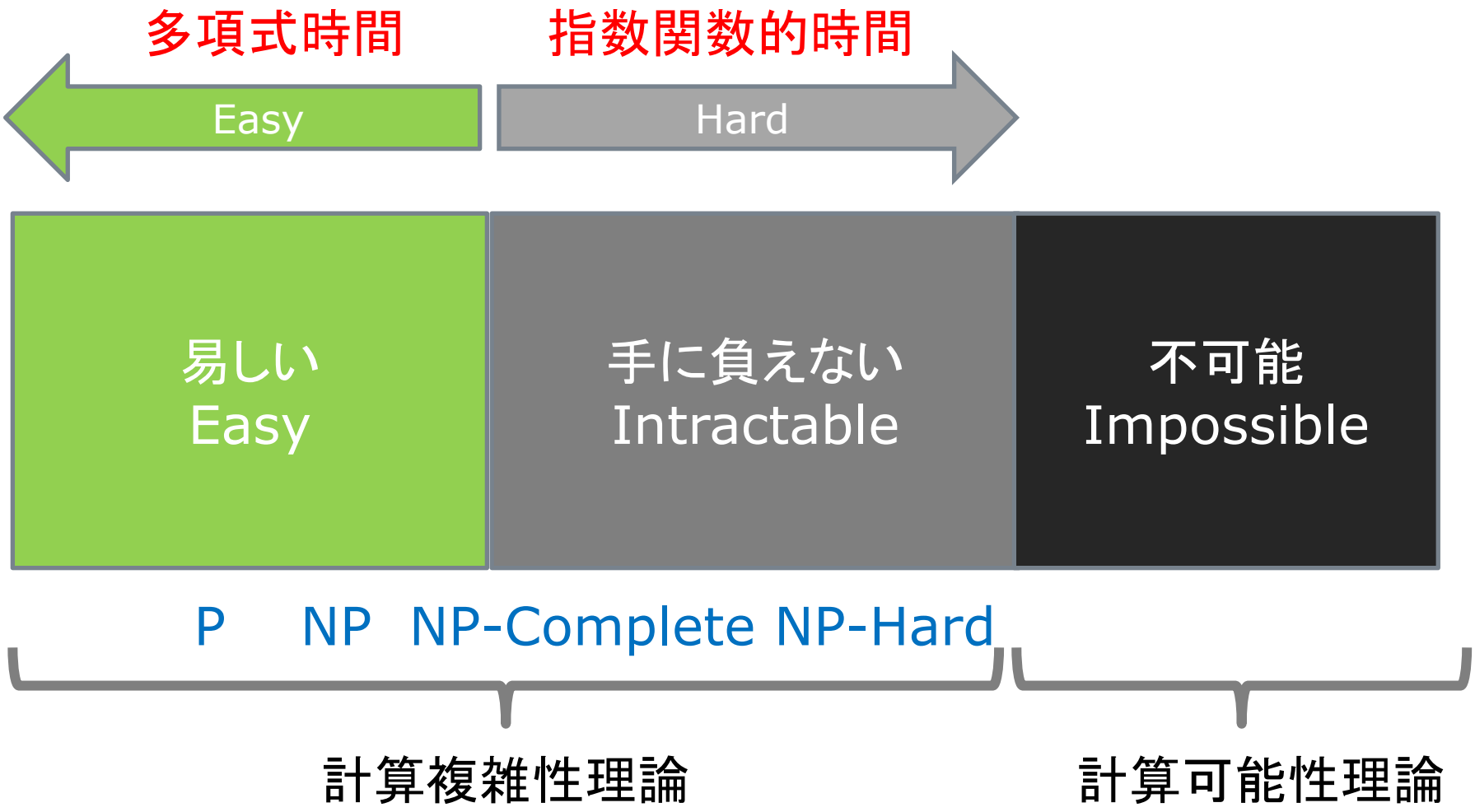
70年代に入ると、「計算の難しさ」に対する新しいアプローチが活発になる。それは、「計算の難しさ」を、計算に要する「時間」と「メモリー」で評価しようというものである。それを「計算複雑性理論」という。

認識の易しさと難しさ

計算の易しさと難しさ


Easy
Hard
易しい
Easy
手に負えない
Intractable
不可能
Impossible
P
NP
NP-Complete
NP-Hard
計算複雑性理論
計算可能性理論

計算の易しさと難しさ


多項式時間
指数関数的時間
Easy
Hard
易しい
Easy
手に負えない
Intractable
不可能
Impossible
P NP NP-Complete NP-Hard
計算複雑性理論
計算可能性理論

「多項式時間」と「指数関数的時間」

計算時間について言えば、このアプローチで重要なのは、「多項式時間」と「指数関数的時間」を区別することだ。

多項式の例：

$$n^3+10n^2+8n+1$$

$$5n^4+9n^3+7n^2+3n+5$$

指数関数の例：

$$3^n+5^n$$

$$2^{2^{2^n}}+3^n+5$$

その上で、計算時間が「多項式時間」で表される計算を「やさしい計算」、指数関数で表される計算を「難しい計算」と考えるのだ。

P問題

入力の文字列の長さをnとするある問題が、nの「多項式時間」で解ける時、この問題を「P問題」という。

「P問題」は、「やさしい計算」問題と考えていい。

計算複雑性の理論で、「P問題」と並んで重要なクラスに「NP問題」がある。「NP問題」は、解くのに「指数関数的時間」がかかる「Not-P問題」ではない。それについては、次に説明する。それでも、

「NP問題」は、「難しい計算問題」と考えていい。

NP問題

その問題を解くのが「多項式時間」で終わるかどうかはわからなくても、具体的な値で与えられたその問題の「答え」と言われるものが、本当に「正しい答え」であるかどうかを「多項式時間」でチェックできる問題のクラスを、NPという。

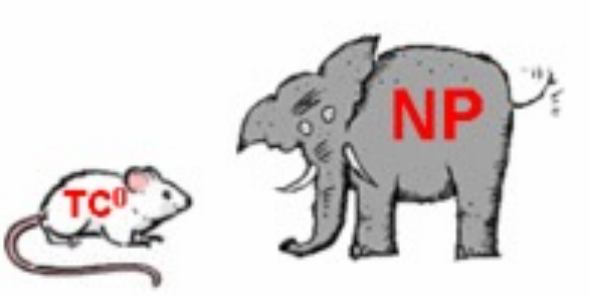
NPクラスの説明によく用いられるのは、素因数への分解問題である。

先に見たように、Nは二つの素数の積であることがわかっていたとしても、Nが大きな数になると、Nを素因数に分解すること、すなわち、N＝pqであるような二つの素数 p, q を見つけることは難しくなる。

ただ、N, p, q が具体的に与えられているなら、「Nは、素数pと素数qに、素因数分解される」という主張は、簡単に検証できる。pとqを掛けて、その結果がNに等しいことを示せばいいのだから。掛け算は、もちろん簡単な計算で、多項式時間で終わる。

素因数分解問題は、NP問題である。

P=NP?問題



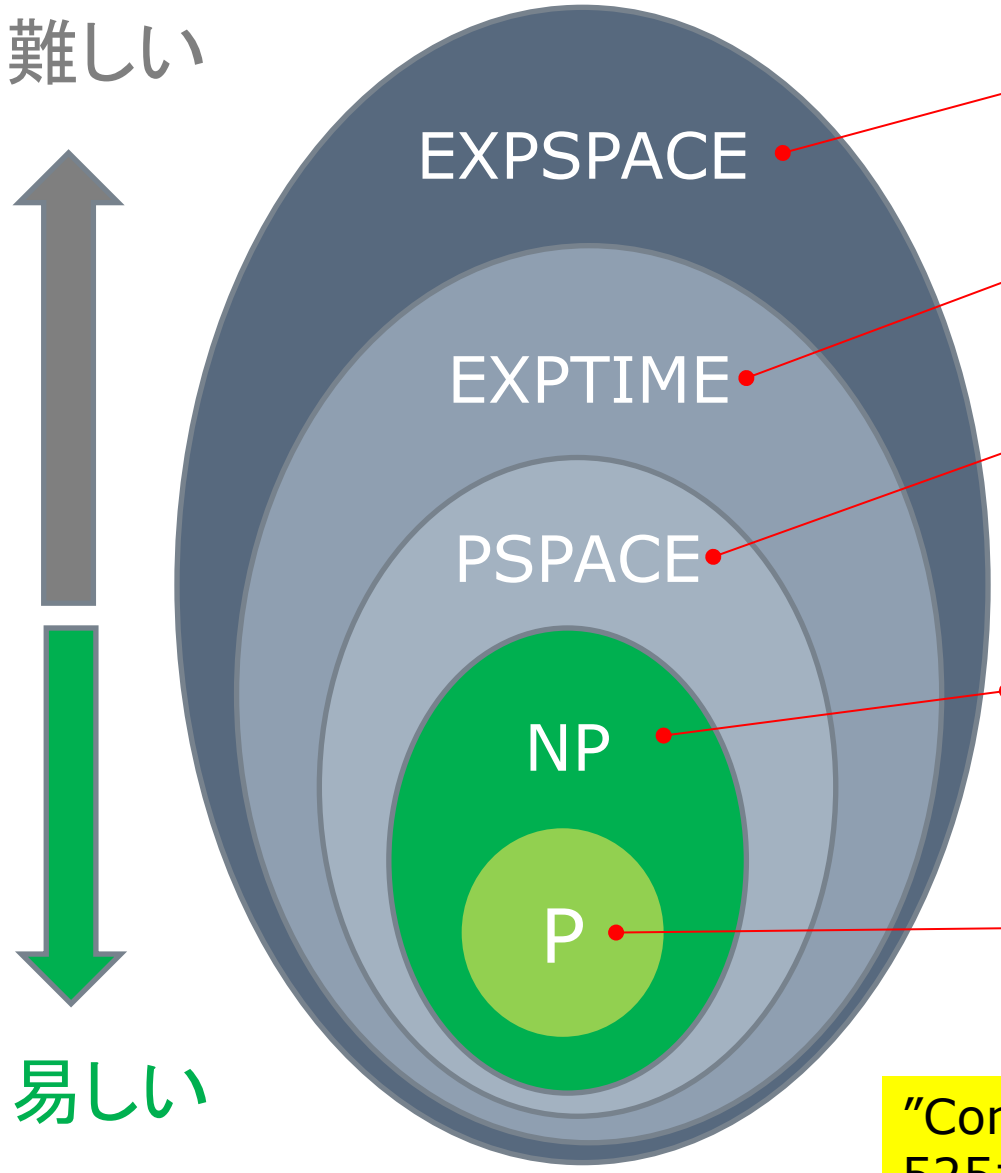
「計算複雑性理論」の中心問題は、このPとNPの関係である。
もしも、PとNPが実際には等しいクラスであることが証明できれば、見かけは「むずかしい計算問題」に見えるNP問題が、実は「やさしい計算」で解けることになる。

多くの数学者は、PとNPは、等しくないと考えている。
ただ、この「P=NP?問題」は、現在に至るも解決されていない。

この問題は、現代数学上の未解決問題の中でも最も重要な問題の一つであり、2000年にクレイ数学研究所のミレニアム懸賞問題の一つとして、この問題に対して100万ドルの懸賞金がかけられている。

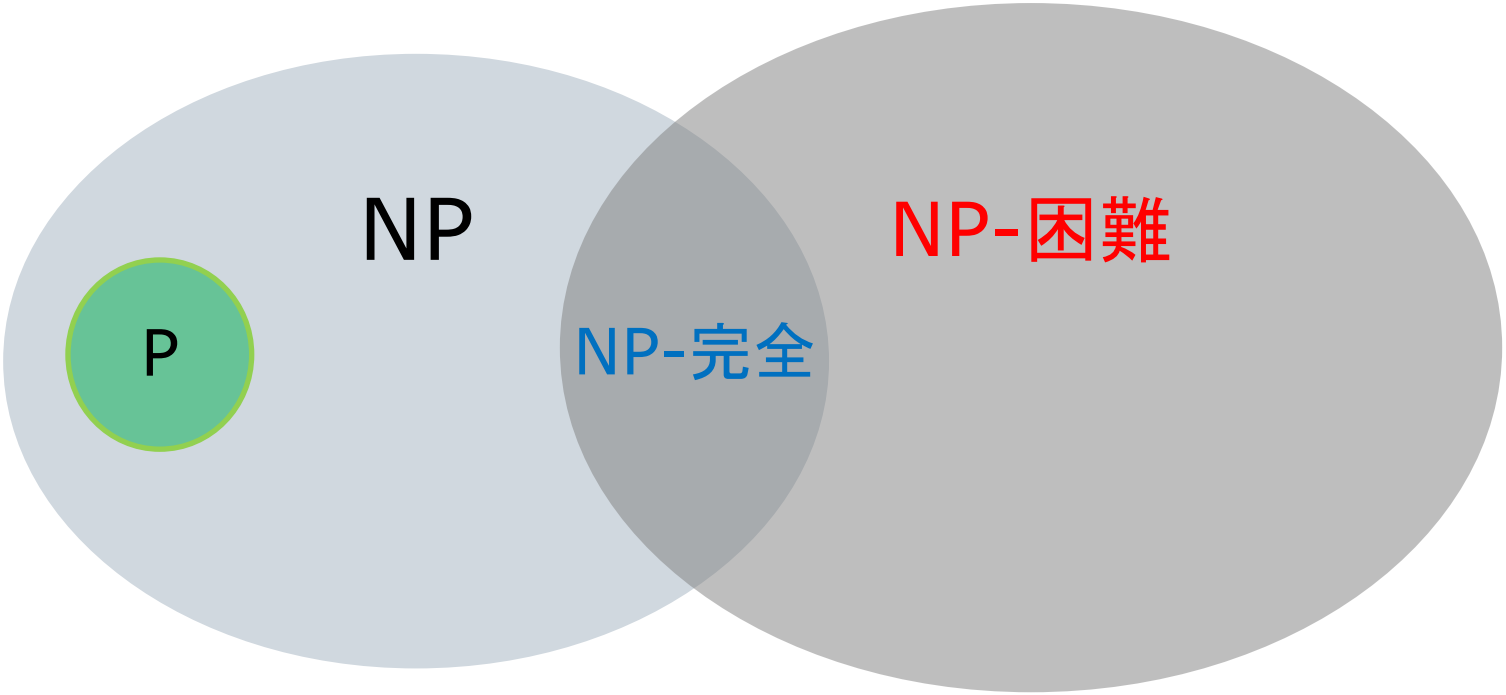
現在のようなスタイルで、N = NP? 問題を最初に定式化したのは、バークレーのStephen Cookである。1967年のことだ。

代表的な複雑性クラスの階層



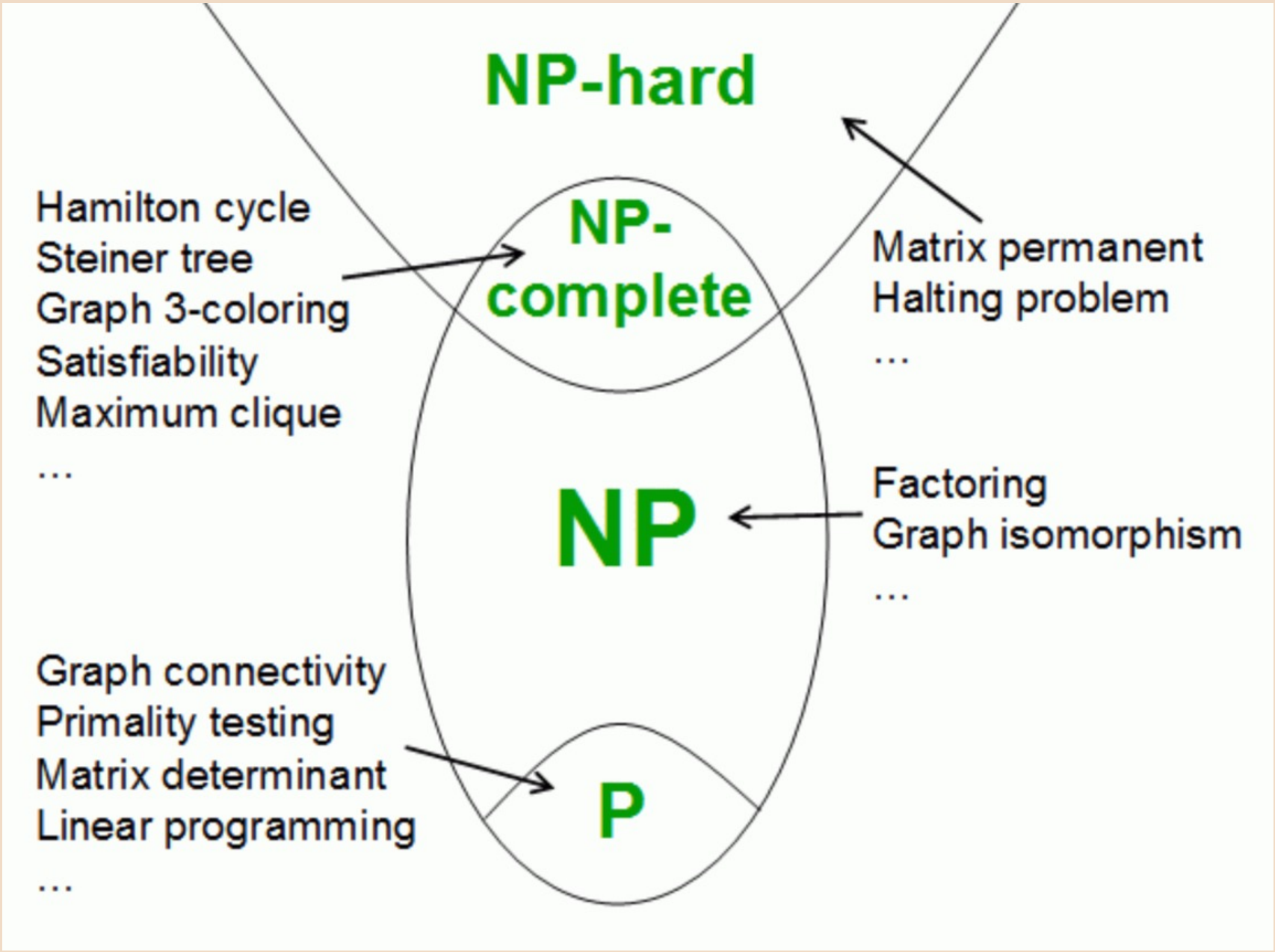
- 指数関数的なメモリーを使用
- 指数関数的な計算時間
- 多項式関数的なメモリーを使用
- 答えが正しいかは多項式時間でチェックできる問題
- 多項式時間で計算できる問題

”Complexity Zoo” http://bit.ly/30FlQxo
525もの複雑性のクラスが収録されている！



P:　多項式時間で解ける問題
NP: 多項式時間で解けるとは限らないが、
　　答えが正しいかは多項式時間でチェックできる問題

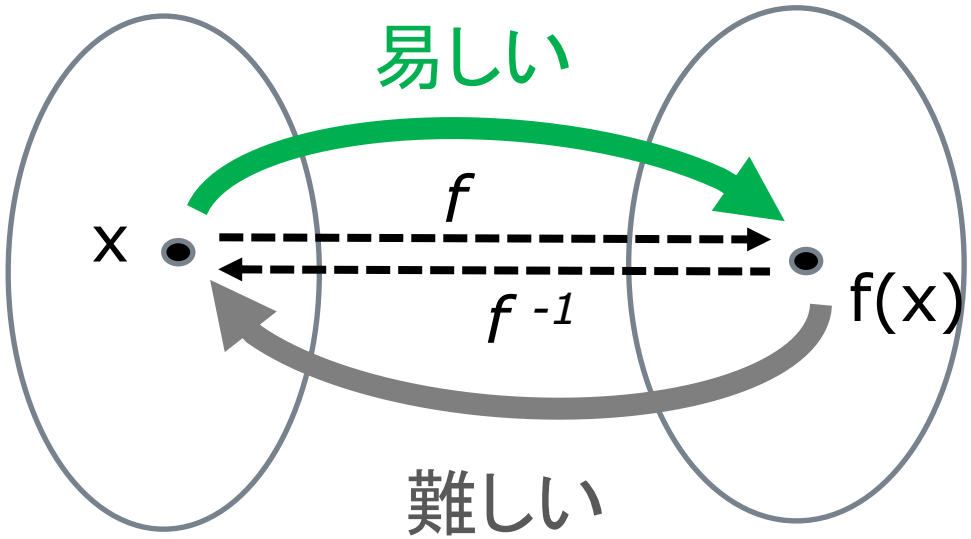
NP-困難: 全てのNP問題が、このクラスの問題に帰着される問題
　　（この帰着は多項式時間で行われなければいけない）
NP-完全: NP問題で、かつ、NP-困難な問題


NP-hard
NP-complete
NP
P
Hamilton cycle
Steiner tree
Graph 3-coloring
Satisfiability
Maximum clique
…
Matrix permanent
Halting problem
…
Factoring
Graph isomorphism
…
Graph connectivity
Primality testing
Matrix determinant
Linear programming
…

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。
  - こうした要請に応える一方向関数の「難しさ」の定義があるのだが、それについてはのちに簡単に触れる。
- 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　One Way Function 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。
  例えば、e=3,x=4,p=17とすれば、$3^4=81$だから、それを17で割った余りは13となるので、$3^4 \equiv 13 \pmod{17}$、よって
  $4 \equiv \log_3 13 \pmod{17}$となる。
  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。


© NASA/ESA/CSA/Judy Schmidt


第二部　Shorのアルゴリズムの発見

第二部　Shorのアルゴリズムの発見

概説

-- 「量子の世紀」の先駆け --

Shorのアルゴリズムを実行する量子コンピューターは、まだ、存在しない

この第二部では、現代の暗号技術の動向に、決定的な影響を与えた「Shorのアルゴリズムの発見」を取り上げる。

最初の節で、Shorの発見についての典型的な反応を取り上げる。それらは、非現実的な過大な期待と過小評価の現実論がジグザグに交差するものだ。なぜ、一つの理論的発見の評価をめぐって、こうした激しい振幅が起きるのか？

おそらく、その最大の理由はShorのアルゴリズムを実行する「量子コンピューター」は、いまだ存在しないからだと思う。

Shorのもう一つの「功績」

ただ、「暗号解読」という窓口からだけ、Shorの発見を評価するのは、狭い見方だと思う。

確かに、Shor自身、そのことを強く訴求したのは事実である。ただ、それによってFeynmanら当時の最先端の物理学者の関心事でしかなかった「量子コンピューター」に対する関心を大衆的なものに一挙に拡大した。

それ以上に重要なのは、Shorの発見は、多くの研究者に「量子コンピューター」の潜在的可能性とその研究の魅力に目覚めさせたことである。「量子コンピュータ」の研究者は大幅に増大した。それは、Shorのもう一つの大きな功績である。

Shorのアルゴリズムは
「量子の世紀」の先駆け

暗号解読への応用という現実的見方をすこし離れてみると、むしろ、Shorの発見の素晴らしさは際立つ。ちなみに、数年前ビットコインの大暴落を引き起こした、Googleの「量子優越性」の実証実験の成功は、Shorのアルゴリズムにはまったく届かない、はるかにプリミティブなものである。

量子論の現実的課題への応用について、我々が現在持つ知識や技術や未来の見通しは、ガリレオの「天文対話」のシンプリチオのように単純で貧弱なものであると思っていい。

それでも前進は続く。Shorの発見は、20世紀が達成した量子アルゴリズムの「金字塔」であり、「量子の世紀」の先駆けであり、その導きであると僕は考えている。

量子コンピューターと
現在のコンピューター技術

いくつか留意すべきことがある。

将来的には、現在のコンピューターが量子コンピューターに置き換わるというのは、おそらく間違ったビジョンだ。

いわゆる「Shorのアルゴリズム」でも、素因数分解アルゴリズムの主要部分は、普通のコンピューターが担っている。量子コンピューターが担当しているのは、その一部分だ。もっともそれが、計算の高速化には本質的にきいてくるのだが。

我々が予想できる範囲では、量子コンピューターは普通のコンピューター技術の助けを必要としている。両者の協力・共存関係は、長く続くだろう。

Quantum Parallelismの問題

量子コンピューターでは、n個の入力に対して、2^n個の計算が並列に走って、2^n個の出力が得られると考えらることがある。これを、Quantum Parallelismという。

ただ、大きな問題がある。出力されるのは2^n個の計算結果ではなく、2^n個の計算結果の状態の重ね合わせである。

問題は、この状態を観測して計算結果を取り出そうとすると、2^n個の計算結果の状態の重ね合わせは忽然と消え去って、一つの状態しか取り出せないということである。

確かに、観測しなければいい。それでは計算する意味がない。

量子複雑性の世界

Shorの発見に少し先立って、複雑性理論の理論家たちは、量子コンピューターで多項式時間で計算可能な複雑性のクラス（これをBQPと呼ぶ）は、普通のコンピューターで多項式時間で計算可能な複雑性のクラス（これをPと呼ぶ）を包摂することを発見する。量子複雑性の世界がこうして開かれる。

この発見は、量子コンピューターが普通のコンピューターにはない計算能力を持つことを、複雑性理論の言葉で語ったものだ。Shorのアルゴリズムの発見は、それを実証したものだ。

Shor自身は、量子コンピューターが、複雑性理論でいうNP-完全問題を解くことができると考えていた。ただ、そうした見通しを、現在の複雑性の研究者の大部分は、否定している。

暗号技術の現在 **Agenda**

第二部　ショアのアルゴリズムの発見

- Shorのアルゴリズムのインパクト
- 確率的素数判定
- 量子回路で「周期」を発見する
- 計算複雑性の新しいクラス BQP

Shorのアルゴリズムのインパクト

Shorのアルゴリズムの発見

1994年、Peter Shorは、量子コンピュータを利用すれば、RSA暗号の基礎である素因数分解が、多項式時間で実行できることを、理論的に証明した。

このアルゴリズムを用いれば、RSAでの素因数分解問題だけでなく、楕円曲線暗号の基礎である離散対数問題も多項式時間で解くことが示され、暗号技術の世界に大きな衝撃を与えた。

If computers that you build are quantum,
Then spies everywhere will all want 'em.
Our codes will all fail,
And they'll read our email,
Till we get crypto that's quantum, and daunt 'em.

– Jennifer and Peter Shor

量子コンピュータ実現の困難さの認識と楽観論の広がり

ただ、Shorのアルゴリズムを実行する量子コンピュータの実現は、当時の技術では、極めて困難だった。

こうして、時間と共に、Shorのアルゴリズムが現在の暗号技術への「当面の脅威」ではないという楽観論が、むしろ広く共有された。

確かに、現在に至るも、Shorのアルゴリズムを用いて、現在利用されている暗号化を破るような量子コンピュータは、いまだ存在しない。

To read our E-mail, how mean
of the spies and their quantum machine;
be comforted though,
they do not yet know
how to factorize twelve or fifteen.

– Volker Strassen

NSAの警告 – 2015年

こうした認識に大きな変化が現れるのは、Shorの発見から、約20年後の2015年になってからのことである。

2015年、NSAは次のような重要な決定を公表した。

「我々は、来るべき量子耐性アルゴリズムへの移行について、早いうちから計画づくりとコミュニケーションを開始することを決定した。我々の最終的な目標は、量子コンピュータの潜在的な能力に対して、コスト効率の良いセキュリティを提供することである。」

NSAの決定を受けて、NISTは2016年から “Post-Quantum Cryptography”の標準化の策定の作業を開始した。その計画によれば、早ければ2022年、遅くとも2024年までには、新しい「ポスト量子暗号技術」の標準を決定するという。

I estimate a 1/7 chance of breaking RSA-2048 by 2026 and a 1/2 chance by 2031.

-- Michele Mosca

Unfortunately, the growth of elliptic curve use has bumped up against the fact of continued progress in the research on quantum computing, which has made it clear that elliptic curve cryptography is not the long term solution many once hoped it would be. Thus, we have been obligated to update our strategy.

-- NSA

不幸なことに、楕円曲線の利用の拡大は、量子コンピューティング研究の絶え間ない進歩の事実と衝突するものである。
すなわち、量子コンピューティングの研究は、楕円曲線暗号化は、多くの人がかってそうなるだろうと期待したような長期間にわたって有効なソリューションではないことを明らかにした。
こうして、我々は、戦略の見直しを余儀なくされてきた。

-- National Security Agency　2015年

量子優越性のマイルストーンの達成 – 2019年

2019年10月、GoogleのMartinisらは、科学誌 Nature上で、Googleが開発した53qubitの量子プロセッサー Sycamore が、普通のコンピュータで解けば1万年以上かかる問題を 200秒で解いたとして「量子優越性」を達成したと発表した。
https://www.nature.com/articles/s41586-019-1666-5

この発表は、多くのメディアでも取り上げられ、ある政治家は「もはや、破れない暗号はない」とツィートし、またBit Coin が暴落するなど、大きな波紋を呼んだ。

発表されたGoogleの量子コンピュータが、現代の暗号を解くかと言えば、これは誤解に基づく反応だと言っていいのだが。

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

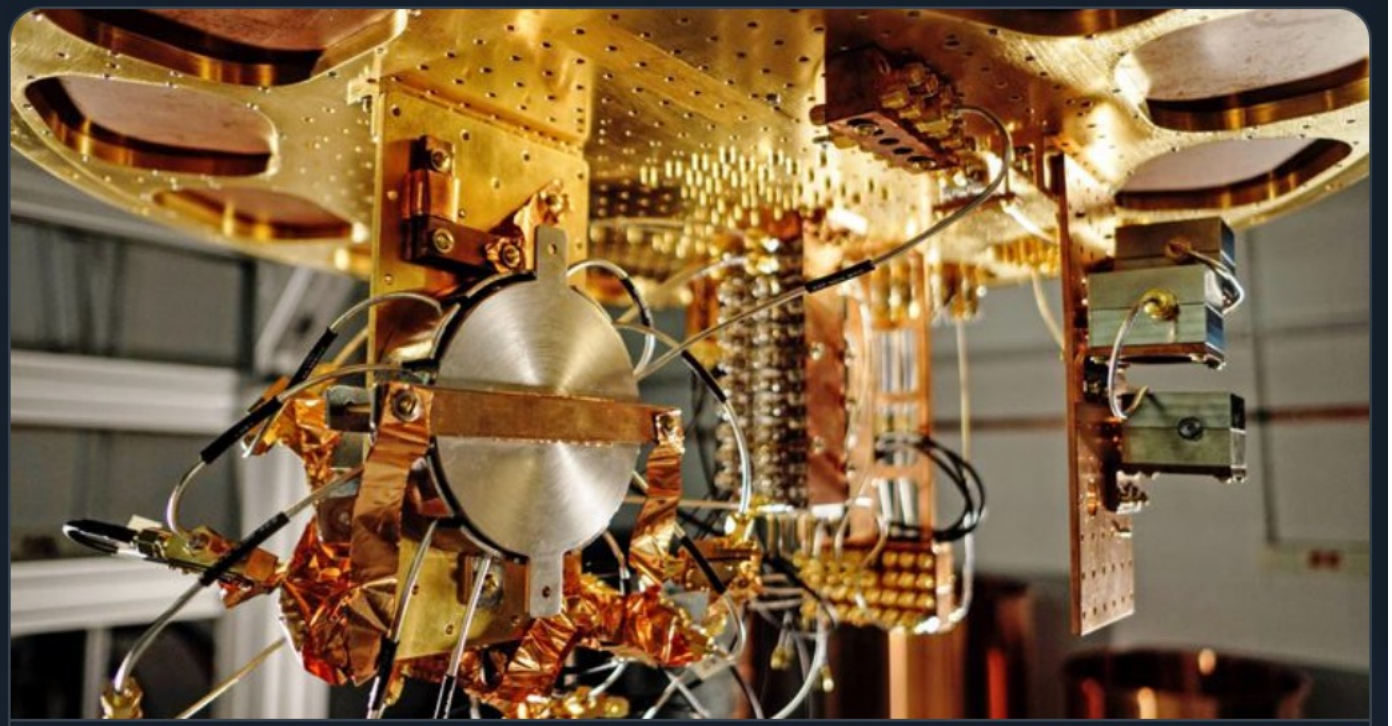
Andrew Yang 🧢 ✓
@AndrewYang

民主党の大統領候補の一人

Google achieving quantum computing is a huge deal. It means, among many other things, that no code is uncrackable.

クラックされないコードはない

ツイートを翻訳



Google reportedly attains 'quantum supremacy'
Its quantum computer can solve tasks that are otherwise unsolvable, a report says

ポスト量子暗号標準の提案
NIST IR 8413 -- 2022年

After three rounds of evaluation and analysis, NIST has selected the first algorithms it will standardize as a result of the PQC Standardization Process.

- The public-key encapsulation mechanism (KEM) that will be standardized is CRYSTALS–KYBER.
- The digital signatures that will be standardized are CRYSTALS–Dilithium, FALCON, and SPHINCS+. While there are multiple signature algorithms selected, NIST recommends CRYSTALS–Dilithium as the primary algorithm to be implemented.

In addition, four of the alternate KEM candidate algorithms will advance to a fourth round of evaluation: BIKE, Classic McEliece, HQC, and SIKE. These candidates will be considered for future standardization at the conclusion of the fourth round.

確率的素数判定

ショアの素因数分解アルゴリズムは、量子コンピュータを必要としない部分とそうでない部分に分かれる。

ここでは、主に、量子コンピュータを必要としない、確率的素数判定の方法について説明する。

フェルマーの小定理

Nが素数の時、Nと互いに素な整数aについて、次の式が成り立つ。

$a^N \equiv a \pmod N$　　この式は、両辺をaで割って

$a^{N-1} \equiv 1 \pmod N$　　と同じである。

$2^3 = 8 = 3 \times 2 + 2 \equiv 2 \pmod 3$

$2^5 = 32 = 5 \times 6 + 2 \equiv 2 \pmod 5$

$2^7 = 128 = 7 \times 18 + 2 \equiv 2 \pmod 7$

$2^{11} = 2048 = 11 \times 186 + 2 \equiv 2 \pmod{11}$

対偶をとれば、

aとNが互いに素で、$a^{N-1} \not\equiv 1 \pmod N$ ならば、Nは素数ではない

ことがわかる

フェルマー・テスト　確率的素数判定

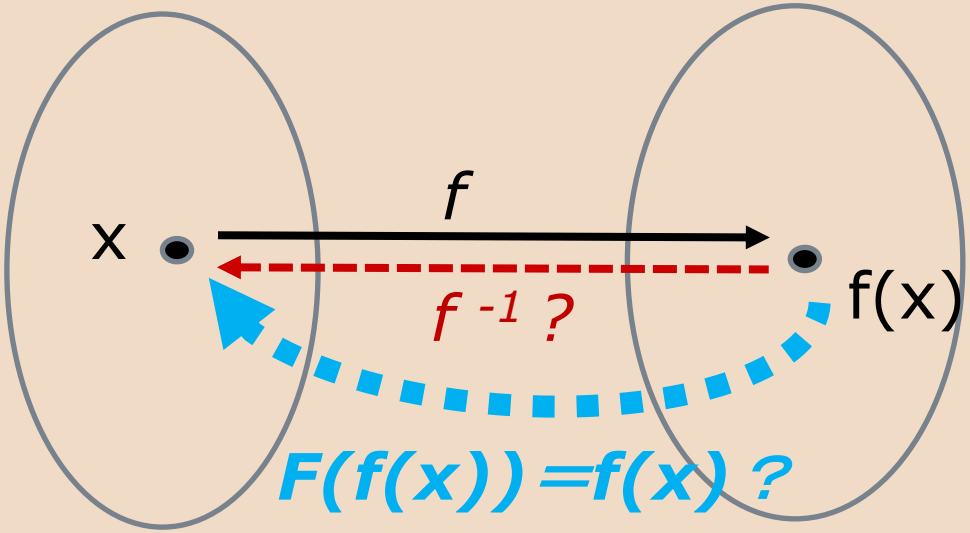
1. N以下の自然数aを一つ選ぶ
2. aとNが互いに素でないなら、Nは素数でない
3. $a^{N-1} \not\equiv 1 \pmod{N}$ ならば、Nは素数でない
4. そうでないなら、Nは素数である確率がある

この時、別のaを選んで、この処理を繰り返す。

十分な回数だけ a を取り替えて繰りかえしても、Nがこのテストをパスするなら、その数は実際に素数である可能性が高い。

shorのアルゴリズムも、同じような確率的方法をとる。

一方向関数の「難しさ」の確率的定義（簡略版）



f(x)の値からxの値をランダムに推定するすべてのアルゴリズムFに対して、f(F(f(x))=f(x)が、一定の確率以下でしか成り立たないなら、fは「難しい」と判断できる。

f(x)は既知として、f(x)の値からxの値を推定することを目指す。

あるアルゴリズム Fがあってそれによるxの推定値をF(f(x))とする。

このxの推定が正しいかは、この値をfに代入して、それがf(x)と等しいかをチェックすれば良い。

f(F(f(x))=f(x) ？

もしFがf^{-1}と完全に一致していれば、100%の確率で、次の式が成り立つ。

f(F(f(x))=f(x)

関数a^xの周期（あるいは、群の位数）

$a^r \equiv 1 \pmod N$ なるrが存在する時、この最小のrを（Nについての）aの周期（あるいは位数）という。

例えば、$N = 15, a = 7$ とすると、周期は4である。

$7^0 = 1 \pmod{15}$
$7^1 = 7 \pmod{15}$
$7^2 = 4 \pmod{15}$
$7^3 = 13 \pmod{15}$
$7^4 = 1 \pmod{15}$
$7^5 = 7 \pmod{15}$
$7^6 = 4 \pmod{15}$
$7^7 = 13 \pmod{15}$
$7^8 = 1 \pmod{15}$
$7^9 = 7 \pmod{15}$

周期（位数）を使った素因数分解

xの周期をrとする。rが偶数の時、次のようにかける。

$$
\begin{aligned}
x^r &\equiv 1 \mod N &\iff \\
(x^{r/2})^2 &\equiv 1 \mod N &\iff \\
(x^{r/2}+1)(x^{r/2}-1) &\equiv 0 \mod N &\iff \\
(x^{r/2}+1)(x^{r/2}-1) &= kN \text{ for some } k.
\end{aligned}
$$

すなわち、$x^{r/2}-1$ と$x^{r/2}+1$は、Nと約数を共有している。

ただ、$x^{r/2}-1 \equiv 0 \bmod N$にはならない。rは $x^r-1 \equiv 0 \bmod N$となる「最小」の r だから。Nは、$x^{r/2}-1$ を割らない。N∤ $x^{r/2}-1$

もし、$x^{r/2}+1 \not\equiv 0 \bmod N$ で、Nが $x^{r/2}+1$も割らないなら

gcd($x^{r/2}+1$,N)かgcd($x^{r/2}-1$,N)のいずれかが、Nを割ることになる。

周期（位数）を使った素因数分解

例えば、先の、N=15, a=7, r=4の時、

$\gcd(a^{r/2}+1,N)=\gcd(7^2+1,15)=\gcd(50,15)=5$
$\gcd(a^{r/2}-1,N)=\gcd(7^2-1,15)=\gcd(48,15)=3$

3,5は、15を割る。

周期を使ったショアの素因数分解アルゴリズム

1. a < N なるaをランダムに選ぶ。
2. aとNの最大公約数gcd(a, N)を計算する。（ユークリッドの互除法を用いる） gcd(a, N)が1でなかったら、その数がNの約数である。
3. f(x)＝ a^x mod N なる関数の周期をr とする。$a^r \equiv 1 \bmod N$ すなわち、$a^r - 1$は、Nで割れる。$N \mid a^r - 1$
   （この周期rを求めるのに、量子コンピュータを利用する）
4. rが奇数なら、1.に戻る。
5. rが偶数なら $a^r - 1 = (a^{r/2} + 1)(a^{r/2} - 1)$と表せる。
6. もし、$a^{r/2} = -1 \bmod N$ なら 1.に戻る。
7. そのいずれでもないなら、$\gcd(a^{r/2} + 1, N)$か$\gcd(a^{r/2} - 1, N)$のいずれかが、Nを割る。

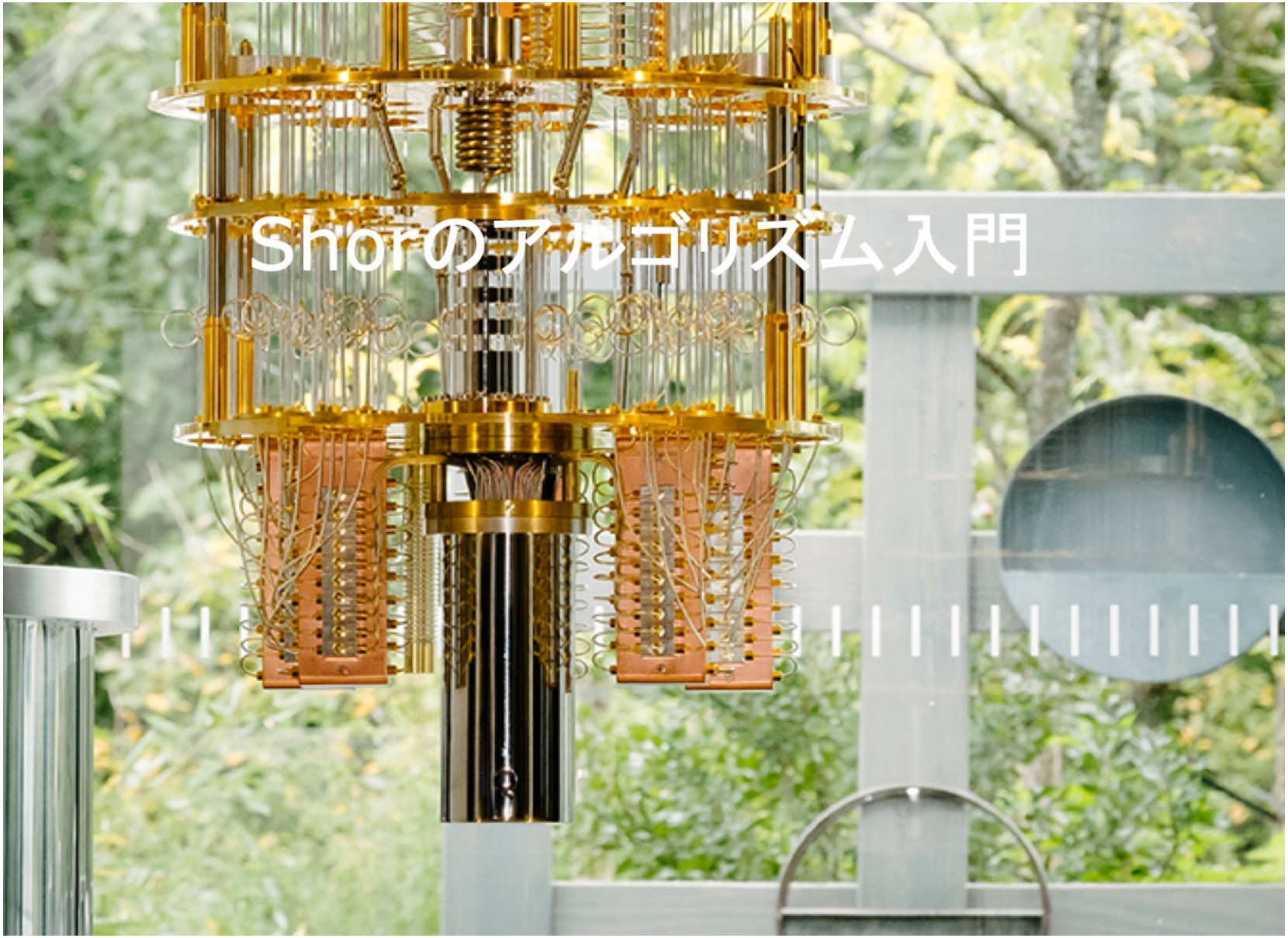
量子回路で「周期」を発見する

Shorのアルゴリズムを解く量子回路

量子コンピューターの世界では、「アルゴリズム」は「量子回路」によって表現される。（残念ながら）

Shorのアルゴリズムは、次のような量子回路から構成される。

- 任意のf(x)を計算する量子回路
- 可能なすべての基底の等しい重ね合わせを作る回路
- Quantum Parallelism 回路
- Simonの問題を解く回路
- 量子フーリエ変換 QFT 回路
- 周期を求める Phase Estimator回路

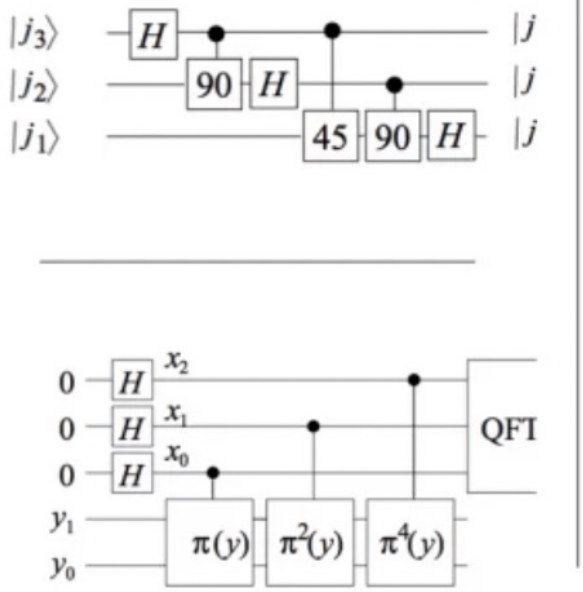


https://www.marulabo.net/docs/shor/

量子アルゴリズム入門 -- 量子フーリエ変換を学ぶ

2018年11月2日　2021年12月24日　MaruyamaFujio

11/2 マルレク 第四回
「量子アルゴリズム入門」



「ショアのアルゴリズム」の中核である「量子フーリエ変換」を学ぶ

https://www.marulabo.net/docs/20181102-marulec04

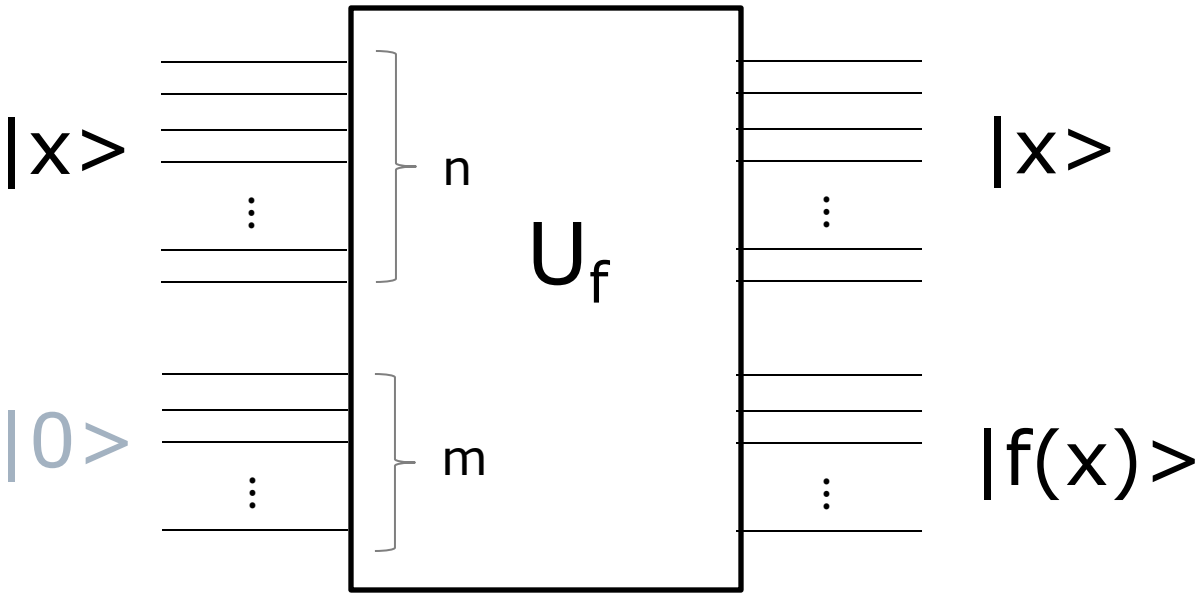
任意のf(x)を計算する
量子回路を考える

https://drive.google.com/file/d/1sUNkAd4ahHZ14nWxK5zz-ynn4IdmtlAw/view?usp=sharing

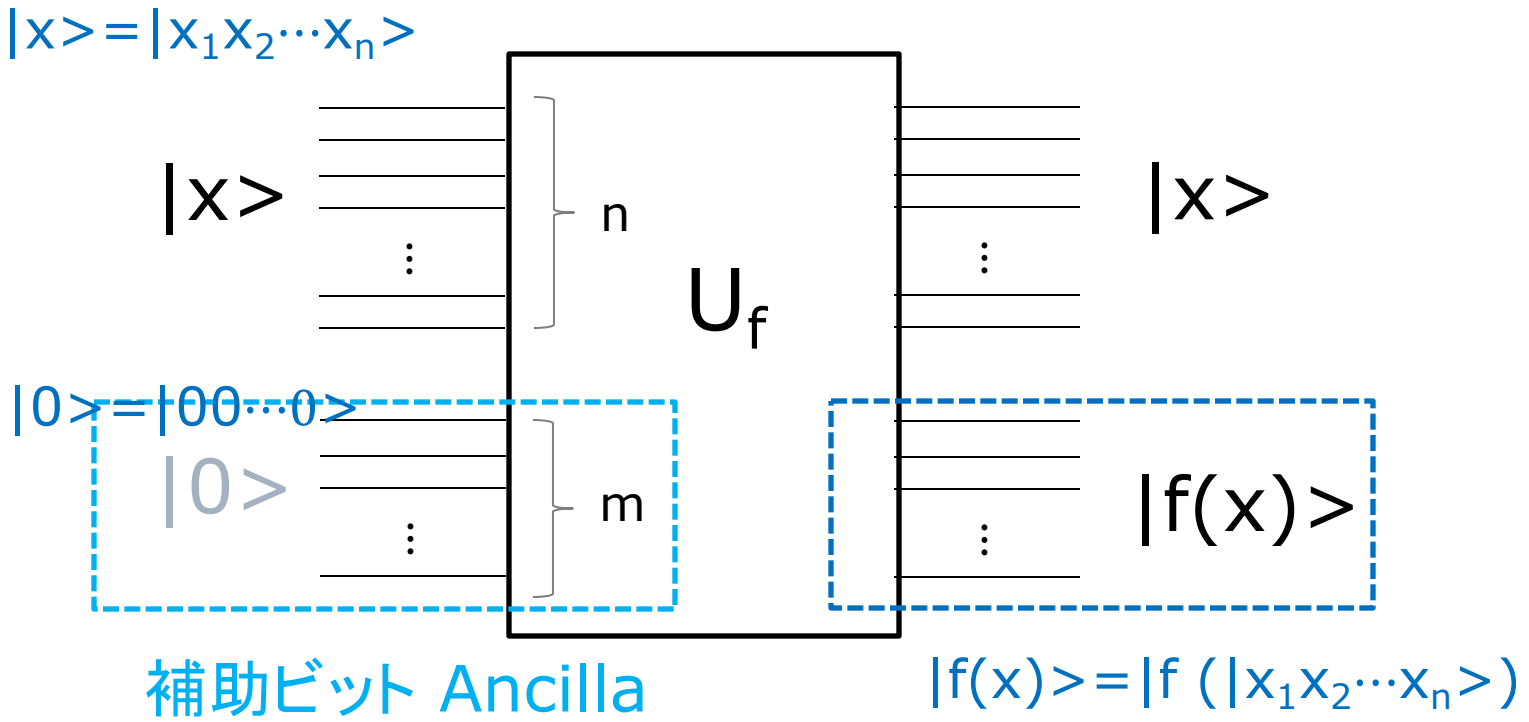
「Shorのアルゴリズム入門」 Part I
https://www.marulabo.net/docs/shor/

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

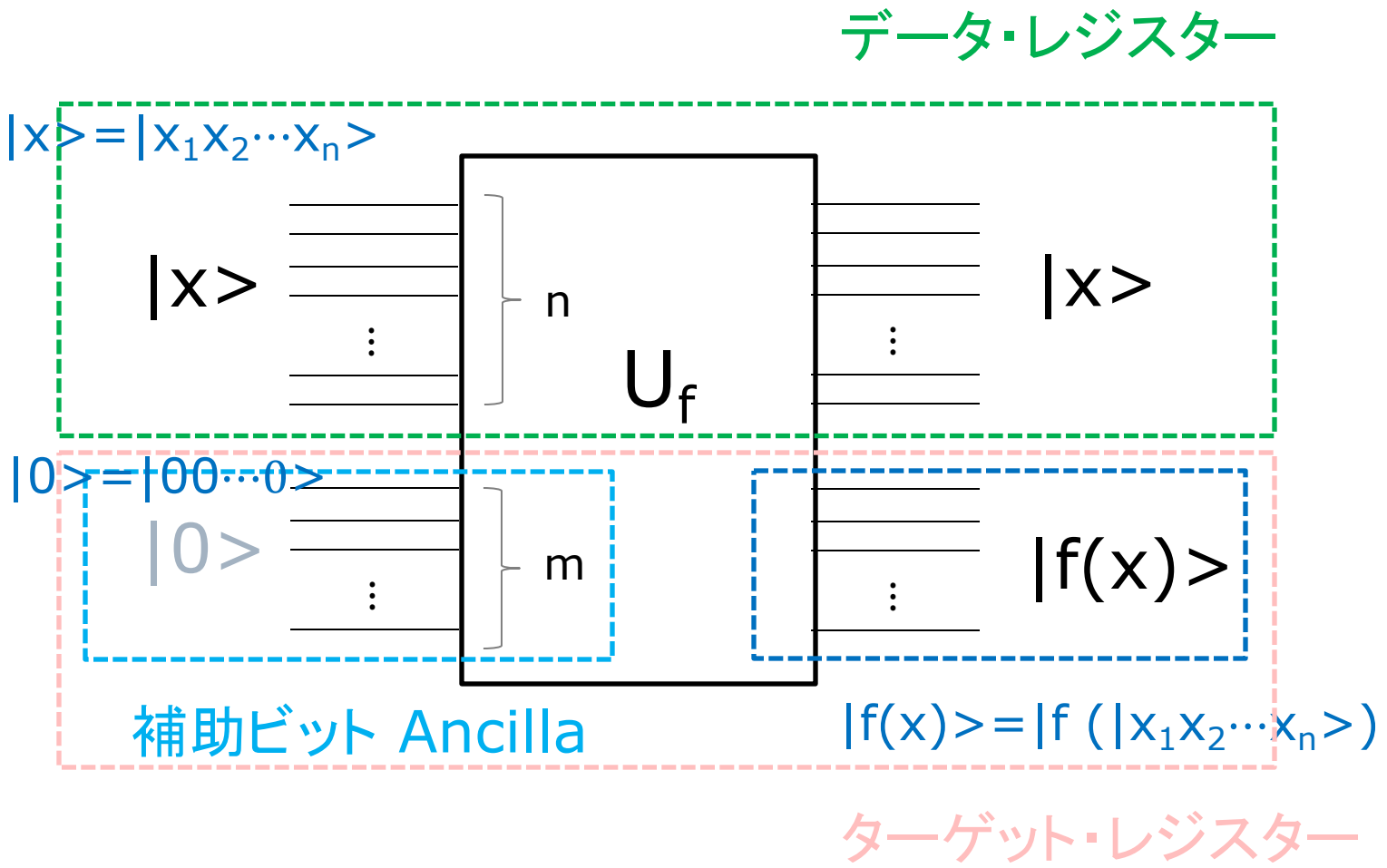
f(x)の具体的な実装を与えているわけではない。
ブラックボックスとかOracleと言ったりする



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

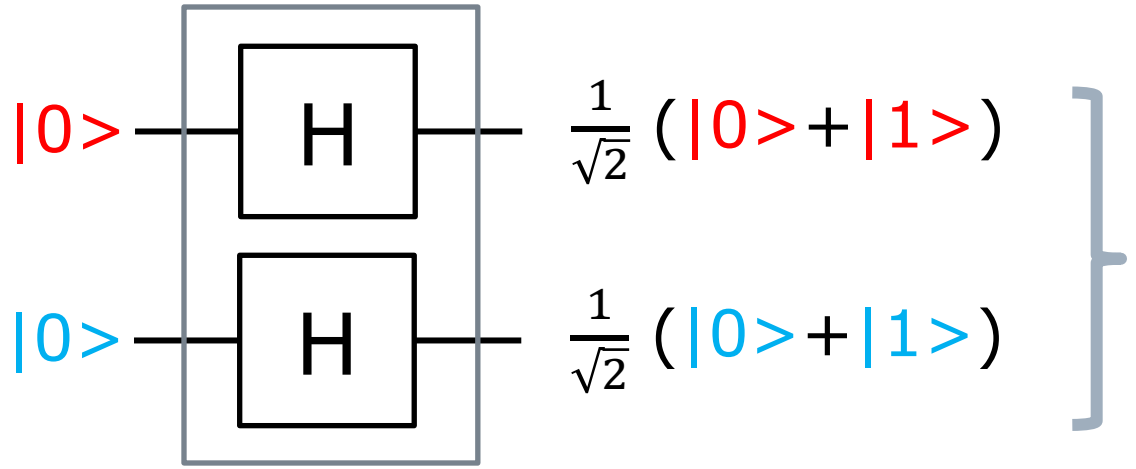


可能なすべての基底の
等しい重ね合わせを作る

https://drive.google.com/file/d/1120ypQXMZKz4m8m5EegtB4-z6tJo4xRA/view?usp=sharing

「Shorのアルゴリズム入門」 Part II
https://www.marulabo.net/docs/shor/

アダマール･ゲート 2 個の場合



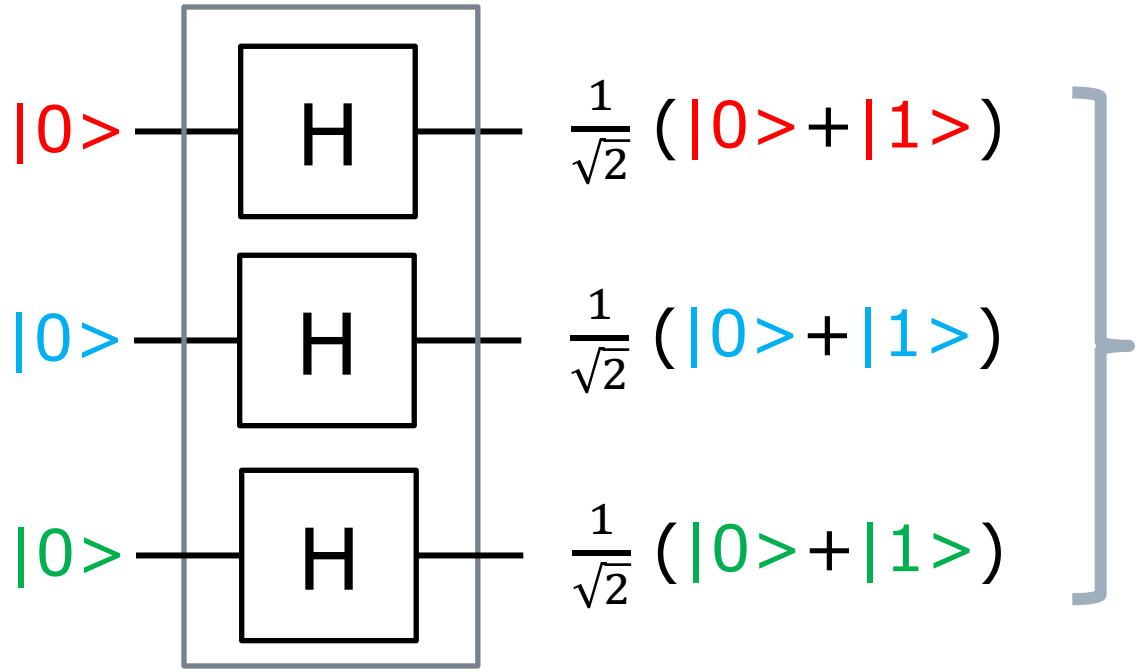
$$|0\rangle\otimes|0\rangle \Rightarrow \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)\otimes\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$$

$$= \left(\frac{1}{\sqrt{2}}\right)^2(|0\rangle\otimes|0\rangle+|0\rangle\otimes|1\rangle+|1\rangle\otimes|0\rangle+|1\rangle\otimes|1\rangle)$$

$$= \left(\frac{1}{\sqrt{2}}\right)^2(|00\rangle+|01\rangle+|10\rangle+|11\rangle)$$

これは、可能な基底 |00>, |01>, |10>, |11>
の等しい重ね合わせである。

アダマール･ゲート 3 個の場合



$$\Rightarrow \frac{1}{\sqrt{2}}(|0>+|1>)\otimes\frac{1}{\sqrt{2}}(|0>+|1>)\otimes\frac{1}{\sqrt{2}}(|0>+|1>)$$

$$= \left(\frac{1}{\sqrt{2}}\right)^3(\ |000>+|001>+|010>+|011>+$$

$$|100>+|101>+|110>+|111> \ \ )$$

これは、可能な基底すべての等しい重ね合わせである。

アダマール変換は、全ての計算基底の等しい重ね合わせを生み出す。これは、n個のゲートを使って2^n個の状態の重ね合わせを生み出す非常に効率的な方法である。

n個の出力のテンソル積

n個

$$\Rightarrow \frac{1}{\sqrt{2}}(|0>+|1>)\otimes\frac{1}{\sqrt{2}}(|0>+|1>)\otimes\cdots\otimes\frac{1}{\sqrt{2}}(|0>+|1>)$$

$$=\left(\frac{1}{\sqrt{2}}\right)^n\sum|Xi>$$

$|Xi>$は、次のようなすべての基底である

n桁

$$|X_0> = |000\cdots\cdots000>$$
$$|X_1> = |000\cdots\cdots001>$$
$$|X_2> = |000\cdots\cdots010>$$
$$|X_3> = |000\cdots\cdots011>$$
$$\vdots \qquad\qquad \vdots$$
$$|X_k> = |111\cdots\cdots111>$$

2^n個

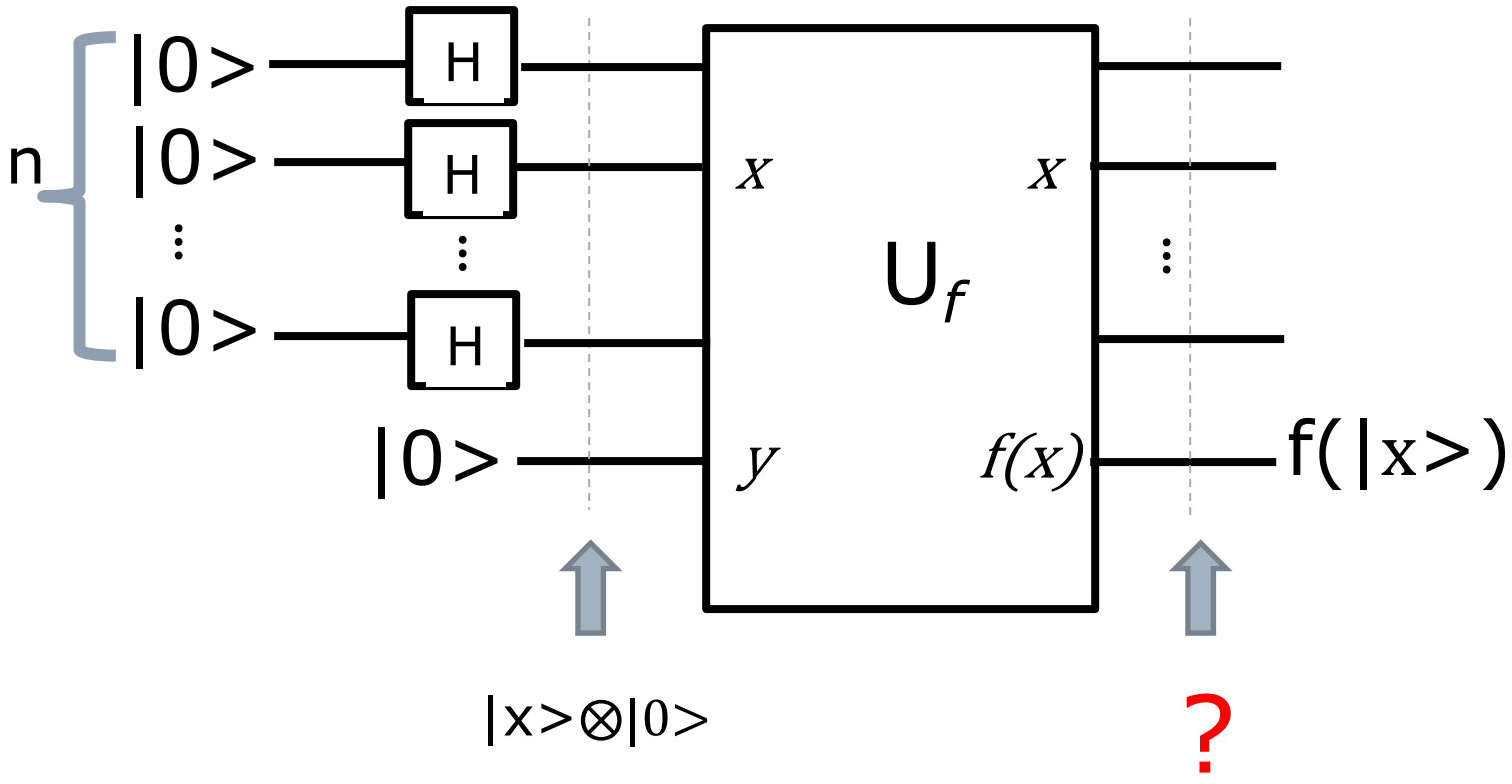
Quantum Parallelism

https://drive.google.com/file/d/1120ypQXMZKz4m8m5EegtB4-z6tJo4xRA/view?usp=sharing

「Shorのアルゴリズム入門」 Part II
https://www.marulabo.net/docs/shor/

(n+1)-qubit(fの入力　n-qubit, fの出力 1-qubit)の回路U_fを考える

Ufの入力に、パラレルに置かれたアダマール・ゲート n個の出力（それぞれの入力は|0>）をつなげてみよう



先の回路U_fの出力を考える

- Ufへの入力xは、$H^{\otimes n}|0\rangle^{\otimes n}$ で、これは$\left(\frac{1}{\sqrt{2}}\right)^n \sum |Xi\rangle$ に等しい。ただし、$Xi \in \{0,1\}^n$であるすべてについて和をとる。
- $Uf(|x\rangle \otimes |0\rangle) = Uf\left(\left(\frac{1}{\sqrt{2}}\right)^n \sum |Xi\rangle \otimes |0\rangle\right)$

  $= \left(\frac{1}{\sqrt{2}}\right)^n \sum Uf(|Xi\rangle \otimes |0\rangle)$

  $= \left(\frac{1}{\sqrt{2}}\right)^n \sum (|Xi\rangle \otimes f(|Xi\rangle))$

Quantum Parallelism

- 例えば、n=3の時、Ufの出力は次のようになる。

$$\left(\frac{1}{\sqrt{2}}\right)^3 (\; |000> f(|000>) + |001> f(|001>) + |010> f(|010>) + |011> f(|011>) + |100> f(|100>) + |101> f(|101>) + |110> f(|110>) + |111> f(|111>) \;)$$

ただし、|x> ⊗ f(|x>)のテンソル記号を省略している。
三つのqubitに対する一回のUfの呼び出しが、f(|000>)から
f(|111>)までの8つのfの値を計算していることがわかる。

一般に、n個のqubitに対するUfの呼び出しは、2^n個の f の値を計算することになる。

一般化されたBornのルール

システムAの状態が $|k\rangle_A^n$ で表され、システムBの状態が $|\psi_k\rangle_B^m$ で表される時、次の状態は、A ⊗ Bの状態である。

$$|\varphi\rangle^{n+m} = |0\rangle_A^n \, |\psi_0\rangle_B^m + |1\rangle_A^n \, |\psi_1\rangle_B^m + \cdots + |2^n-1\rangle_A^n \, |\psi_{2^n-1}\rangle_B^m$$

この状態が観測された時、
システムAの状態は、特定の一つのk_0について $|k_0\rangle_A^n$ に変化し、
システムBの状態は、この同じk_0について $|\psi_{k_0}\rangle_B^m$ に変化する。
システムA ⊗ Bとしてみれば、
状態 $|\varphi\rangle^{n+m}$ は、新しい状態 $|k_0\rangle_A^n \, |\psi_{k_0}\rangle_B^m$ に変化する。

何かが必要である

- 量子並列計算は、直ちに有用なわけではない。
  最初の単一qubitのサンプルで $\frac{|0, f(0)\rangle + |1, f(1)\rangle}{\sqrt{2}}$、

  この状態の測定は、|0> f(0) 〉か |1> f(1)〉の値のどれかを返すだけだ。もっと一般的に、状態 Σ_x |x> f(x) 〉の測定は、一つの値xについてのf(x)の値を返すだけだ。もちろん、古典的なコンピュータは、そうしたことを簡単にやってのける。
- 量子計算が役に立つためには、単なる量子並行計算以上の何かが必要になる。すなわち、Σ_x |x> f(x) 〉のような重ね合わせの状態から、f(x)の一つの値より多くの値の情報を引き出せるような能力が必要になる。

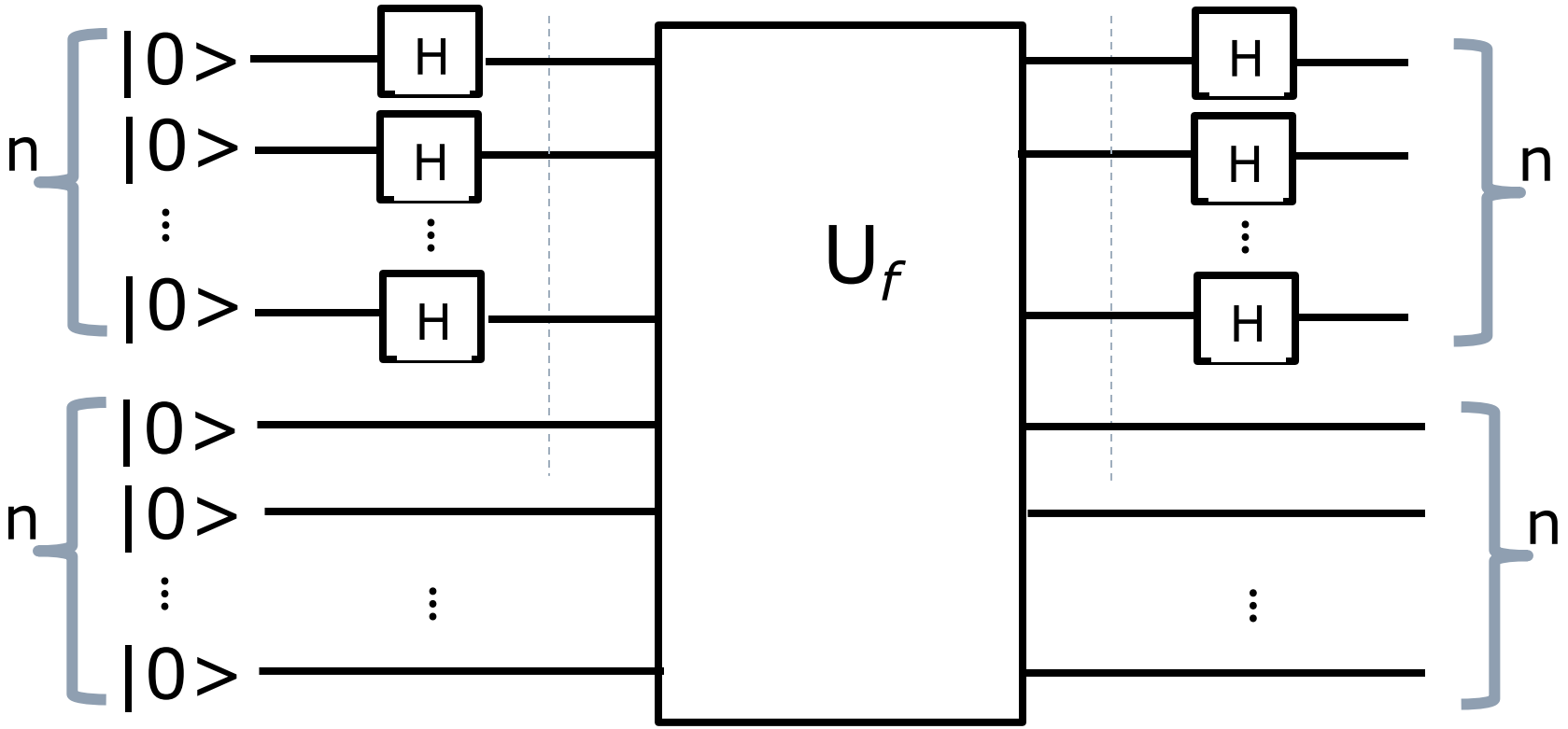
Simonの問題

$f(x)=f(x \oplus a)$ となる 関数fの「周期」 aを求める

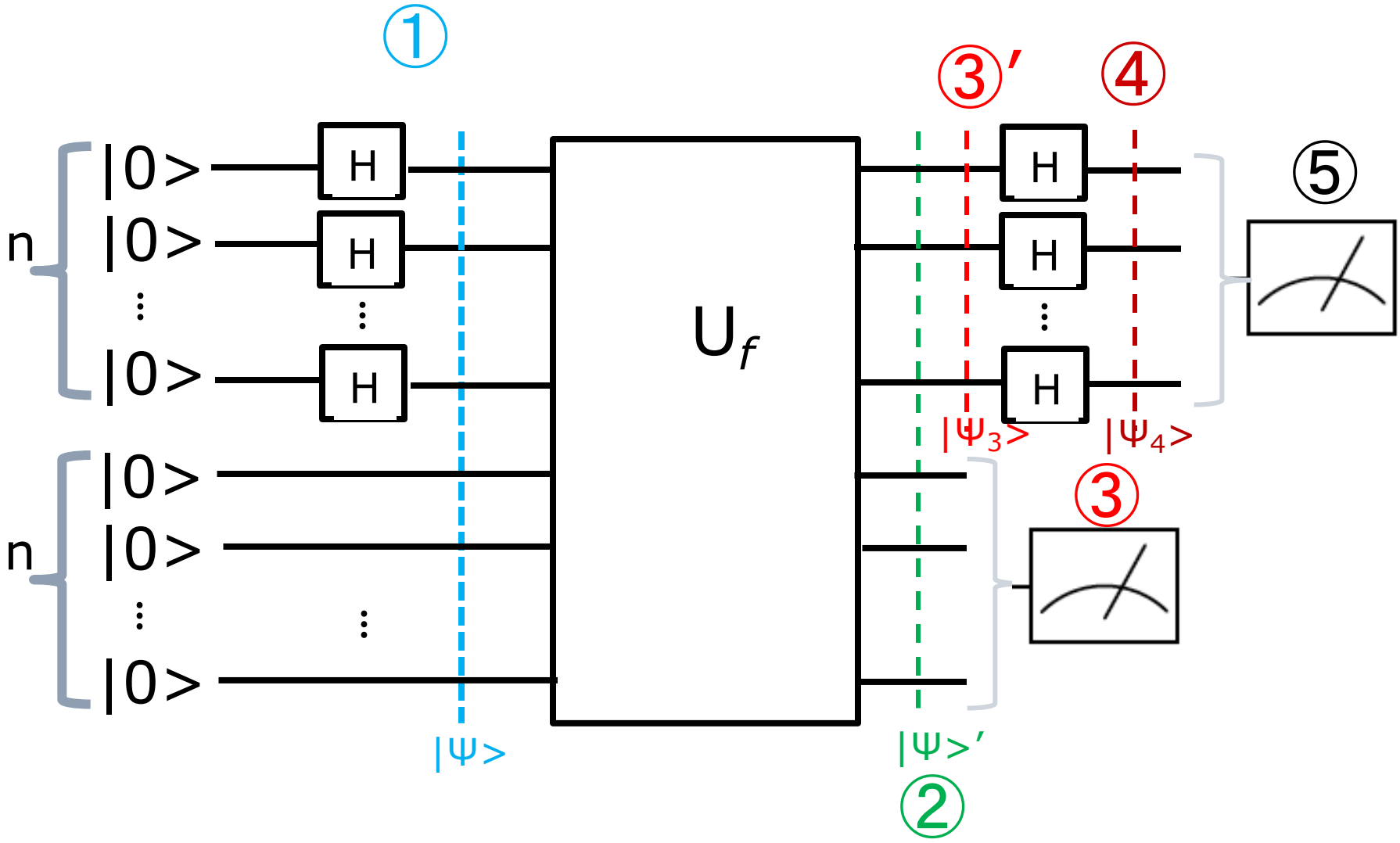
https://drive.google.com/file/d/1crCF5xkF3wr1hOvTtY0zaCEtY7vOkKDy/view?usp=sharing

「Shorのアルゴリズム入門」 Part IIi
https://www.marulabo.net/docs/shor/

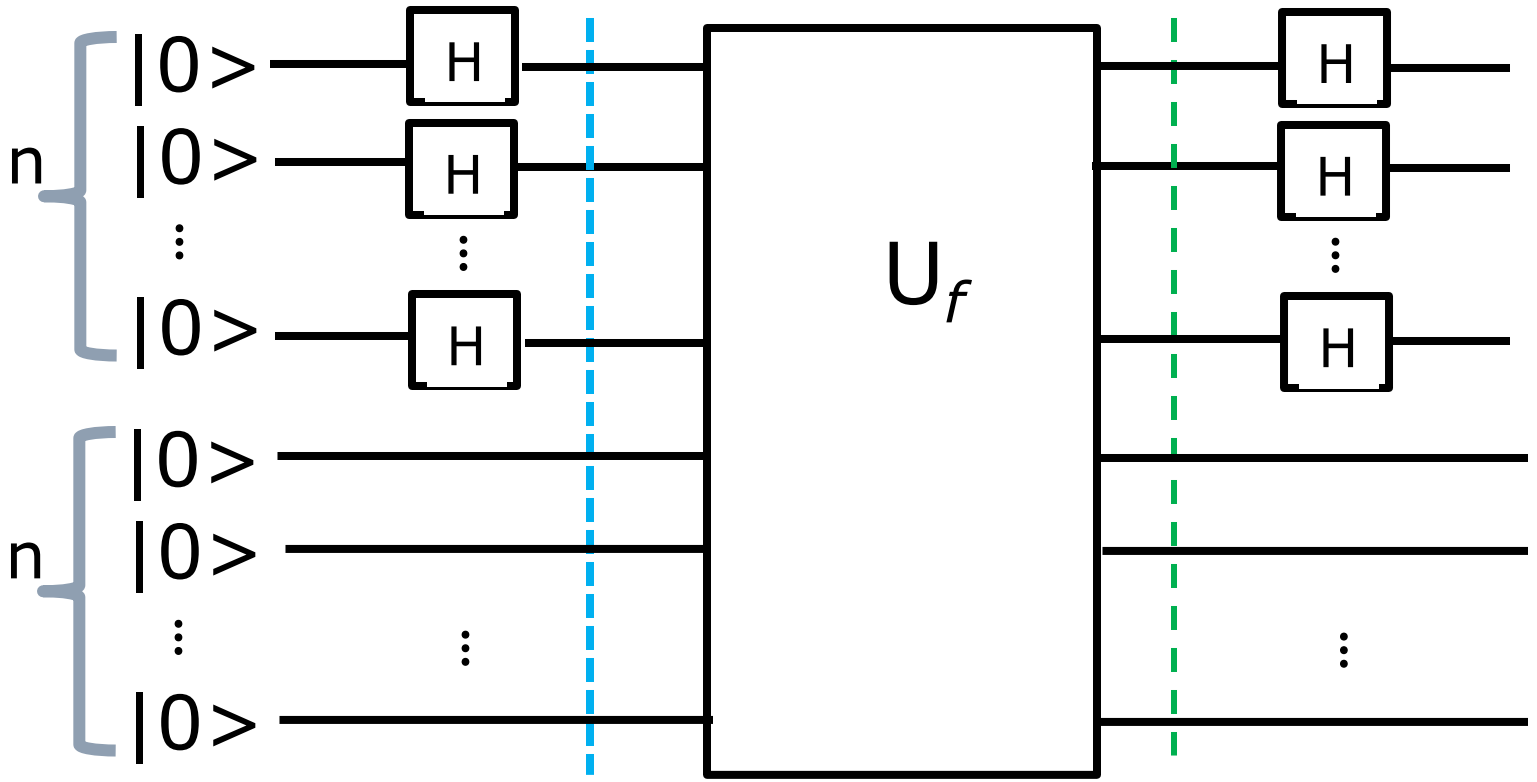
Simonの問題を解く量子回路



一連の流れ



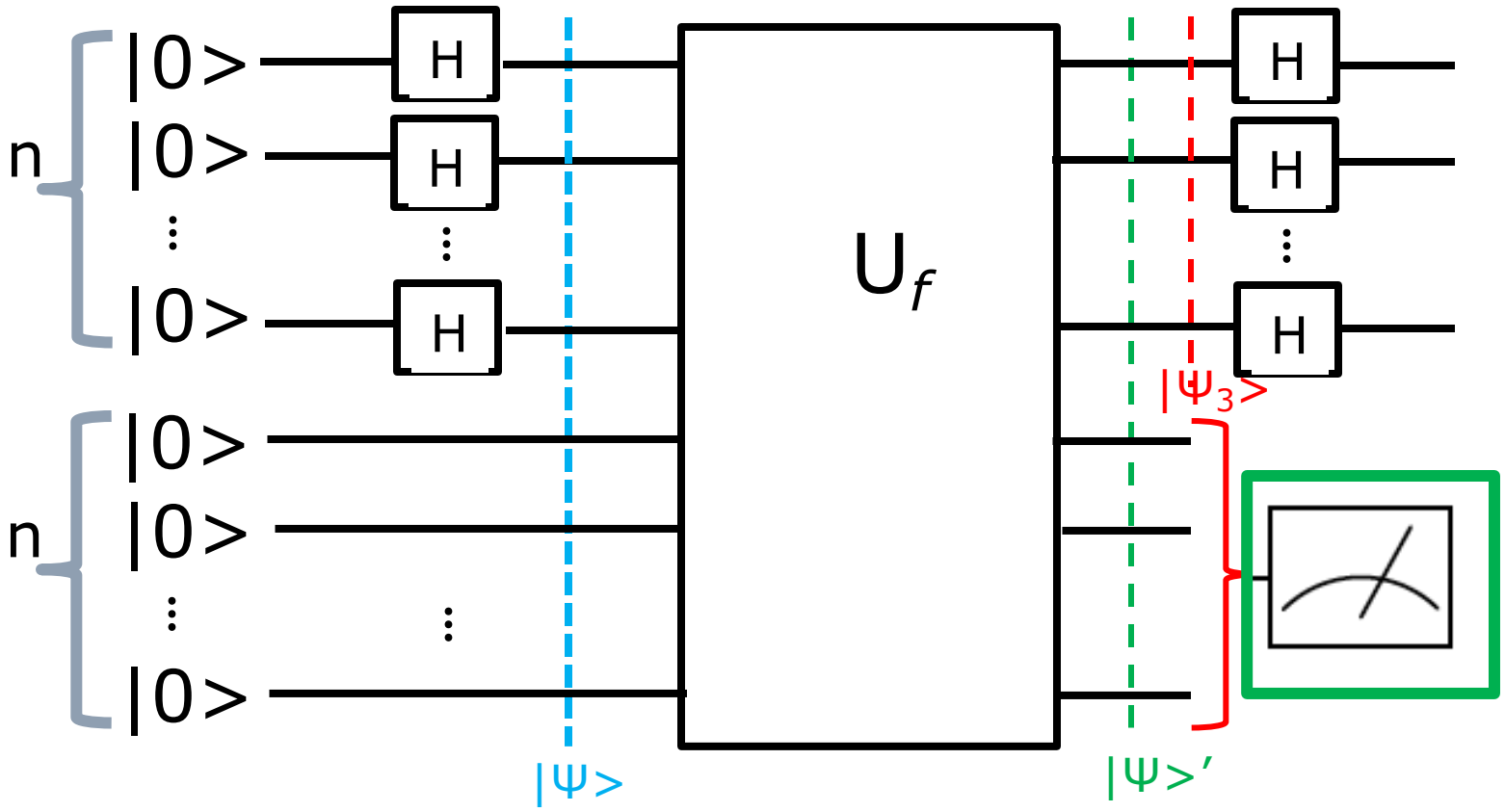
各段階での |Ψ>, |Ψ>′の状態は、すでに見たように、次のように計算できる



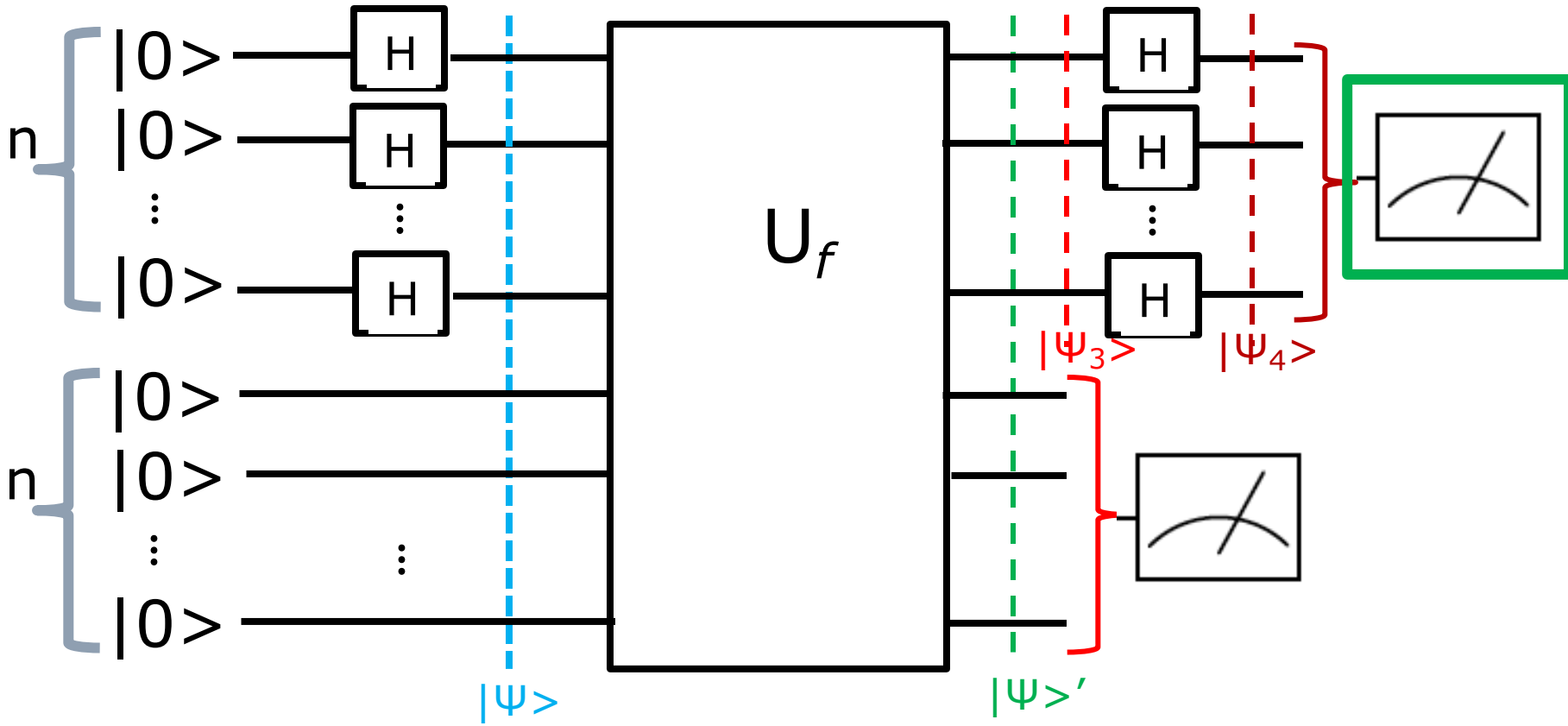
$$|\Psi\rangle = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (|x\rangle \otimes |0^n\rangle)$$

$$|\Psi\rangle' = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (|x\rangle \otimes |f(x)\rangle)$$

ターゲット・レジスターを観測する。観測によって変化したデータ・レジスターの状態を$|\Psi_3\rangle$とする



$|\Psi_3\rangle$をn個のアダマールに通し、その結果の$|\Psi_4\rangle$を観測する



$|\Psi_4\rangle$を観測する

$$|\Psi_4\rangle = \left(\frac{1}{\sqrt{2}}\right)^{n-1} \sum_{\substack{\mathbf{y}\cdot\mathbf{a}=0 \\ (\mathrm{mod}\ 2)}} (-1)^{\mathbf{y}\cdot\mathbf{x_0}} \quad |y\rangle^n$$

を観測すると、$|\Psi_4\rangle$ は、状態 $|y_0\rangle$ に変化する。
ただし、このy_0について、$y_0 \cdot a=0$ である。

量子フーリエ変換 QFT 回路

https://drive.google.com/file/d/1IPSXRocQptwdbN-N63pMmrZzQcL2n5NK/view?usp=sharing

「量子アルゴリズム入門 -- 量子フーリエ変換を学ぶ」Part V
https://www.marulabo.net/docs/20181102-marulec04/

Discrete Fourier Transformと Quantum Fourier Transform

- Discrete Fourier Transform

$$y_k \equiv \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i j k/N} .$$

同じ形をしている。

- Quantum Fourier Transform

$$|j\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N} |k\rangle$$

$$\sum_{j=0}^{N-1} x_j |j\rangle \longrightarrow \sum_{k=0}^{N-1} y_k |k\rangle ,$$

ただし、|j>, |k>は、
|1011..01>のような
形をしている。
n個の基底のテンソル積。
組み合わせとしては、
2^n個ある。

次のページで見るように
これを二進数としてみる。

Quantum Fourier Transform

- N=2^nとして、n qubitの量子コンピュータの計算基底を
  |0>,|1>, …. |2^n -1> とする。
  この時、$j=j_1j_2j_3\cdots j_n$で、$j = j_1 2^{n-1} + j_2 2^{n-2} + \ldots + j_n 2^0$ を表す。
  同様に、$0.j_l j_{l+1} \ldots j_m$で、$j_l/2 + j_{l+1}/4 + \ldots + j_m/2^{m-l+1}$を表す。

  この時、QFTは、次のように表現できる。

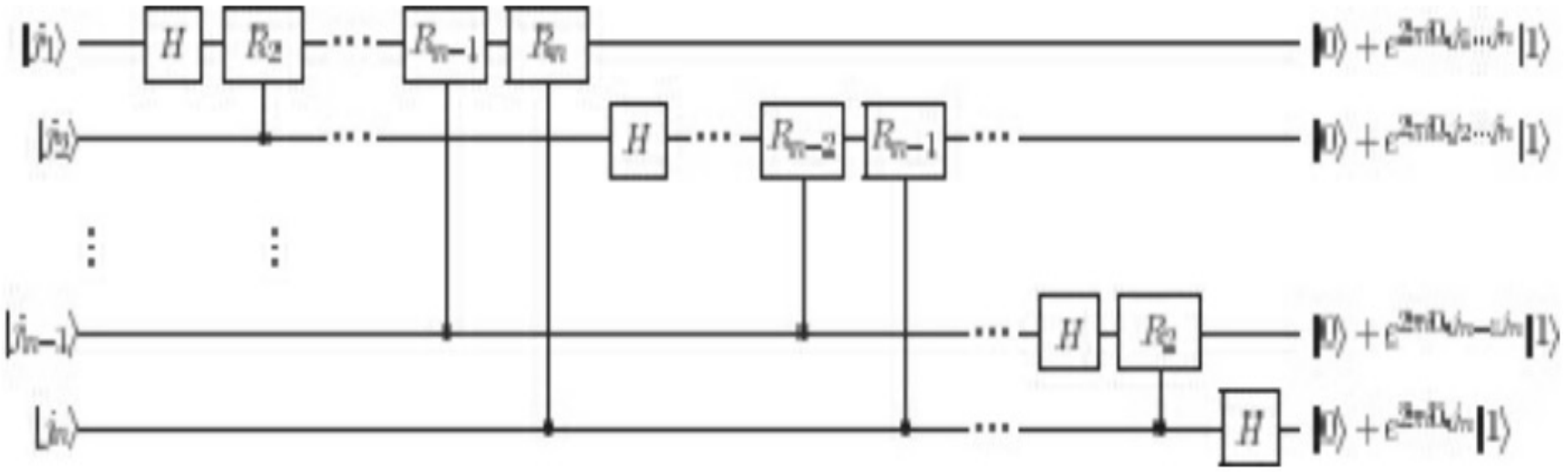
$$|j_1, \ldots, j_n\rangle \rightarrow$$

$$\frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2 \cdots j_n}|1\rangle\right)}{2^{n/2}}$$

以下、それを説明しよう。

QFTの基本公式

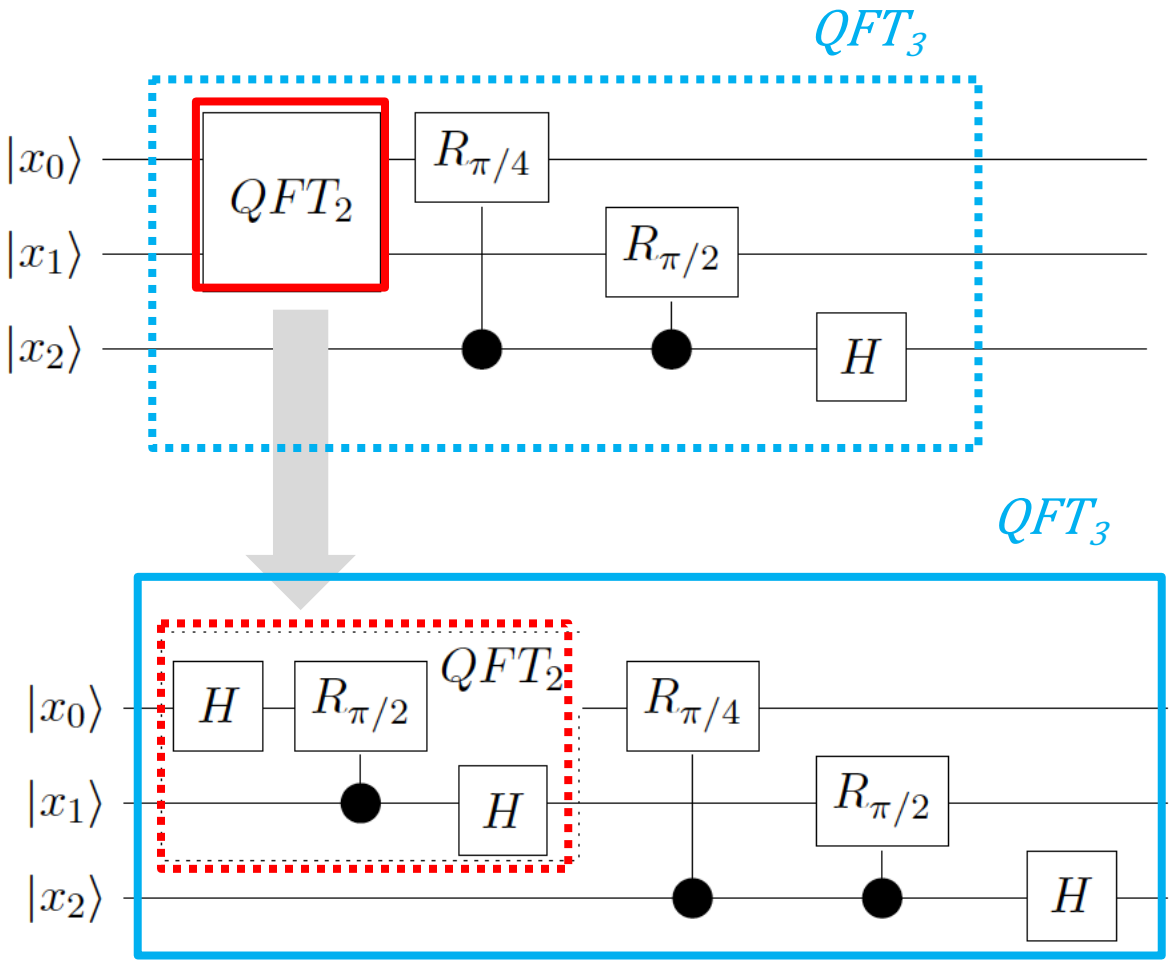
QFTの基本回路



QFTの基本公式

$$= \frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2 \cdots j_n}|1\rangle\right)}{2^{n/2}}.$$

QFT_3 の回路の例

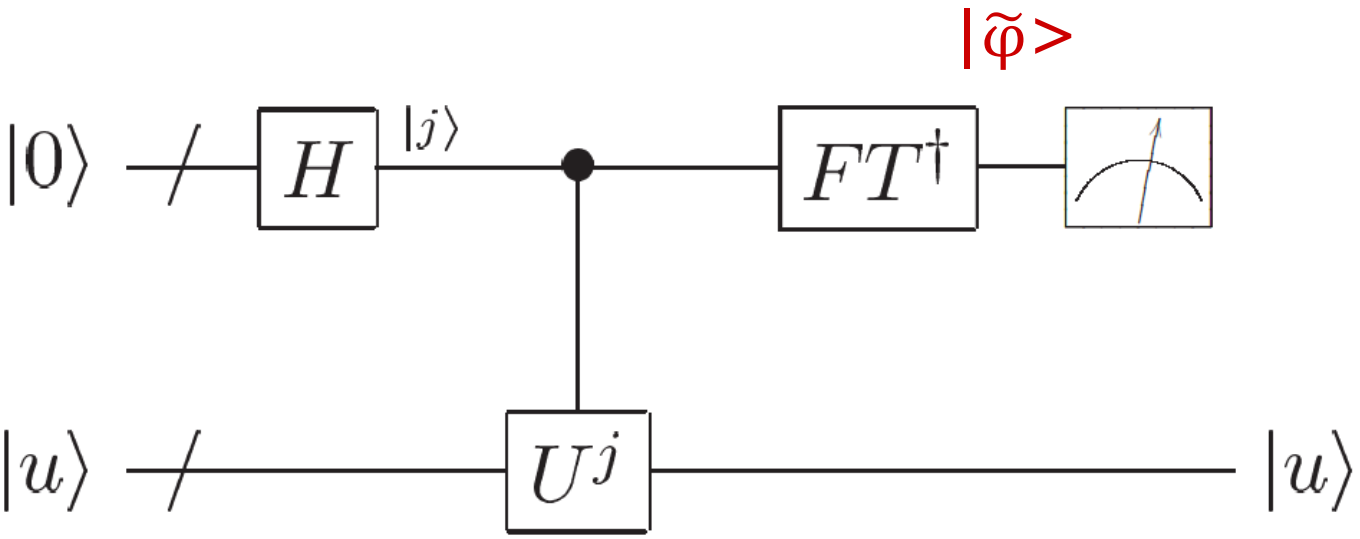

QFT_3
$|x_0\rangle$
$|x_1\rangle$
$|x_2\rangle$
QFT_2
$R_{\pi/4}$
$R_{\pi/2}$
H
QFT_3
$|x_0\rangle$
$|x_1\rangle$
$|x_2\rangle$
H
$R_{\pi/2}$
QFT_2
H
$R_{\pi/4}$
$R_{\pi/2}$
H

Phase Estimatorを使って、周期を求める

https://drive.google.com/file/d/1Pz4LasPcCUqjPYo2y8S5WQaVgqb-3YXO/view?usp=sharing

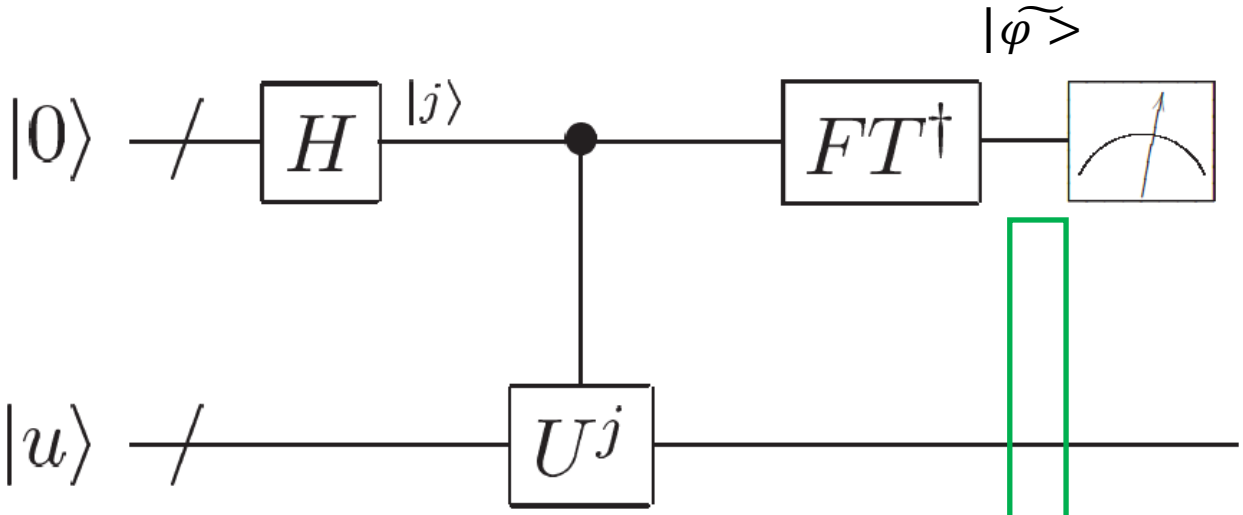
「Shorのアルゴリズム入門」 Part IV
https://www.marulabo.net/docs/shor/

Phase estimator 回路



$U|u\rangle = e^{2\pi i\varphi}|u\rangle$

すなわち、|u>がUの固有ベクトルで、$e^{2\pi i\varphi}$ がUの固有値になるようなφを求める回路



$$\frac{1}{2^{t/2}} \sum_{j=0}^{2^t-1} e^{2\pi i \varphi j} |j\rangle |u\rangle \rightarrow |\tilde{\varphi}\rangle |u\rangle$$

FT: Fourier 变换

$$|j_1,\ldots,j_n\rangle \rightarrow \frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2\cdots j_n}|1\rangle\right)}{2^{n/2}}$$

FT†:逆Fourier 变换

$$\frac{\left(|0\rangle + e^{2\pi i 0.j_n}|1\rangle\right)\left(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 0.j_1 j_2\cdots j_n}|1\rangle\right)}{2^{n/2}} \rightarrow |j_1,\ldots,j_n\rangle$$

Phase Estimatorの働き

1. $|0\rangle|u\rangle$ 初期状態

2. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|u\rangle$ 重ね合わせを作る

3. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle U^j|u\rangle$ ブラックボックスU_jを作用させる

$= \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} e^{2\pi i j \varphi_u} |j\rangle|u\rangle$ ブラックボックスU_jの作用の結果

4. $\rightarrow |\varphi_u\rangle|u\rangle$ 逆フーリエ変換を作用させる

5. $\rightarrow \varphi_u$ 第一レジスターを測定する

Phase Estimatorを使って、周期を求める

1. $|0\rangle|0\rangle$ 初期状態

2. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} |x\rangle|0\rangle$ 重ね合わせを作る

3. $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t-1} |x\rangle|f(x)\rangle$ Uを作用させる

   $\approx \frac{1}{\sqrt{r2^t}} \sum_{\ell=0}^{r-1} \sum_{x=0}^{2^t-1} e^{2\pi i \ell x/r} |x\rangle|\hat{f}(\ell)\rangle$ Uの作用の結果

4. $\rightarrow \frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} |\ell/r\rangle|\hat{f}(\ell)\rangle$ 逆フーリエ変換を作用させる

5. $\rightarrow \ell/r$ 第一レジスターを測定する

6. $\rightarrow r$ 周期 r を求める

Period-finding	$\mathbf{Z}$, +	any finite set	$\{0, r, 2r, \ldots\}$ $r \in G$	$f(x+r) = f(x)$
Order-finding	$\mathbf{Z}$, +	$\{a^j\}$ $j \in Z_r$ $a^r = 1$	$\{0, r, 2r, \ldots\}$ $r \in G$	$f(x) = a^x$ $f(x+r) = f(x)$
Discrete logarithm	$\mathbf{Z}_r \times \mathbf{Z}_r$ + (mod r)	$\{a^j\}$ $j \in Z_r$ $a^r = 1$	$(\ell, -\ell s)$ $\ell, s \in \mathbf{Z}_r$	$f(x_1, x_2) = a^{kx_1+x_2}$ $f(x_1+\ell, x_2-\ell s) = f(x_1, x_2)$
Order of a permutation	$\mathbf{Z}_{2^m} \times \mathbf{Z}_{2^n}$ + (mod 2^m)	$\mathbf{Z}_{2^n}$	$\{0, r, 2r, \ldots\}$ $r \in X$	$f(x, y) = \pi^x(y)$ $f(x+r, y) = f(x, y)$ π = permutation on X
Hidden linear function	$\mathbf{Z} \times \mathbf{Z}$, +	$\mathbf{Z}_N$	$(\ell, -\ell s)$ $\ell, s \in X$	$f(x_1, x_2) =$ $\pi(sx_1 + x_2 \bmod N)$ π = permutation on X
Abelian stabilizer	(H, X) H = any Abelian group	any finite set	$\{s \in H \mid$ $f(s, x) = x,$ $\forall x \in X\}$	$f(gh, x) = f(g, f(h, x))$ $f(gs, x) = f(g, x)$

Shorのアルゴリズムで解ける問題

http://bit.ly/2Ww2Kua

計算複雑性の新しいクラス BQP

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。

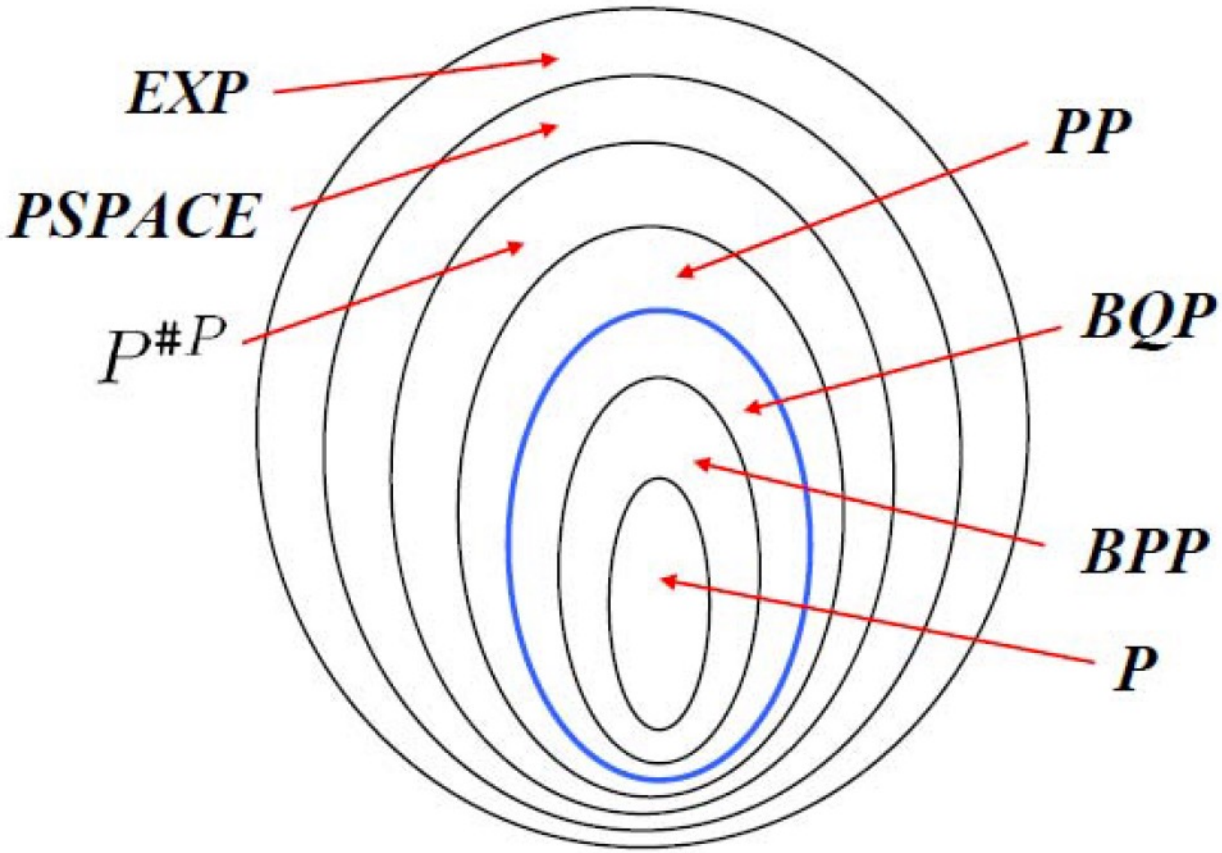
本論文では、量子チューリングマシンが古典的な確率的チューリングマシンよりも強力である可能性があるという最初の証拠を提示する。

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

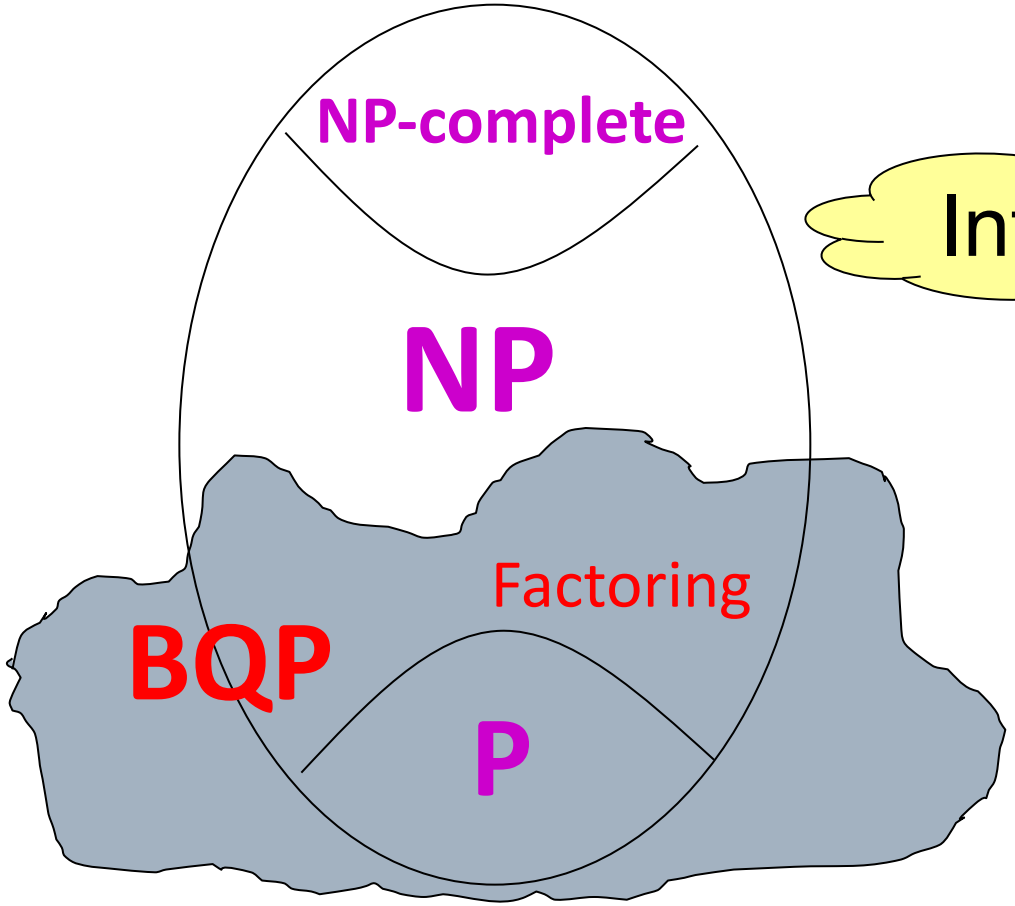
$$\mathbf{P} \subseteq \mathbf{BQP}$$

Basic properties of BQP



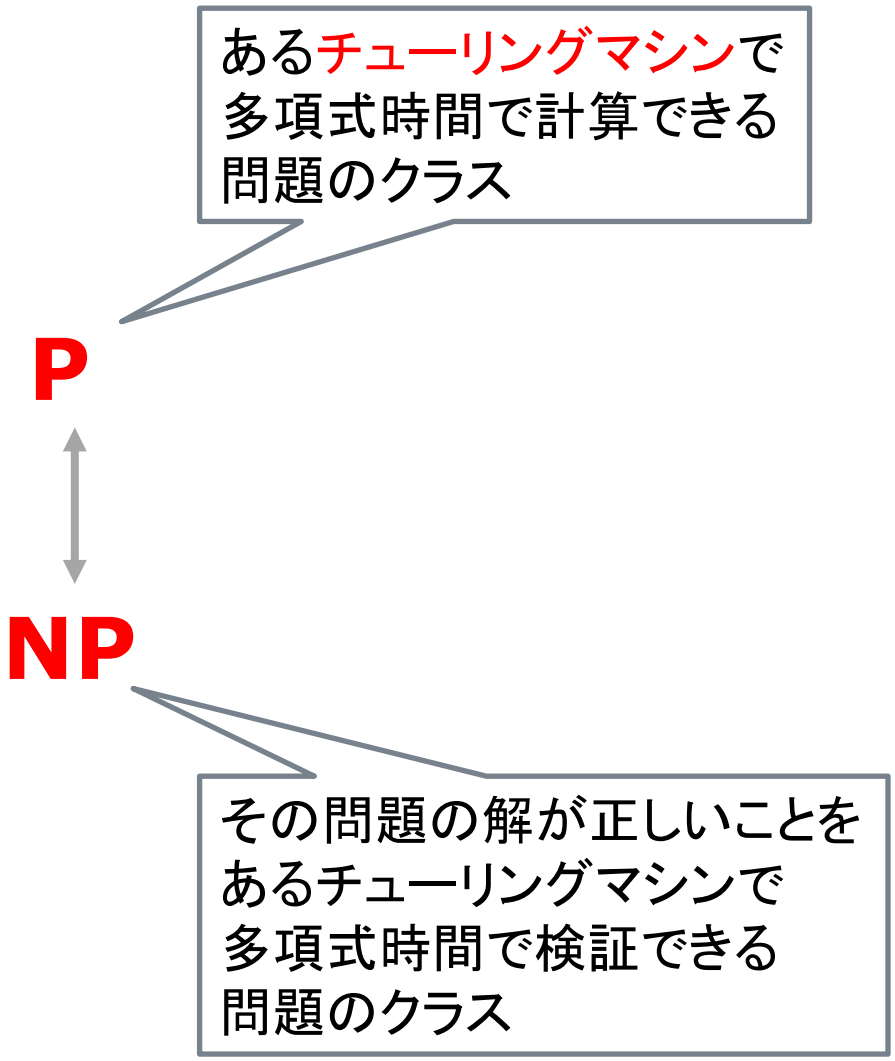
BQP (Bounded-Error Quantum Polynomial-Time): The class of problems solvable efficiently by a quantum computer, defined by Bernstein and Vazirani in 1993

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



基本的な複雑性のクラス

古典複雑性：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

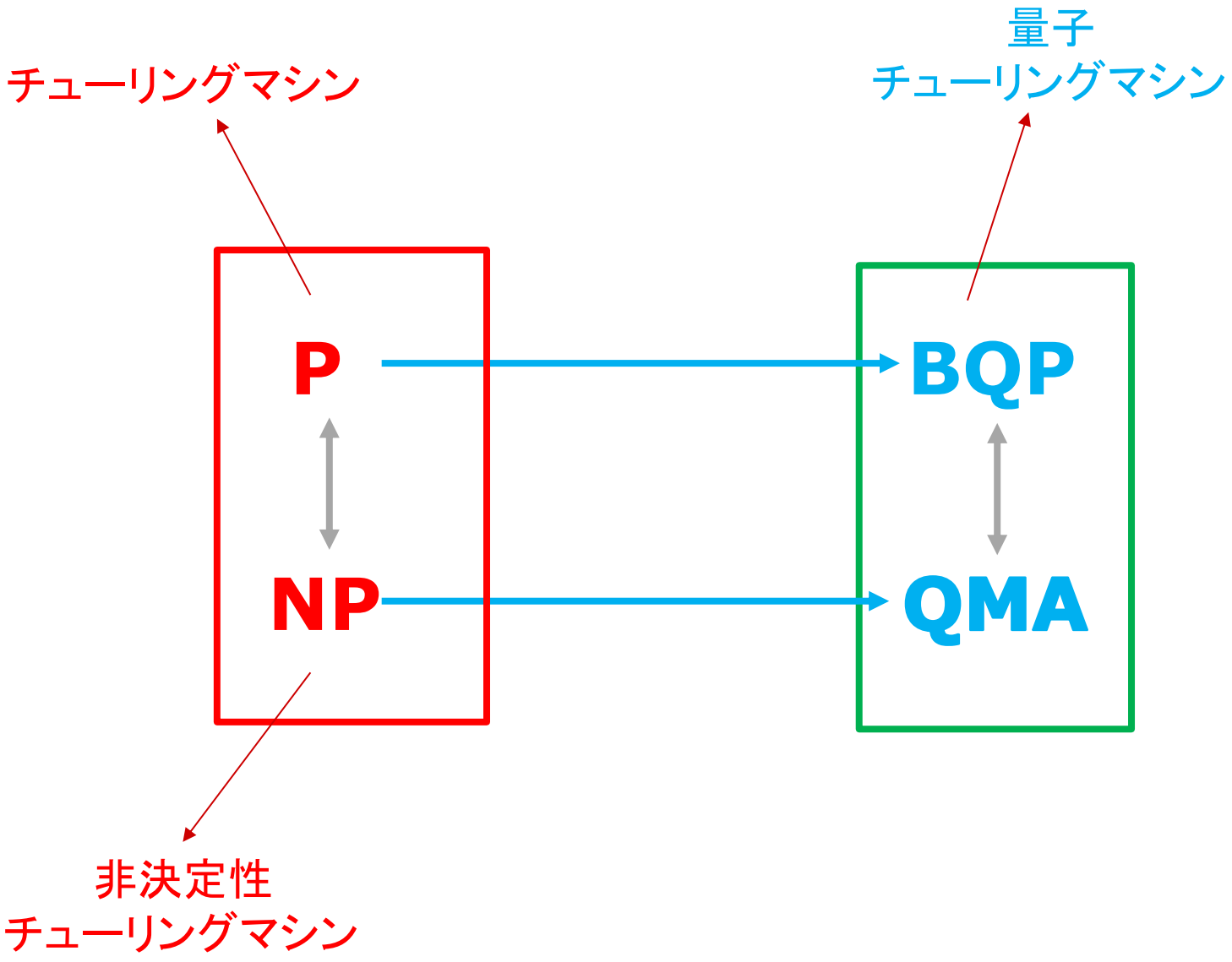
BQP：Bounded-Error Quantum
Polynomial-Time

QMA：Quantum MA

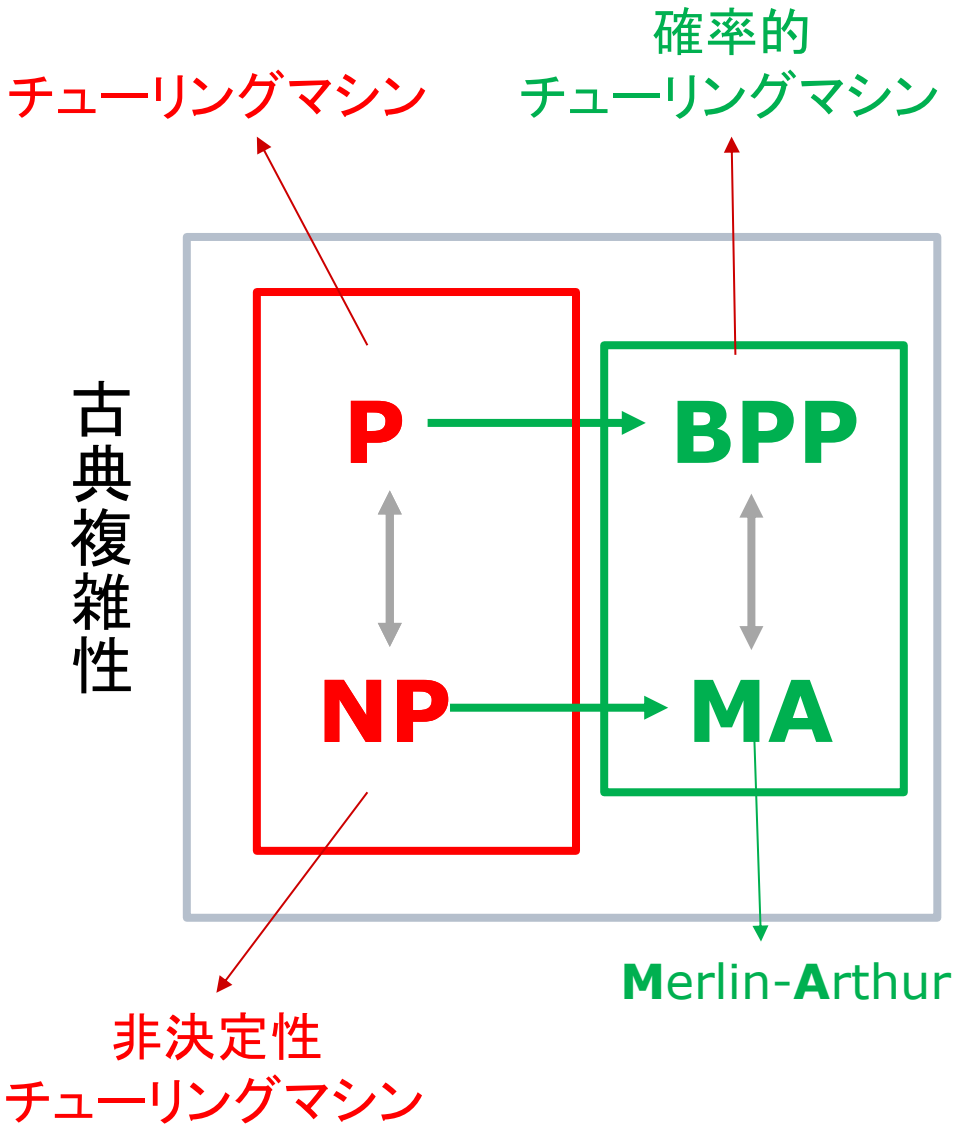
QMA

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

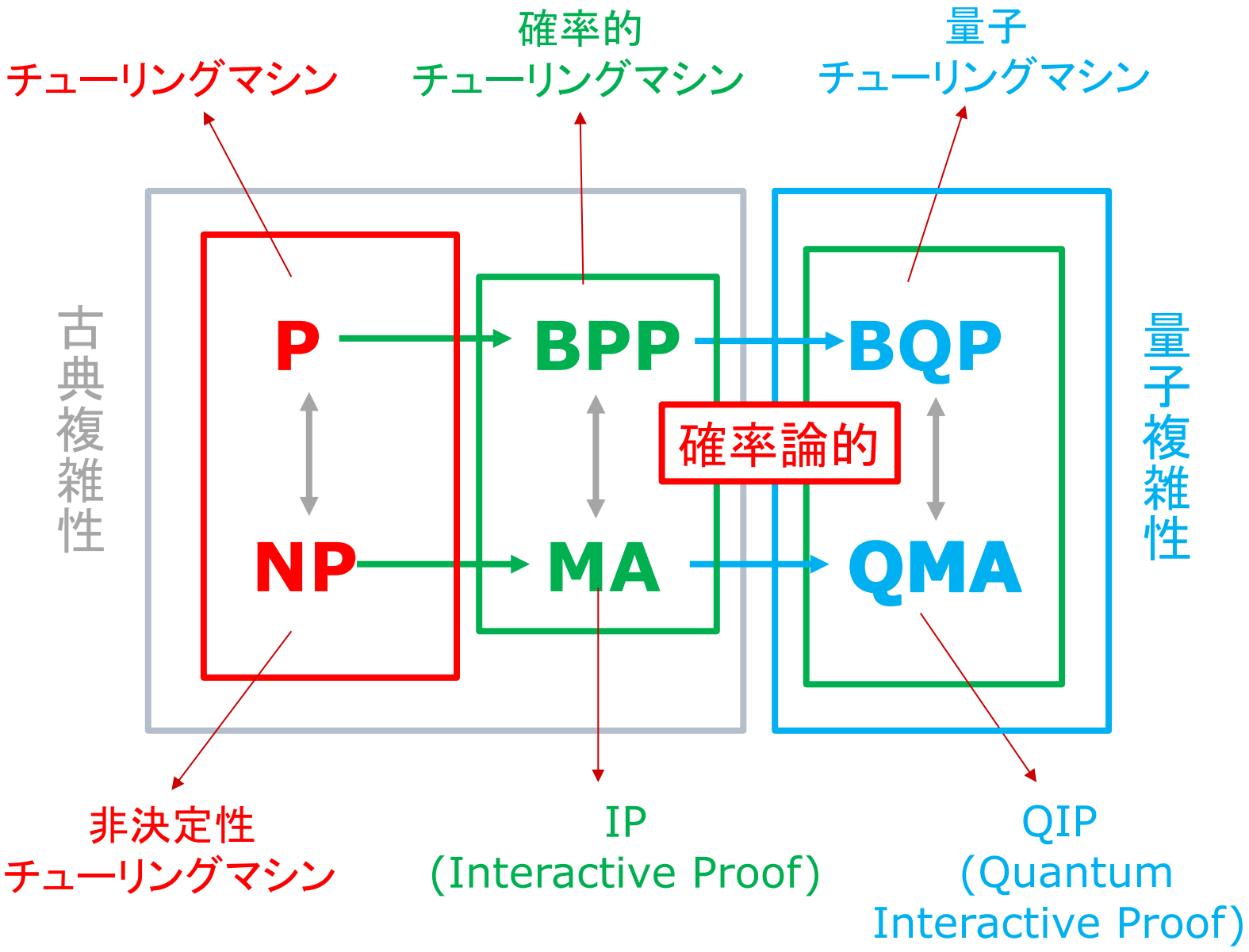
チューリングマシンと量子チューリングマシン



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




© NASA/ESA/CSA/Judy Schmidt


第三部　ラティス暗号の時代の始まり

暗号技術の現在 **Agenda**

第三部 ラティス暗号の時代の始まり

- 概説 －－ 主要なプレーヤー達
- ラティス入門
  - ラティスとは何か -- 格子点を表現する
  - 基底でラティスを定義する
  - ラティス問題
  - 同じラティスを生成する複数の基底

概説 –– 主要なプレーヤー達

ラティス暗号の時代の始まり

ポスト量子暗号の技術の中核はラティス暗号である。現代は、ラティス暗号の時代の始まりと考えてよい。

このセッションでは、ラティス暗号の現在までの発展のステージを、次の四つにまとめた。

- One way functionへのLattice problemの利用
- 先行したLattice 暗号の実装
- LWE(Learning with errors)
- Homomorphic encryption

あわせて、代表的なプレーヤーを紹介しようと思う。

One way functionへの
Lattice problemの利用

Generating Hard Instances of Lattice Problems

M. Ajtai, Generating Hard Instances of Lattice Problems, Proceedings 28th Annual ACM Symposium on Theory of Computing, 1996

https://dl.acm.org/doi/pdf/10.1145/237814.237838

Miklós Ajtai

Knuth Prize (2003)

https://en.wikipedia.org/wiki/Mikl%C3%B3s_Ajtai

A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence

M. Ajtai, C. Dwork, A Public-Key Cryptosystem with Average-Case/Worst-Case Equivalence, Electrom”c Colloquium on Computational Complexity TR96-065,

https://dl.acm.org/doi/pdf/10.1145/258533.258604

Cynthia Dwork

Dijkstra Prize (2007)
Gödel Prize (2017)
Knuth Prize (2020)

https://en.wikipedia.org/wiki/Cynthia_Dwork

先行したLattice 暗号の実装

- NTRU暗号
- GGH暗号

NTRU: A Ring-Based Public Key Cryptosystem

Jerey Hostein, Jill Pipher, Joseph H. Silverman

NTRU: A Ring-Based Public Key Cryptosystem

https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.25.8422&rep=rep1&type=pdf

Jill Pipher

president of the
American Mathematical Society

https://en.wikipedia.org/wiki/Jill_Pipher

Public-key cryptosystems from lattice reduction problems

Goldreich, Oded; Goldwasser, Shafi; Halevi, Shai (1997). "Public-key cryptosystems from lattice reduction problems". CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology. London: Springer-Verlag. pp. 112–131.

https://www.wisdom.weizmann.ac.il/~oded/PSX/pkcs.pdf

Shafi Goldwasser

Gödel Prize (1993, 2001)
Turing Award (2012)

https://en.wikipedia.org/wiki/Shafi_Goldwasser

Learning with errors

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

Oded Regev, "On lattices, learning with errors, random linear codes, and cryptography," in Proceedings of the thirty-seventh annual ACM symposium on Theory of computing (Baltimore, MD, USA: ACM, 2005)

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

Oded Regev

Gödel Prize (2018)

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

Gödel Prize (2018)

Regev's work has ushered in a revolution in cryptography, in both theory and practice. On the theoretical side, LWE has served as a simple and yet amazingly versatile foundation for nearly every kind of cryptographic object imaginable—along with many that were unimaginable until recently, and which still have no known constructions without LWE. Toward the practical end, LWE and its direct descendants are at the heart of several efficient real-world cryptosystems.

NIST IR 8413 での評価

Table 4. Algorithms to be Standardized

Public-Key Encryption/KEMs	Digital Signatures
CRYSTALS–KYBER	CRYSTALS–Dilithium
	FALCON
	SPHINCS$^+$

Homomorphic encryption

Fully homomorphic encryption using ideal lattices

Craig Gentry. Fully Homomorphic Encryption Using Ideal Lattices. In the 41st ACM Symposium on Theory of Computing (STOC), 2009.

https://dl.acm.org/doi/10.1145/1536414.1536440

Craig Gentry

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

ラティス入門

ラティスとは何か -- 格子点を表現する

ラティスとは何か

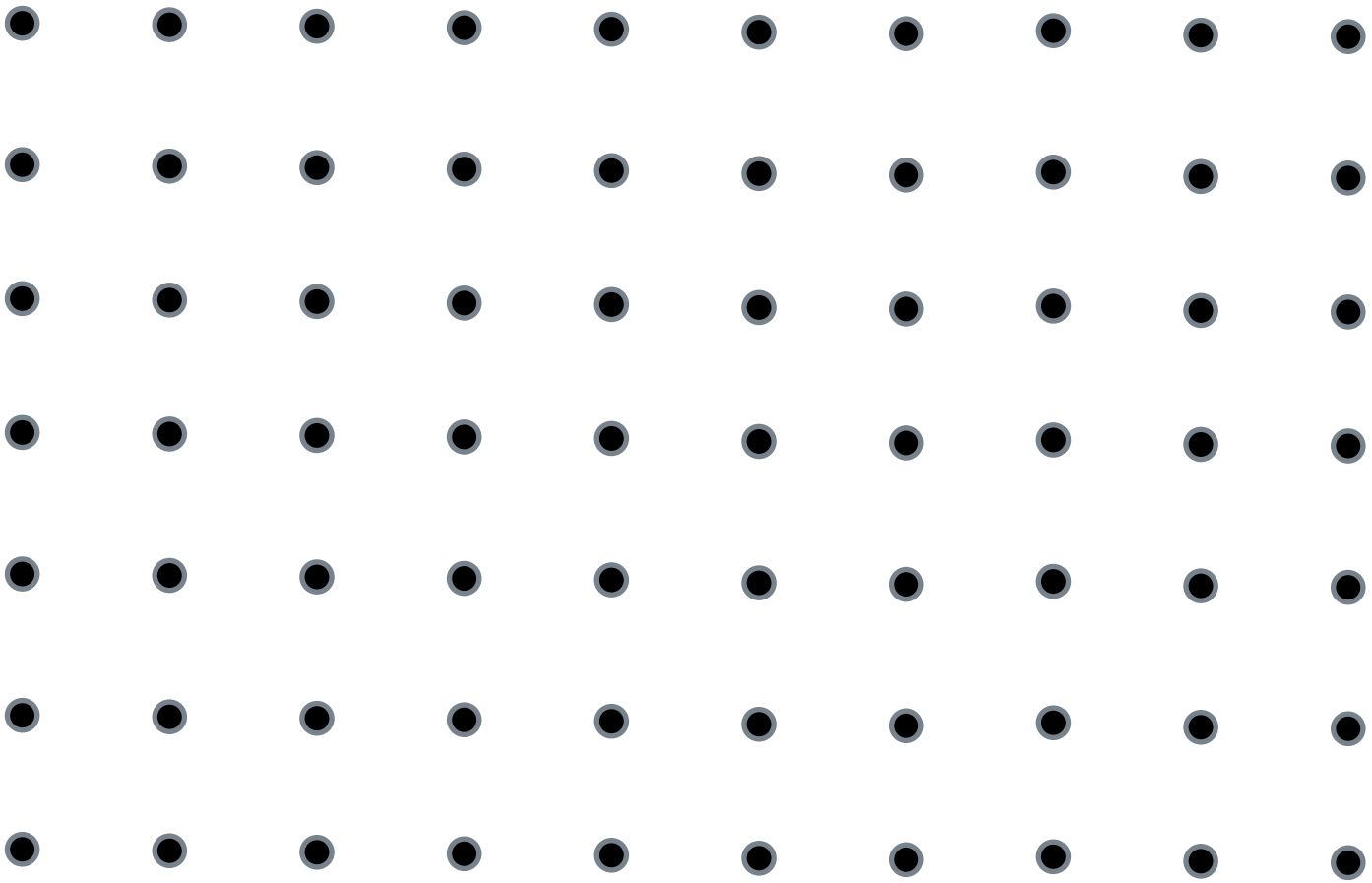
ラティスとは、「格子」のことである。

ラティスとは、空間上に規則的に格子状に配置された点の集合である。

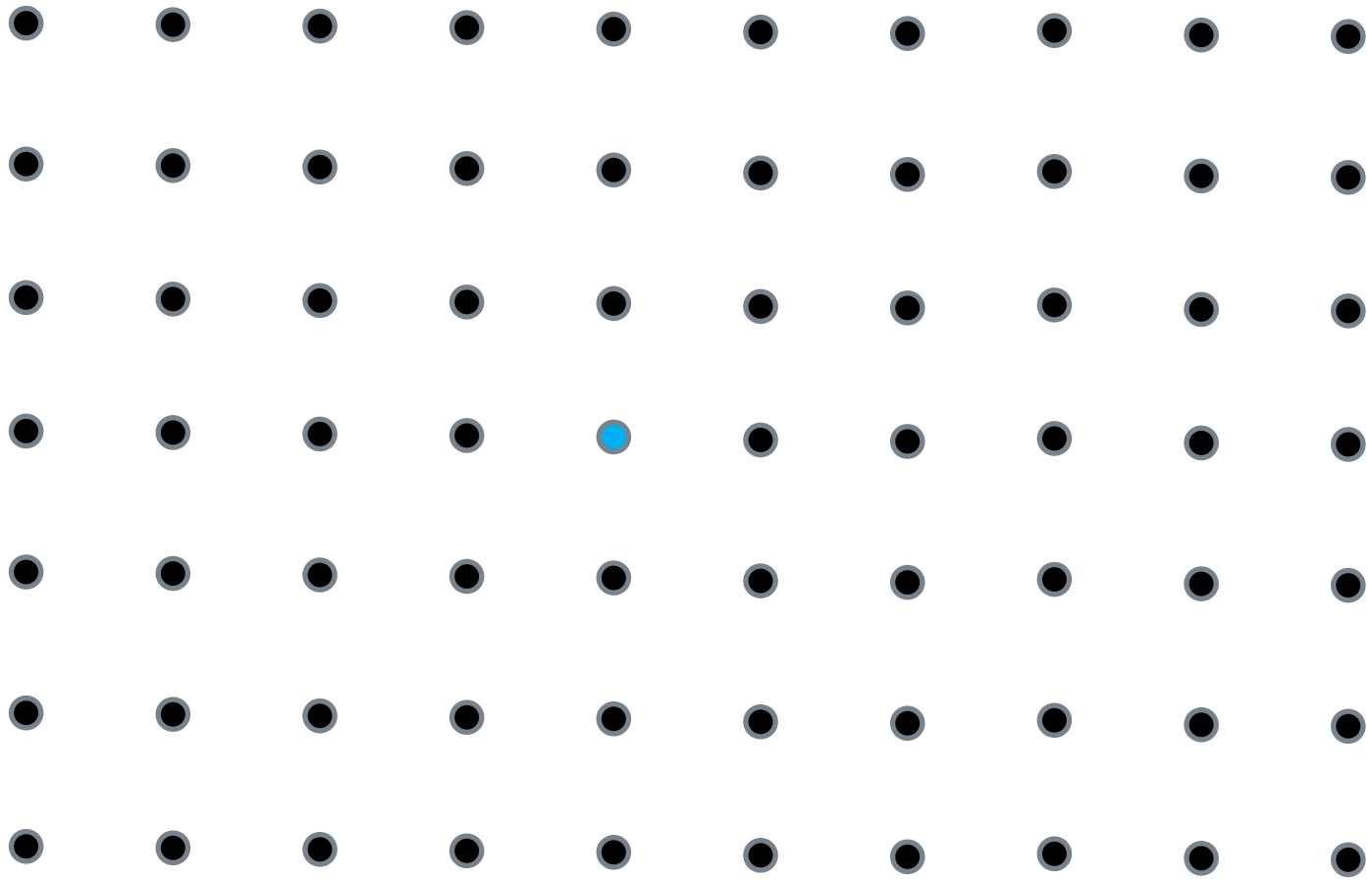
ここでは、こうした格子点の集合をどのように特徴づけるかを考える。

次の図の点の集合は、ラティスである。

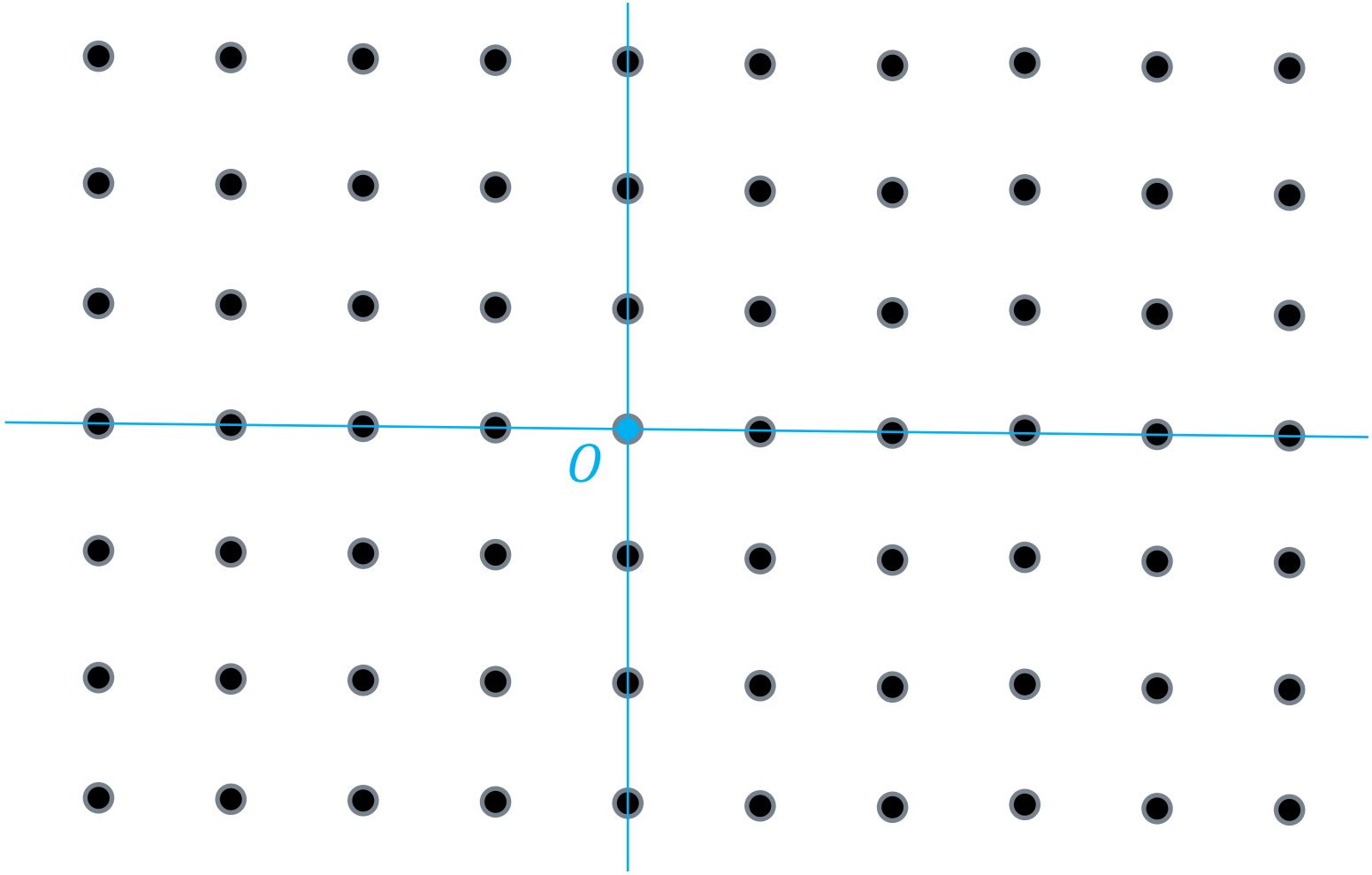
ラティスの例



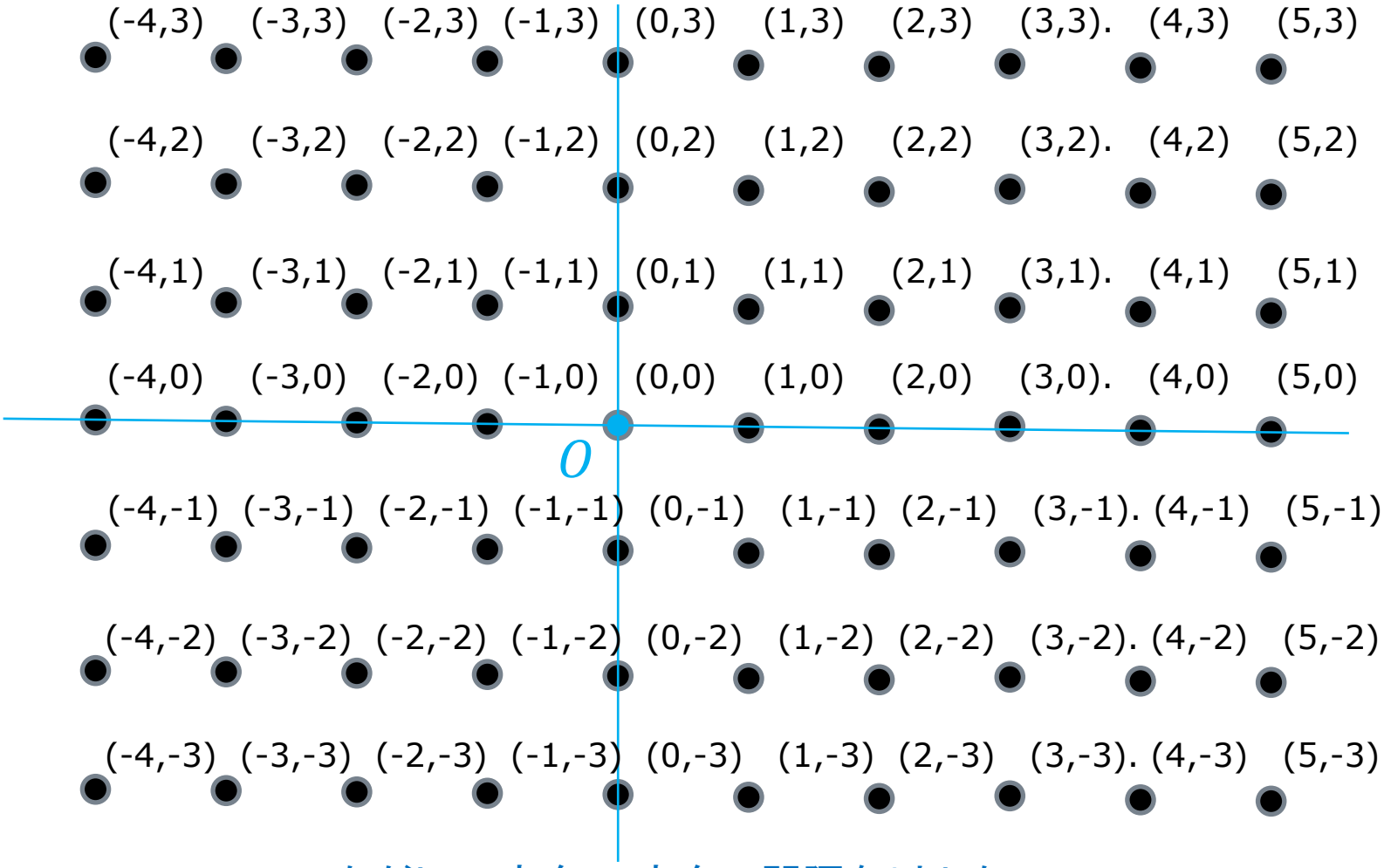
この点の集合の「規則正しさ」を考える



この点の集合の「規則正しさ」を考える
それぞれの点に座標を入れて見る



この点の集合の「規則正しさ」を考える
それぞれの点に座標を入れて見る



ただし、x方向、y方向の間隔を1とした

この点の集合の「規則正しさ」を考える
それぞれの点の座標は二つの整数の組である

(-4,3) (-3,3) (-2,3) (-1,3) (0,3) (1,3) (2,3) (3,3). (4,3) (5,3)

(-4,2) (-3,2) (-2,2) (-1,2) (0,2) (1,2) (2,2) (3,2). (4,2) (5,2)

(-4,1) (-3,1) (-2,1) (-1,1) (0,1) (1,1) (2,1) (3,1). (4,1) (5,1)

(-4,0) (-3,0) (-2,0) (-1,0) (0,0) (1,0) (2,0) (3,0). (4,0) (5,0)

(-4,-1) (-3,-1) (-2,-1) (-1,-1) (0,-1) (1,-1) (2,-1) (3,-1). (4,-1) (5,-1)

(-4,-2) (-3,-2) (-2,-2) (-1,-2) (0,-2) (1,-2) (2,-2) (3,-2). (4,-2) (5,-2)

(-4,-3) (-3,-3) (-2,-3) (-1,-3) (0,-3) (1,-3) (2,-3) (3,-3). (4,-3) (5,-3)

ただし、x方向、y方向の間隔を1とした

$\mathbb{Z}^2$ は、ラティスである

平面上の格子点の座標は、二つの整数の組である。

整数を $\mathbb{Z}$で表す。

$$\mathbb{Z} = \{ \ldots, -4, -3, -2, -1, 0, 1, 2, 3, 4 \ \ldots \}$$

整数の二つの組を、$\mathbb{Z} \times \mathbb{Z} = \mathbb{Z}^2$ と表す。

$\mathbb{Z}^2$ は、ラティスである。

一般に、n次元の空間で、$\mathbb{Z}^n$ は、ラティスである。

ラティスとその「基底」

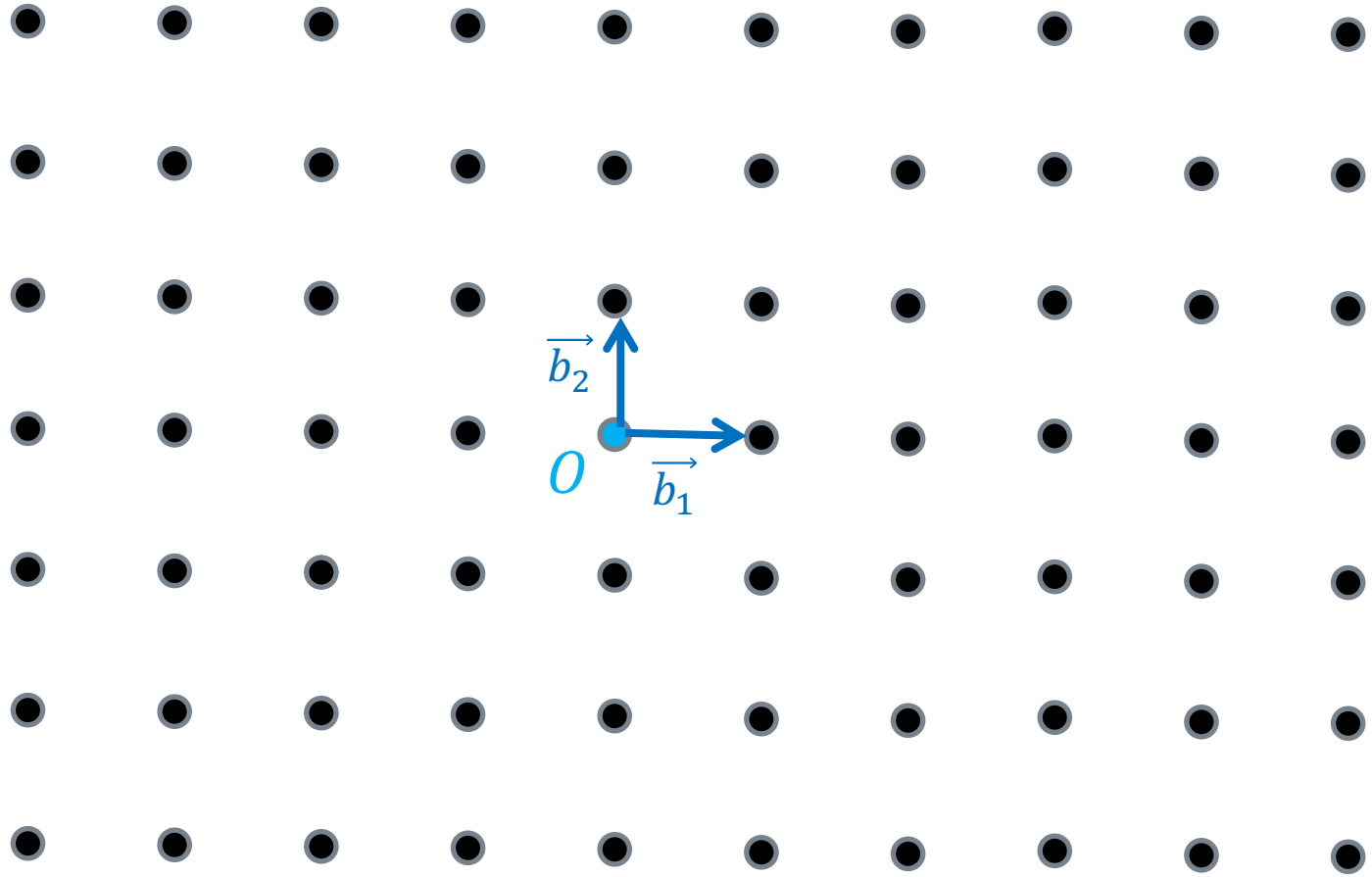
先に、二つの整数の組としてラティスを見てきたが、ここでは、他の捉え方もしてみよう。

それは、二つのベクトル $\overrightarrow{b_1}$, $\overrightarrow{b_2}$ の線型結合のベクトルとして、ラティスの点集合を捉えることである。

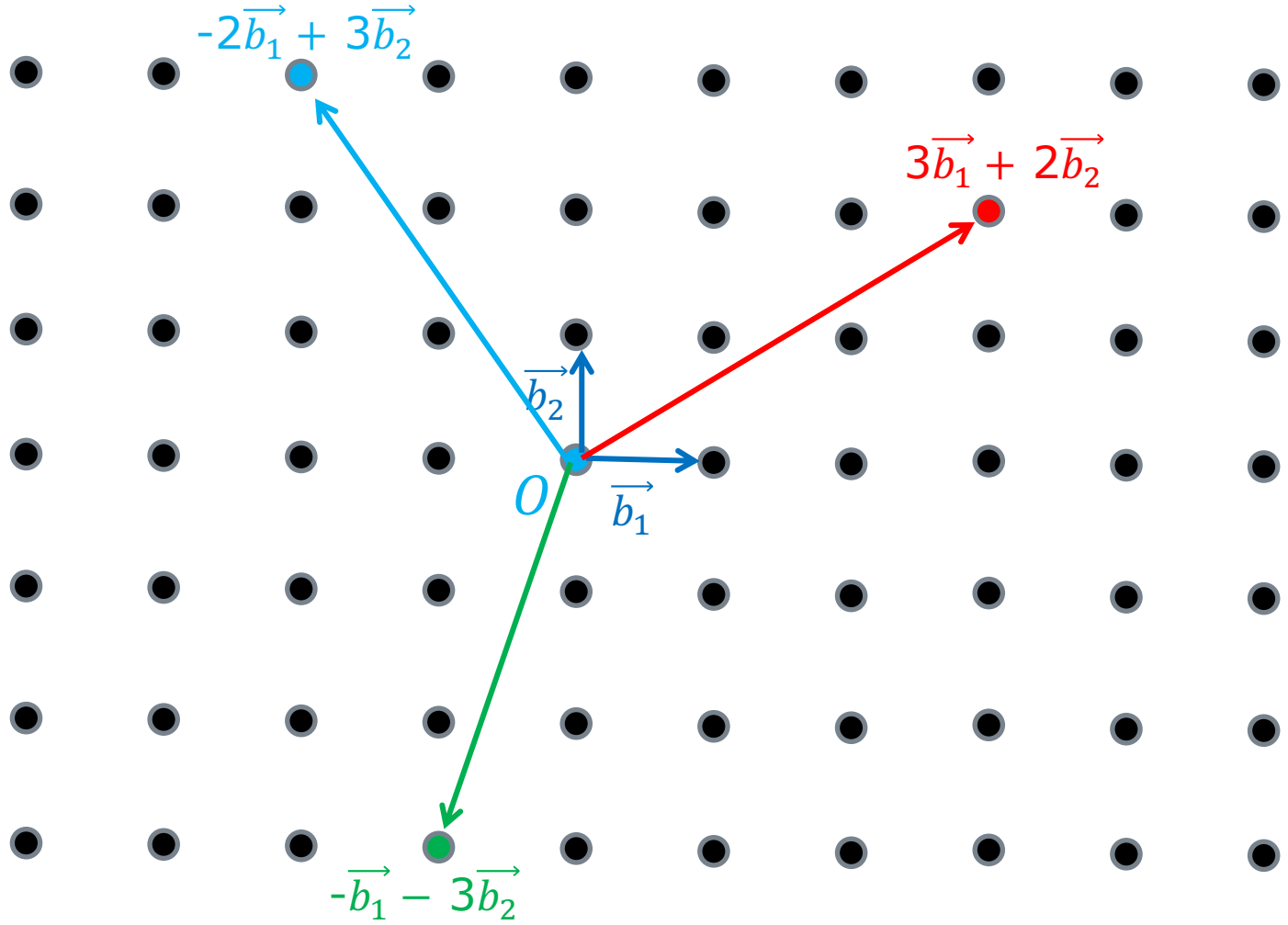
ラティス上の全ての点がそうした表現を持つ時、ベクトル $\overrightarrow{b_1}$, $\overrightarrow{b_2}$をそのラティスの「基底」という。

ベクトル $\overrightarrow{b_1}=(1,0)$, $\overrightarrow{b_2}=(0,1)$ は、先のラティスの基底である。

基底 $\overrightarrow{b_1}$=(1,0), $\overrightarrow{b_2}$=(0,1)

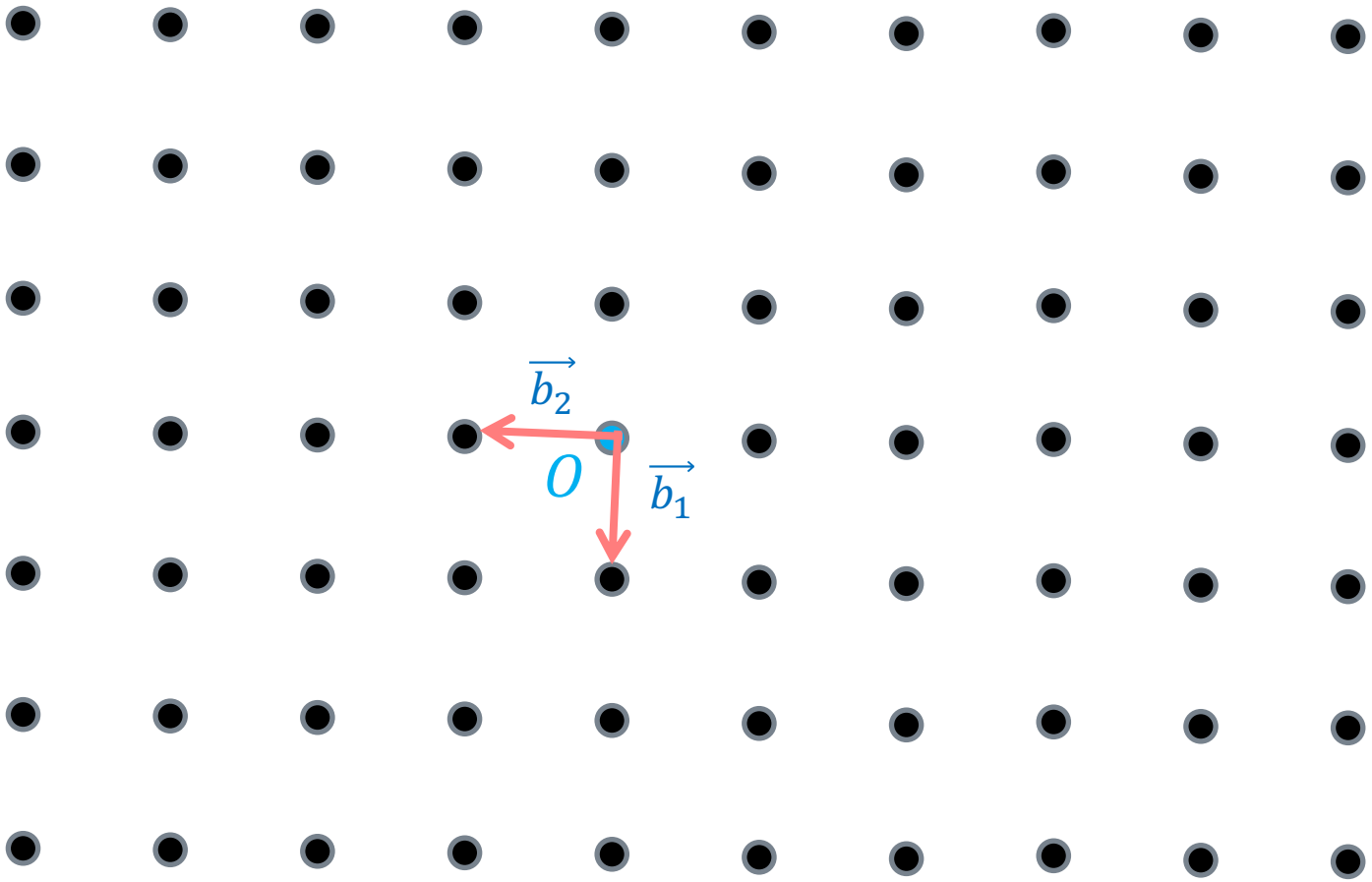


基底 $\overrightarrow{b_1}$=(1,0), $\overrightarrow{b_2}$=(0,1) の線型結合

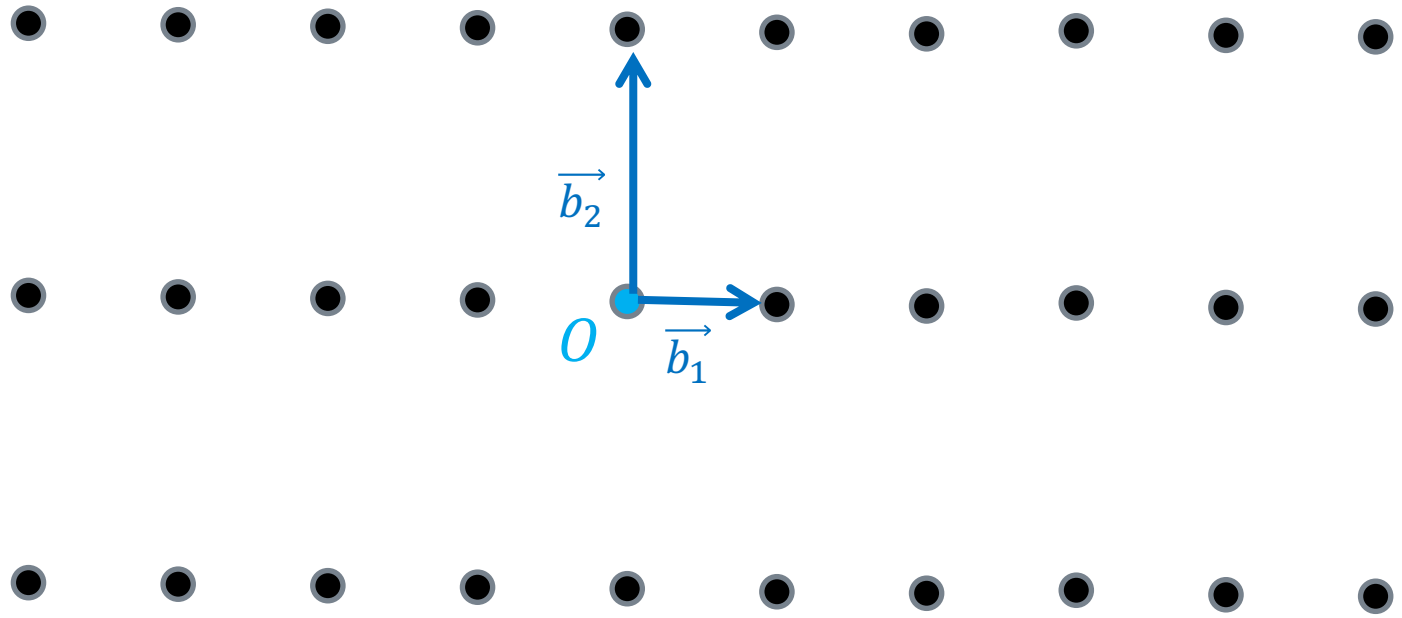


基底 は一つではない

$\overrightarrow{b_1}$=(0,-1), $\overrightarrow{b_2}$=(-1,0) も基底である

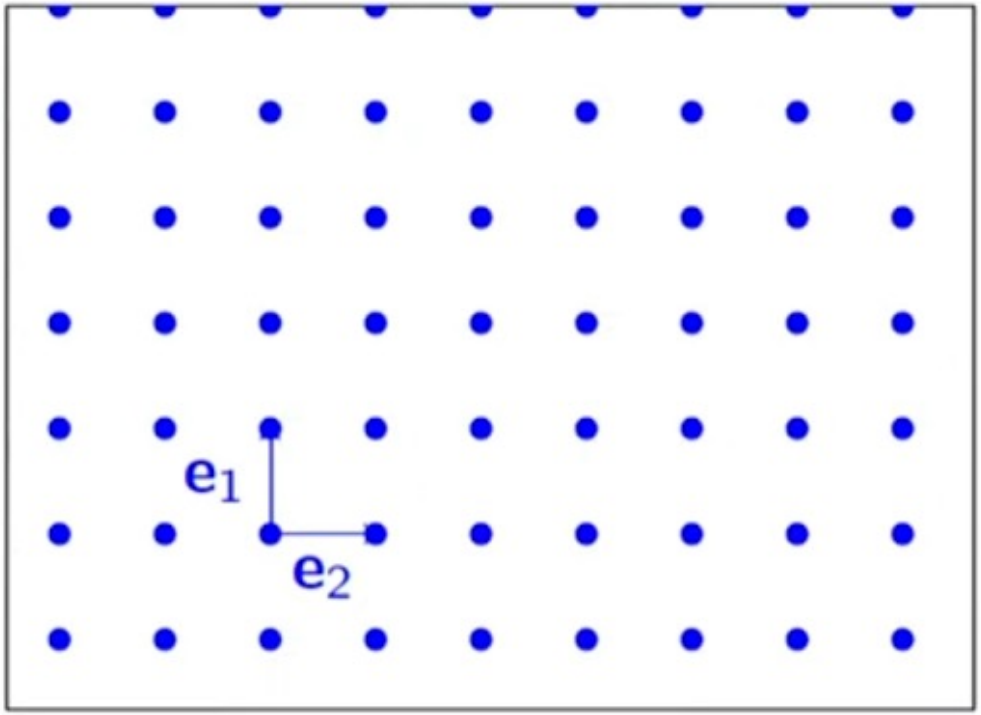


基底 $\overrightarrow{b_1}$=(1,0), $\overrightarrow{b_2}$=(0,2) から作られるラティス



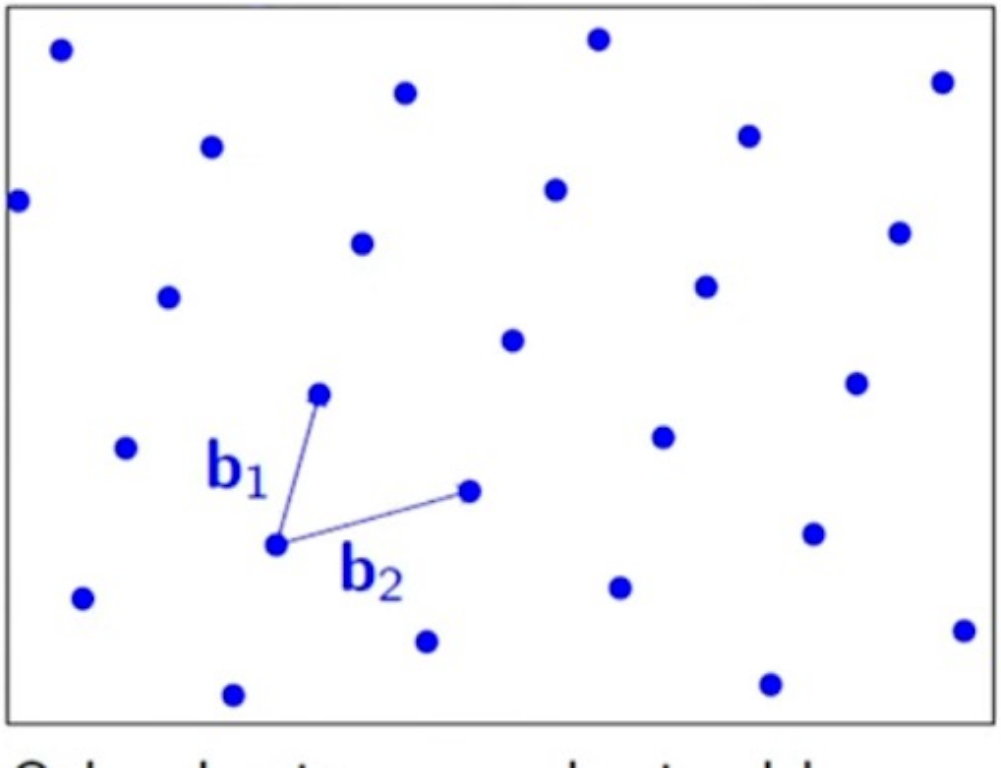
基底でラティスを定義する

前回見た単純なラティス



基底 e_1, e_2 で定義されるラティスLを $L = L(e_1, e_2)$ と表わそう。
正規直交基底 e_1, e_2 で定義されるラティスは、$\mathbb{Z}^2$ に等しい。
一般のn次元では、正規直交基底で定義されるラティスは、$\mathbb{Z}^n$ になる 。

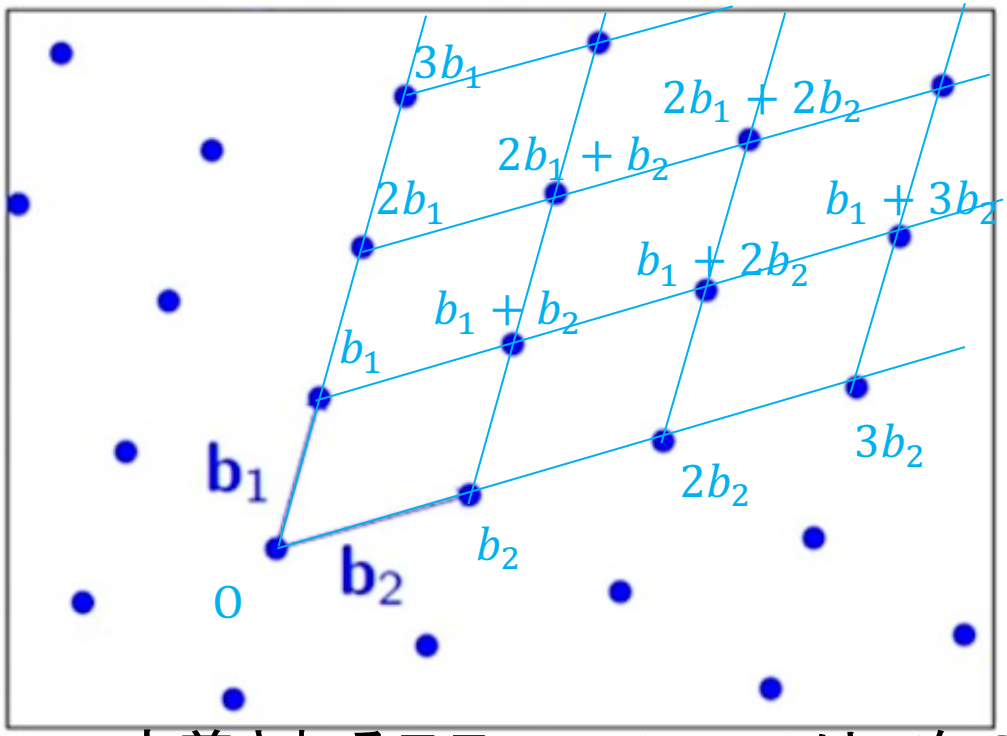
基底は、直交する単位ベクトルとは限らない



基底は、直交する単位ベクトルとは限らない。

上の図形は、基底 b_1, b_2で定義されるラティス$\mathrm{L}(b_1, b_2)$ である。

ラティスは、基底の整数倍の和で表わされる



この基底$b_1.b_2$で定義されるラティス$L(b_1.b_2)$は、次のような格子点から構成される。

$$o, b_2, 2b_2, 3b_2, \ldots, b_1, b_1 + b_2, b_1 + 2b_2, b_1 + 3b_2, \ldots$$

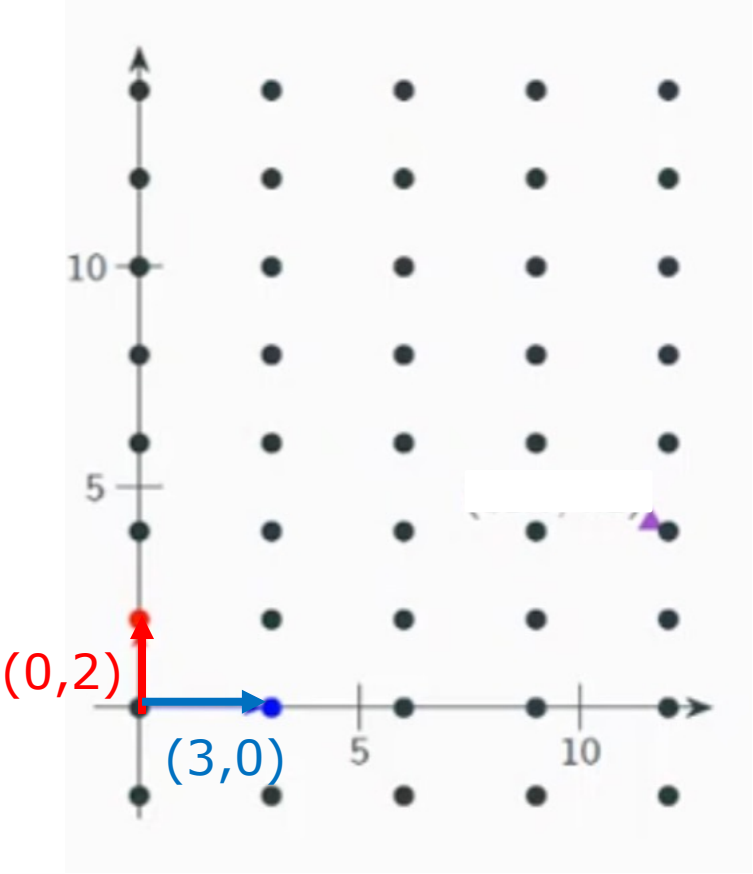
ラティスは、基底の整数倍の和で表わされる

ラティスと基底 – 一般形

n次元のラティスの基底を$b_1, b_2, \ldots, b_n$とする。
ラティスの各点は、n個の基底$b_1, b_2, \ldots, b_n$の整数倍の和で表される。

$$L(b_1, b_2, \ldots, b_n) = \sum_{i=1}^{n} b_i \mathbb{Z}$$

同一のラティスでも、複数の基底がある

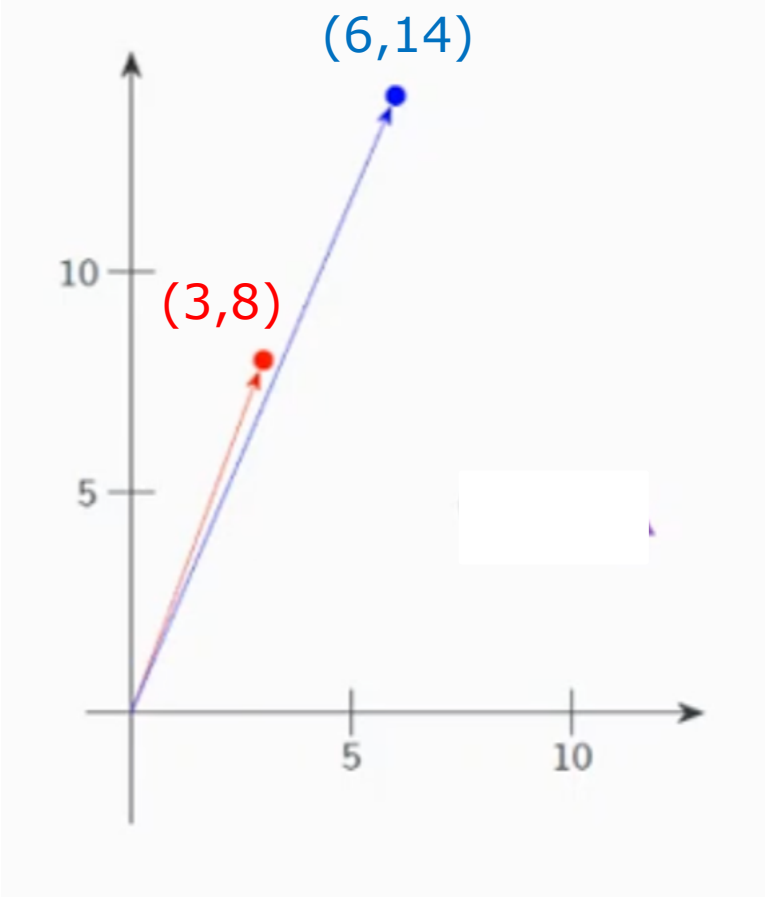


左のラティスには、
(3, 0) と(0,2) という基底があるのはすぐわかる。

ただ、同一のラティスでも、基底は複数存在する。

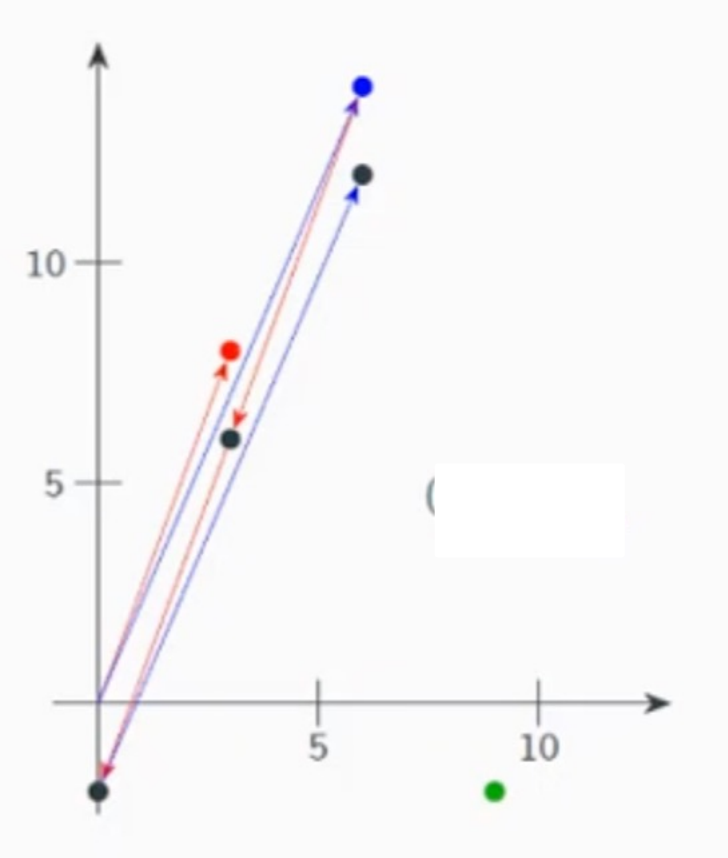
次のものは、このラティスの基底である。

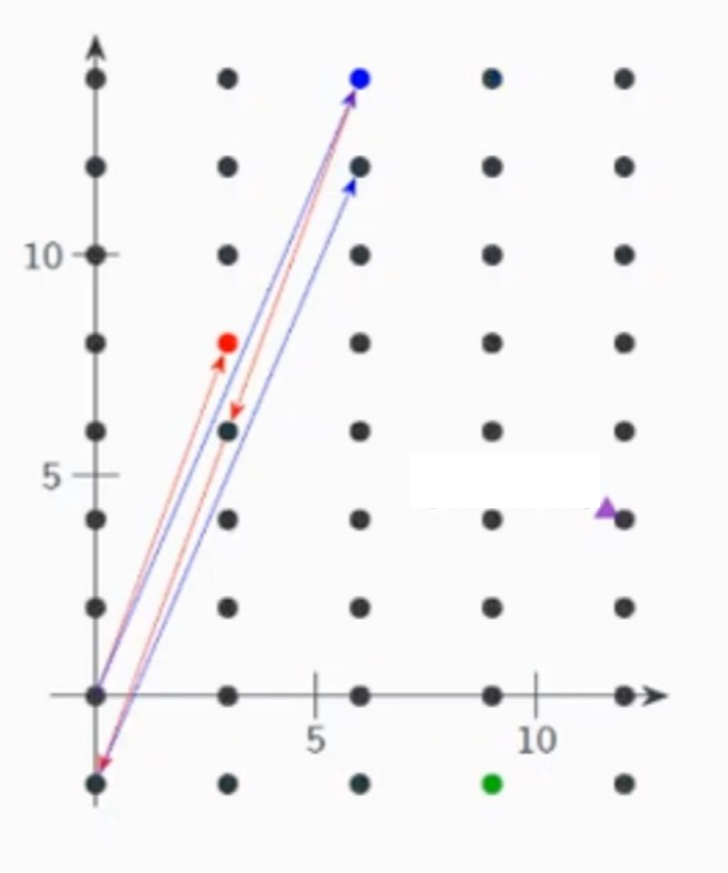
もう一つの基底



この基底 (3,8) , (6,14) は、次のように、先のラティスの各点をすべてカバーする。

先のラティスの各点をすべてカバーする。





ラティス問題 – SVPとCVP

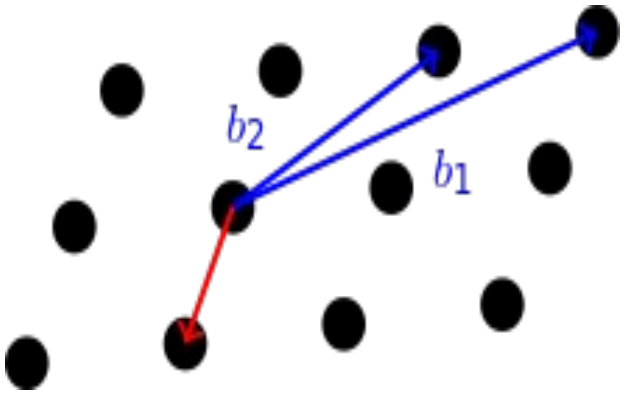
ラティス問題

ラティスに関して、いくつか基本的な問題がある。ここでは次の二つの問題を紹介する。

- Shortest vector problem (SVP)
- Closest vector problem (CVP)

これらの問題は、次元が高くなると一般には解くのが難しい。その難しさが、ラティス暗号の基礎になっている。

Shortest vector problem (SVP)

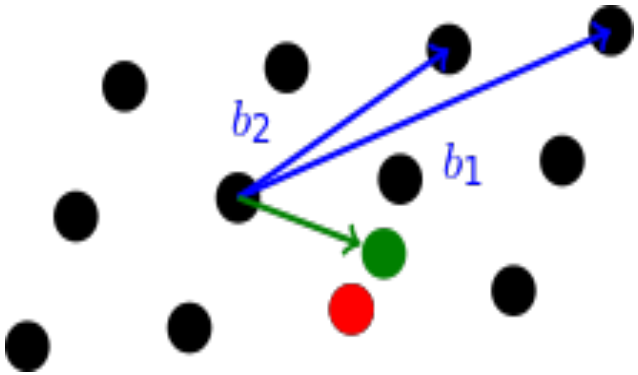


左の図のように基底b_1, b_2で張られるラティスがあるとする。

このラティス上の点で、互いに一番近い二点を求めよ。

（答は、赤い線で結ばれた二点である）

Closest vector problem (CVP)



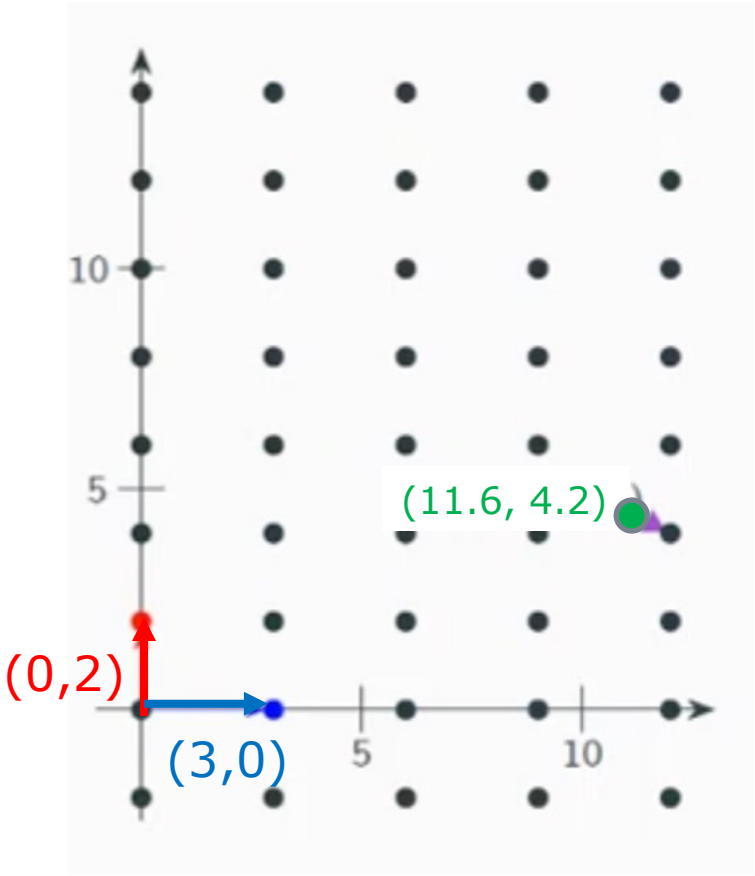
左の図のように基底b_1, b_2で張られるラティスがあるとする。

この平面上に、ラティスに属さない点を一つとる。（緑の点）

ラティス上の点で、この緑の点に一番近い点を求めよ。

（答は、赤い点である）

基底(3, 0) と(0,2)で張られるラティスで (11.6, 4.2) という点に一番近いラティスの点は？



$m(3,0) + n(0,2) = (11.6, 4.2)$

を解いてみる。

$3m = 11.6\,, 2n = 4.2$

$m = 3.87, n = 2.1$

m, n は整数だから、一番近い整数に丸めると、

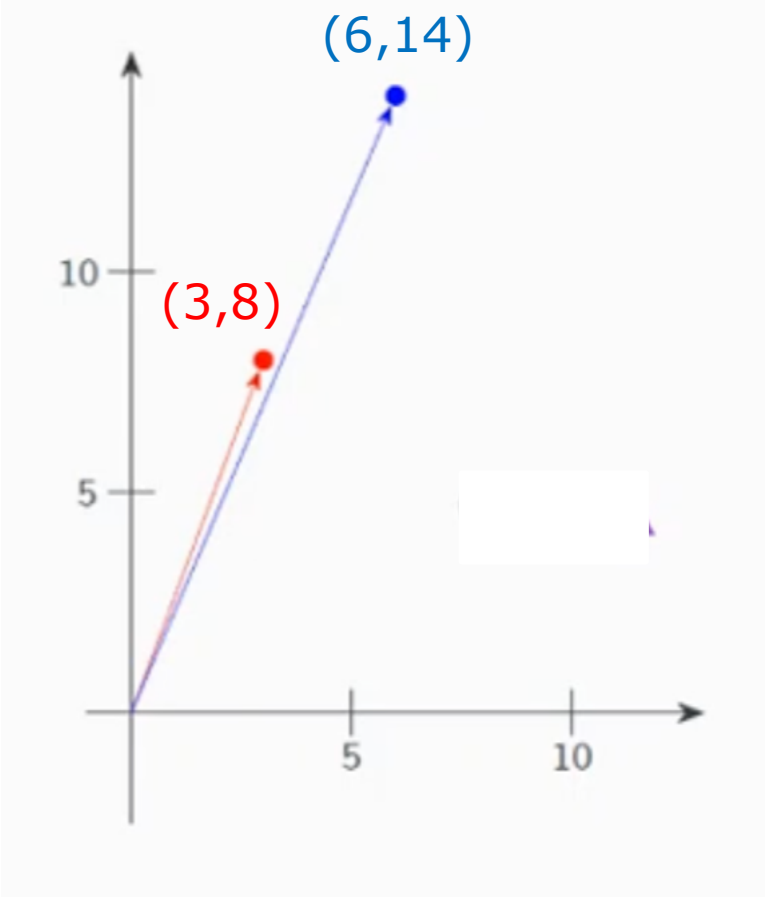
$m = 4, n = 2$。この時、

$4(3,0) + 2(0,2) = (12, 4)$

$\approx (11.6, 4.2)$

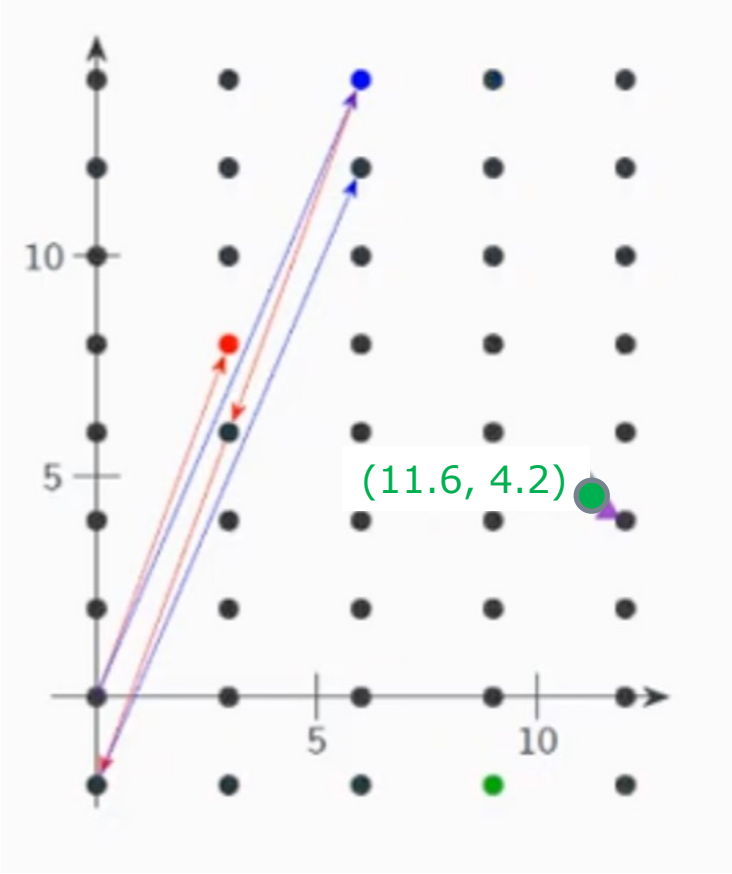
Babaiのアルゴリズム

もう一つの基底



この基底 (3,8) , (6,14) は、次のように、先のラティスの各点をすべてカバーする。

基底(3, 8) と(6,14)で張られるラティスで (11.6, 4.2) という点に一番近いラティスの点は？



$m(3,8) + n(6,14) = (11.6, 4.2)$

を解いてみる。

$$3m + 6n = 11.6$$
$$8m + 14n = 4.2$$

答えを整数に丸めると、

$m = -23, n = 13$。

この時、

$$-23(3,8) + 13(6,14)$$
$$= (9, -2)$$

さっきの点と違う。

$$-24(3,8) + 14(6,14)$$
$$= (12, 4)$$

で、先の結果と一致する。

良い基底と悪い基底

Babaiのアルゴリズムの整数近似解で Closest vector が正しく求まる基底を「良い基底」、そうでないなら「悪い基底」と、仮に呼ぼう。

どうやら、二つのベクトルの角度が大きい時に「良い基底」に、二つのベクトルの角度が小さい時に「悪い基底」になりそうである。

二つのベクトルが作る角度θ
（二次元の場合）

ベクトルを成分で表して$a = (a_1, a_2), b = (b_1, b_2)$とする。

ベクトルa, bの長さを$|a|, |b|$とすると

$$|a|^2 = a_1^2 + a_2^2,\ \ |b|^2 = b_1^2 + b_2^2$$

ベクトルa, bの内積$a \cdot b$は、ベクトルa, bが作る角度をθとすれば、

$$a \cdot b = |a||b|cos\theta = a_1 b_1 + a_2 b_2$$

$$cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a_1 b_1 + a_2 b_2}{\sqrt{a_1^2 + a_2^2}\sqrt{\mathrm{b}_1^2 + \mathrm{b}_2^2}}$$

二つのベクトル(5,1),(-2,8)が作る角度θを求める

$$cos\theta = \frac{a \cdot b}{|a||b|} = \frac{5 \cdot (-2) + 1 \cdot 8}{\sqrt{5^2 + 1^2}\sqrt{(-2)^2 + 8^2}}$$

$$= \frac{-2}{\sqrt{26}\sqrt{68}}$$

$$\approx -0.05$$

$cos\theta$ は、ゼロに近いので、この基底(5,1),(-2,8)については、Babaiのアルゴリズムは機能しそうである。

先に失敗した基底(3, 8) と(6,14)が作る角度θを求める

$$cos\theta = \frac{a \cdot b}{|a||b|} = \frac{3 \cdot 6 + 8 \cdot 14}{\sqrt{3^2 + 8^2}\sqrt{6^2 + 14^2}}$$

$$= \frac{18 + 112}{\sqrt{9 + 64}\sqrt{36 + 196}} = \frac{130}{\sqrt{73}\sqrt{232}} \approx \frac{130}{8.54 \cdot 15.2} = \frac{130}{129.8}$$

$$\approx 1$$

$cos\theta$ は、ほぼ1である。

この基底(3, 8) ,(6,14)については、Babaiのアルゴリズムは機能しそうもない。

同じラティスを生成する複数の基底

同じラティスを生成する「等価」な基底

基底 $b_1, b_2, \cdots, b_n$ から作られるラティスを、次のように表す。

$$L = L(b_1, b_2, \cdots, b_n)$$

ラティスLは、複数の基底を持つ。
もう一つの基底を $b'_1, b'_2, \cdots, b'_n$ とすれば、

$$L = L(b_1, b_2, \cdots, b_n) = L(b'_1, b'_2, \cdots, b'_n)$$

である。

この時、基底$(b_1, b_2, \cdots, b_n)$と基底$(b'_1, b'_2, \cdots, b'_n)$は、$L$について「等価」であると呼んで、次のように表そう。

$$(b_1, b_2, \cdots, b_n) \Leftrightarrow (b'_1, b'_2, \cdots, b'_n)$$

等価な基底の間には、どんな関係があるかを考えてみよう。

$(b_1, b_2, \cdots, b_n)$がLの基底であること

ベクトルのn個の並び $(b_1, b_2, \cdots, b_n)$がLの基底であるということは、次のことを意味する。

Lの全ての点は、n個の基底ベクトル$b_1, b_2, \cdots, b_n$と n個の整数 $n_1, n_2, \cdots, n_n$ で、次のように表される。

$$n_1 b_1 + n_2 b_2 + \cdots + n_n b_n$$

$n_1 b_1 + n_2 b_2 + \cdots + n_n b_n \in L(b_1, b_2, \cdots, b_n)$である
整数の組 $(n_1, n_2, \cdots, n_n) \in \mathbb{Z}^n$ が存在する。

基底の順番を入れ替えたものも基底である

$L(b_1, b_2, \cdots, b_i \cdots, b_j, \cdots, b_n)$ は

$n_1 b_1 + n_2 b_2 + \cdots + n_i b_i + \cdots + n_j b_j + \cdots + n_n b_n \in L$ である

整数の組 $(n_1, n_2, \cdots, n_n) \in \mathbb{Z}^n$ が存在するということ。

$$n_1 b_1 + n_2 b_2 + \cdots + n_i b_i + \cdots + n_j b_j + \cdots + n_n b_n$$
$$= n_1 b_1 + n_2 b_2 + \cdots + n_j b_j + \cdots + n_i b_i + \cdots + n_n b_n \in L$$

これから、

$(b_1, b_2, \cdots, b_i \cdots, b_j, \cdots, b_n) \Leftrightarrow (b_1, b_2, \cdots, b_j, \cdots, b_i, \cdots, b_n)$

が言える。

一つの基底の符号を変えたものも基底である

$L(b_1, b_2, \cdots, -b_i \cdots, , b_n)$ は
$n_1 b_1 + n_2 b_2 + \cdots + n_i(-b_i) + \cdots + b_n \in L$である
整数の組 $(n_1, n_2, \cdots, n_n) \in \mathbb{Z}^n$ が存在するということ。

$$n_1 b_1 + n_2 b_2 + \cdots + n_i(-b_i) + \cdots + n_j b_n =$$
$$n_1 b_1 + n_2 b_2 + \cdots + (-n_i) b_i + \cdots + n_n\, b_n \in L$$

これから、

$$(b_1, b_2, \cdots, b_i, \cdots, b_n) \Leftrightarrow (b_1, b_2, , \ldots, -b_i, \cdots, b_n)$$

がいえる。

基底の一つを，その基底に別の基底の整数倍を加えたものに置き換えたものも基底である

$L(b_1, b_2, \cdots, b_i \cdots, , b_n)$ は

$n_1 b_1 + n_2 b_2 + \cdots + n_i b_i + \cdots + n_j b_j + \cdots + n_n b_n \in L$ である

整数の組 $(n_1, n_2, \cdots, n_n) \in \mathbb{Z}^n$ が存在するということ。

$$n_1 b_1 + n_2 b_2 + \cdots + n_i (b_i + k b_j) + \cdots + n_j b_j + \cdots + n_n b_n =$$
$$n_1 b_1 + n_2 b_2 + \cdots + n_i b_i + \cdots + (n_i k + 1) b_j + \cdots + n_j b_n \in L$$

これから、kを整数とすれば、

$$(b_1, b_2, \cdots, b_i, \cdots, b_n) \Leftrightarrow (b_1, b_2, , \ldots, b_i + k b_j, \cdots, b_n)$$

がいえる。

基底を行列で表現する

基底を構成するベクトルを列ベクトルで表し、それを並べて行列を作る。ラティスの基底は、$n \times n$の行列Bで表現される。

$L = L(b_1, b_2, \cdots, b_n) = L(B)$ と表そう。

この時、先の等価な基底についてのルールは、等価性を保ったままでの基底を表現する行列の操作として、次のようにまとめられる。

1. 行列Bの列v_iをv_jと交換したものをB′とすると、$L(B) = L(B')$
2. 行列Bの一列v_iに-1に掛けたものをB′とすると、$L(B) = L(B')$
3. 行列Bの一列v_iを$v_i + kv_j$にかえたものをB′とすると、$L(B) = L(B')$

行列の操作で等価な基底を見つける

例えば、基底(3, 0) と(0,2)、基底(3, 8) と(6,14)は同じラティスを作る。

基底を列ベクトルで表すと

$$\mathrm{L}\left(\begin{pmatrix}3\\0\end{pmatrix}, \begin{pmatrix}0\\2\end{pmatrix}\right) = \mathrm{L}\left(\begin{pmatrix}3\\8\end{pmatrix}, \begin{pmatrix}6\\14\end{pmatrix}\right)$$

行列で基底を表し、

$$L\left(\begin{bmatrix}3 & 0\\0 & 2\end{bmatrix}\right) = L\left(\begin{bmatrix}3 & 6\\8 & 14\end{bmatrix}\right)$$

となることを、行列の操作で確かめてみよう。

行列の操作で等価な基底を見つける

$$\begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 3 & 0 \\ 8 & 2 \end{bmatrix}$$

1列＋4×2列

$$\begin{pmatrix} 3 \\ 8 \end{pmatrix} = \begin{pmatrix} 3 \\ 0 \end{pmatrix} + 4\begin{pmatrix} 0 \\ 2 \end{pmatrix}$$

$$\begin{bmatrix} 3 & 0 \\ 8 & 2 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 3 & 0 \\ 8 & -2 \end{bmatrix}$$

−1×2列

$$\begin{pmatrix} 0 \\ -2 \end{pmatrix} = -\begin{pmatrix} 0 \\ 2 \end{pmatrix}$$

$$\begin{bmatrix} 3 & 0 \\ 8 & -2 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 3 & 6 \\ 8 & 14 \end{bmatrix}$$

2×1列＋2列

$$\begin{pmatrix} 6 \\ 14 \end{pmatrix} = 2\begin{pmatrix} 3 \\ 8 \end{pmatrix} + \begin{pmatrix} 0 \\ -2 \end{pmatrix}$$

$$\begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 3 & 0 \\ 8 & 2 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 3 & 0 \\ 8 & -2 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 3 & 6 \\ 8 & 14 \end{bmatrix}$$


© NASA/ESA/CSA/Judy Schmidt