コンピュータと数学：
計算の複雑さから量子コンピュータへ

名古屋大学大学院多元数理科学研究科
ルガル フランソワ

日本数学会 市民講演会　２０２５年9月２０日

講演の流れ

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独やLINE友だち問題
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

講演の流れ

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独やLINE友だち問題
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

本日の主役： P ≠ NP 予想

P: コンピュータで速く解ける問題（の集合）

NP: コンピュータで解きたい問題（の集合）

P = NP の意味：解きたい問題はすべて速く解ける

P ≠ NP の意味：スパコンを使っても解けない問題は存在する

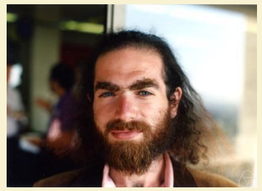
P ≠ NP 予想

ミレニアム懸賞問題

✓ アメリカのクレイ数学研究所によって、2000年に発表された7つの未解決問題

✓ 100万ドルの懸賞金がかけられている

- ❑ ヤン–ミルズ方程式と質量ギャップ問題
- ❑ リーマン予想
- ❑ ナビエ–ストークス方程式の解の存在と滑らかさ
- ❑ ホッジ予想
- ❑ ~~ポアンカレ予想~~ ⟵ グリゴリー・ペレルマンにより解決（2003年）
- ❑ BSD 予想
- ❑ P≠NP 予想



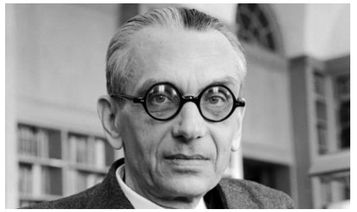
予想の歴史： P = NP の意味について

1956年にジョン・フォン・ノイマンへ手紙を送る

論理学のすべての問題を速く解く機会があれば（すなわち、P = NP ならば）

> “If there really were a machine with $\varphi(n) \sim k \cdot n$ (or even $\sim k \cdot n^2$), this would have consequences of the greatest importance. Namely, it would obviously mean that in spite of the undecidability of the Entscheidungsproblem, the mental work of a mathematician concerning Yes-or-No questions could be completely replaced by a machine.”
>
> **—Kurt Gödel, 1956**

数学の証明は機械で効率よく発見できる



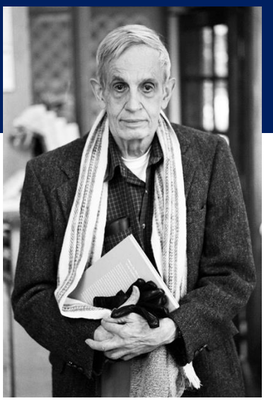
クルト・ゲーデル
(1906年 – 1978年)

P = NP が証明できれば

600万ドルももらえる

- ヤン–ミルズ方程式と質量ギャップ問題　**100万ドル**
- リーマン予想　**100万ドル**
- ナビエ–ストークス方程式の解の存在と滑らかさ **100万ドル**
- ホッジ予想　**100万ドル**
- ~~ポアンカレ予想~~
- BSD 予想　**100万ドル**
- P≠NP 予想　**100万ドル**

予想の歴史： P ≠ NP について



ジョン・ナッシュ
（1928年 – 2015年）
ノーベル経済学賞（1996年）
アーベル賞（2015年）

1955年にNSA（アメリカ国家安全保障局）へ手紙を送る

ほぼ全ての暗号化問題は

"Now my general conjecture is as follows: for almost all sufficiently complex types of enciphering, especially where the instructions given by different portions of the key interact complexly with each other in the determination of their ultimate effects on the enciphering, the mean key computation length increases exponentially with the length of the key, or in other words, the information content of the key ... The nature of this conjecture is such that I cannot *prove* it, even for a special type of ciphers. Nor do I expect it to be proven."—**John Nash, 1955**

私は証明できない

効率よく解けないだろう（すなわち、P ≠ NP）

証明されないだろう

まだ発見されていないアルゴリズムも含む！

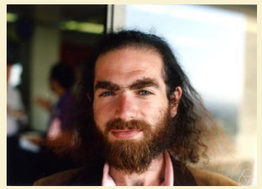
P ≠ NP 予想を証明するため、速い解法（アルゴリズム）がないと
証明しないといけないので、非常に難易度の高い数学の問題

ミレニアム懸賞問題

✓ アメリカのクレイ数学研究所によって、2000年に発表された7つの未解決問題

✓ 100万ドルの懸賞金がかけられている

- ❑ ヤン–ミルズ方程式と質量ギャップ問題
- ❑ リーマン予想
- ❑ ナビエ–ストークス方程式の解の存在と滑らかさ
- ❑ ホッジ予想
- ❑ ~~ポアンカレ予想~~ ⟸ グリゴリー・ペレルマンにより解決（2003年）
- ❑ BSD 予想
- ❑ P≠NP 予想



✓理解しやすい設問

✓毎年数件の誤った証明が発表される…

P-versus-NP ウェブサイト (1986–2016)

管理者：Gerhard Woeginger

https://www.win.tue.nl/~gwoegi/P-versus-NP.htm

116 件の誤った証明

The P-versus-NP page

Milestones

Note: The following paragraphs list many papers that try to contribute to the P-versus-NP question. Among all these papers, there is only **a single paper** that has appeared in a peer-reviewed journal, that has thoroughly been verified by the experts in the area, and whose correctness is accepted by the general research community: The paper by Mihalis Yannakakis. (And this paper does not settle the P-versus-NP question, but "just" shows that a certain approach to settling this question will never work out.)

1. **[Equal]:** In 1986/87 Ted Swart (University of Guelph) wrote a number of papers (some of them had the title: "*P=NP*") that gave linear programming formulations of polynomial size for the Hamiltonian cycle problem. Since linear programming is polynomially solvable and Hamiltonian cycle is NP-hard, Swart deduced that P=NP.
   In 1988, Mihalis Yannakakis closed the discussion with his paper "Expressing combinatorial optimization problems by linear programs" (Proceedings of STOC 1988, pp. 223-228). Yannakakis proved that expressing the traveling salesman problem by a symmetric linear program (as in Swart's approach) requires exponential size. The journal version of this paper has been published in Journal of Computer and System Sciences 43, 1991, pp. 441-466.

2. **[Equal]:** The 1996 issue (Volume 1, 1996, pp. 16-29) of the "SouthWest Journal of Pure and Applied Mathematics" (SWJPAM) contains the article "*Polynomial-Time Partition of a Graph into Cliques*" by the Ukrainian mathematician Anatoly Plotnikov. This article designs a polynomial time algorithm for an NP-hard graph problem, and thus proves P=NP.
   (SWJPAM is an electronic journal devoted to all aspects of Pure and Applied mathematics, and related topics. Authoritative expository and survey articles on subjects of special interest are also welcomed. SWJPAM serves as an international forum for the publication of high-quality strictly peer-reviewed original research articles. The article is usually sent to at least two experts in the area. Two positive reviews are required for the acceptance and publication of any submitted article.)

3. **[Equal]:** Around 1997 Tang Pushan provided a polynomial algorithm for the clique problem. The two relevant papers are "*An algorithm with polynomial time complexity for finding clique in a graph*" by Tang Pushan (Proceedings of 5th International Conference on CAD&CG, Shenzhen, P.R. China, 1997, pp 500-505) and "*HEWN: A polynomial algorithm for CLIQUE problem*" by Tang Pushan and Huang Zhijun (Journal of Computer Science & Technology 13(Supplement), 1998, pp 33-44). Clearly this implies P=NP. Zhu Daming, Luan Junfeng and M. A. Shaohan (all affiliated with Shandong University, China) refute these claims in their paper *"Hardness and methods to solve CLIQUE"* (Journal of Computer Science and Technology 16, 2001, pp 388-391).

112. **[Not equal]:** In February 2016, Mathias Hauptmann showed that P is not equal to NP. Hauptmann starts from the assumption that P equals Sigma-2-p, proves a new variant of the Union Theorem of McCreight and Meyer for Sigma-2-p, and eventually derives a contradiction. This implies P not equal to NP. The paper "*On Alternation and the Union Theorem*" is available at http://arxiv.org/abs/1602.04781.
    (Thanks to Rolf Niedermeier and Ryan Dougherty for providing these links.)

113. **[Equal]:** In March 2016, Steven Meyer established P=NP. Meyer solves the P-versus-NP problem philosophically by showing P is equal to NP in the random access with unit multiply (MRAM) model. More precisely, the P-versus-NP problem is shown to be a scientific rather than a mathematical problem. The assumptions involved in the current definition of the P-versus-NP problem as a problem involving non deterministic Turing Machines (NDTMs) from axiomatic automata theory are criticized. The problem is also shown to be neither a problem in pure nor applied mathematics.
    The paper "*Philosophical Solution to P=?NP: P is Equal to NP*" is available at http://arxiv.org/abs/1603.06018.
    (Thanks to Samuli Leppänen for providing these links.)

114. **[Not equal]:** In April 2016, Javier A. Arroyo-Figueroa showed that P is not equal to NP. Arroyo-Figueroa establishes the existence of a certain class of one-way functions that are (i) computable in polynomial time and (ii) with negligible probability of finding its inverse by any polynomial probabilistic algorithm. This is accomplished by constructing each member in T with a collection of independent universal hash functions that produce a starting coordinate and a path within a sequence of unique random bit matrices. The existence of one-way functions implies that P is not equal to NP.
    The paper "*The Tau One-Way Functions Class: P != NP*" is available at http://arxiv.org/abs/1604.03758.
    (Thanks to Szabolcs Ivan and Christian Lidström for providing these links.)

115. **[Equal]:** In summer 2016, Eli Halylaurin showed P=NP. The main result is that PSPACE is included in P. Because it is already known that P is included in NP and NP is included in PSPACE, this implies the desired P=NP. The paper "*An Attempt to Demonstrate P=NP*" is available at http://vixra.org/abs/1605.0278.

116. **[Not equal]:** In September 2016, Stefan Rass showed that weak one-way functions exist. These are constructed as preimages of sequences of decision problem instances that are sampled randomly by means of an explicit threshold function. As a consequence, P is not equal to NP. The paper "*On the Existence of Weak One-Way Functions* is available at http://arxiv.org/abs/1609.01575.
    (Thanks to Andras Salamon for providing these links.)

2016年以降はさらに増えて年５件以上

講演の流れ

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独やLINE友だち問題
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

コンピュータの数学

１９３６年に「テューリングマシン」として
計算および計算に必要な時間を定式化

⇨ 「計算」に対して数学的な議論が可能に

解法（アルゴリズム）の性能：計算に必要なステップス数

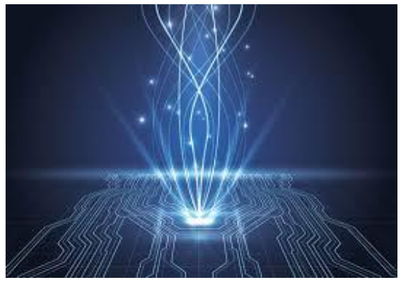
アラン・テューリング
（1912年 – 1954年）

チャーチ＝テューリングの提唱

どんな計算モデルもテューリングマシンで定式化できる
（ステップス数も大きく変わらない）

アロンゾ・チャーチ
（1903年 – 1995年）

現在のスパコンも未来の計算機も含む



足し算

解法（アルゴリズム）の性能：計算に必要なステップス数

右端（一の位）から順に、それぞれの位の数字を足す。足した結果が 10 以上になったら、1 の位の数字だけを書き、十の位は次の桁に「繰り上げる」。

1桁ごとに 3 ステップ

5桁の足し算
15 ステップ

```
  12345
+ 46789
```

10桁の足し算
30 ステップ

```
  123456789
+ 987654321
```

20桁の足し算
60 ステップ

```
  72635273545786043726
+ 53827484732625435473
```

50桁の足し算
150 ステップ

```
  47563739203487456438992305757328576452364568456465744576
+ 98656092843467546234868431987543210979832865874134653472
```

N桁の足し算
3N ステップ

```
  * * * * * * * * *
+ * * * * * * * * *
-----------------
* * * * * * * * * *
```

非常に速い: 入力データを読む（Nステップ）のとほとんど同じくらい速い

掛け算

解法（アルゴリズム）の性能：計算に必要なステップス数

計算量：（およそ）N^2 ステップ

5桁の掛け算
25 ステップ

```
  12345
x 46789
```

10桁の掛け算
100 ステップ

```
  123456789
x 987654321
```

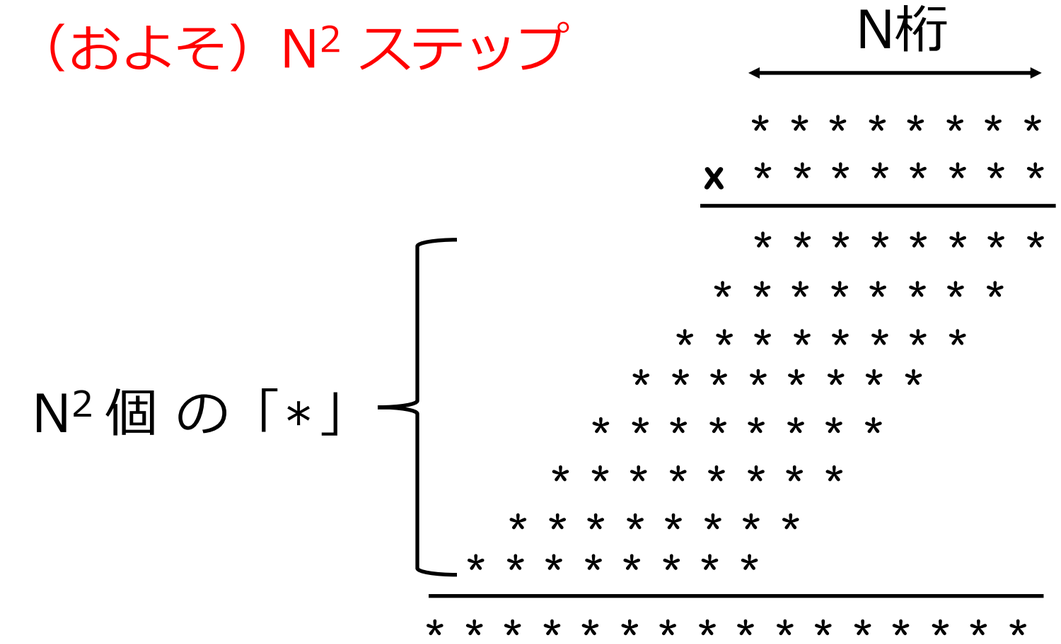
20桁の掛け算
400 ステップ

```
  72635273545786043726
x 53827484732625435473
```

50桁の掛け算
2500 ステップ

```
  47563739203487456438992305757328576452364568456465744576
x 98656092843467546234868431987543210979832865874134653472
```

N桁の掛け算
およそ N^2 ステップ



速い：１万桁でも１秒でできる！

素因数分解

解法（アルゴリズム）の性能：計算に必要なステップス数

15 = 3 x 5

2が割り切れるのか？ 3が割り切れるのか?...

147,573,952,589,676,412,927 = ?193,707,721 x 761,838,257,287

調べる数 B は $\sqrt{A}$ （N/2桁）までで十分

整数 A の素因数分解を求める「試し割り法」

A が N 桁の整数のとき

およそ $10^{N/2}$ ステップ

1. まず $B = 2$ から始めて、B が A を割り切るかどうかを調べる。
2. 割り切れたら、B を因数として記録し、A を割った商で続ける。
3. 次の整数についても同じ操作を繰り返す。

非常に遅い: 1,000桁では１兆年以上かかる

より速いアルゴリズムは存在するのか？

存在するが、それでも非常に遅い！

素因数分解

補足：素数かどうかの判定なら速くできる

“PRIME IS IN P” 2002年


アグラワル


カヤル


サクセナ

ゲーデル賞　2006年
ファルカーソン賞 2006年

A が N 桁の整数のとき

およそ $10^{N/2}$ ステップ

1. まず $B = 2$ から始めて、B が A を割り切るかどうかを調べる。
2. 割り切れたら、B を因数として記録し、A を割った商で続ける。
3. 次の整数についても同じ操作を繰り返す。

非常に遅い: 1,000桁では１兆年以上かかる

より速いアルゴリズムは存在するのか？

存在するが、それでも非常に遅い！

P（Polynomial Time ― 多項式時間）

P: 効率よく解ける問題（の集合）

入力サイズが増えても処理時間が現実的な範囲で増加する問題

入力サイズに対して多項式増加

足し算: 3N ステップ

掛け算: およそ N^2 ステップ

素性判定

最短距離の計算（カーナビゲーションなど）

検索エンジン

医療、生物（DNA鑑定など）

…

NP（Non-deterministic P ― 非決定的 P）

P: 効率よく解ける問題（の集合）

NP: 解を効率よく検証できる問題（の集合）

例：素因数分解

147,573,952,589,676,412,927 =

193,707,721 x 761,838,257,287

数独、クリーク問題、部分和問題など

講演の流れ

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独やLINE友だち問題
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

数独

入力サイズが増えても処理時間がどうなるか調べたい

5	2		8		4		3	9
3								1
			5		7			
2		7	6		8	9		3
				5				
4		3	9		1	8		2
			7		3			
6								5
7	8		4		5		1	6

5	2	6	8	1	4	7	3	9
3	7	4	2	9	6	5	8	1
8	9	1	5	3	7	6	2	4
2	1	7	6	4	8	9	5	3
9	6	8	3	5	2	1	4	7
4	5	3	9	7	1	8	6	2
1	4	5	7	6	3	2	9	8
6	3	2	1	8	9	4	7	5
7	8	9	4	2	5	3	1	6

勝つ条件：

縦・横の各列および各ブロックに各数字（１～９）がちょうど１回ずつ表れる

⇐ 解を簡単に検証できる問題

16 x 16 数独

入力サイズが増えても処理時間がどうなるか調べたい

	13		1	9		15			7		3	2		14	
16			7	10	6		2	11		5	12	8			15
				7							13				
5	11													10	4
1	9	12				13			3				5	6	10
	15				14	11			10	16				12	
13				2	7					8	4				1
	2													4	
	6													5	
12				16	8					9	15				14
	10				4	12			11	6				7	
2	3	15				1			14				6	13	9
6	12													9	5
				1							16				
4			8	15	9		3	1		13	7	11			16
	14		2	5		4			15		10	3		1	

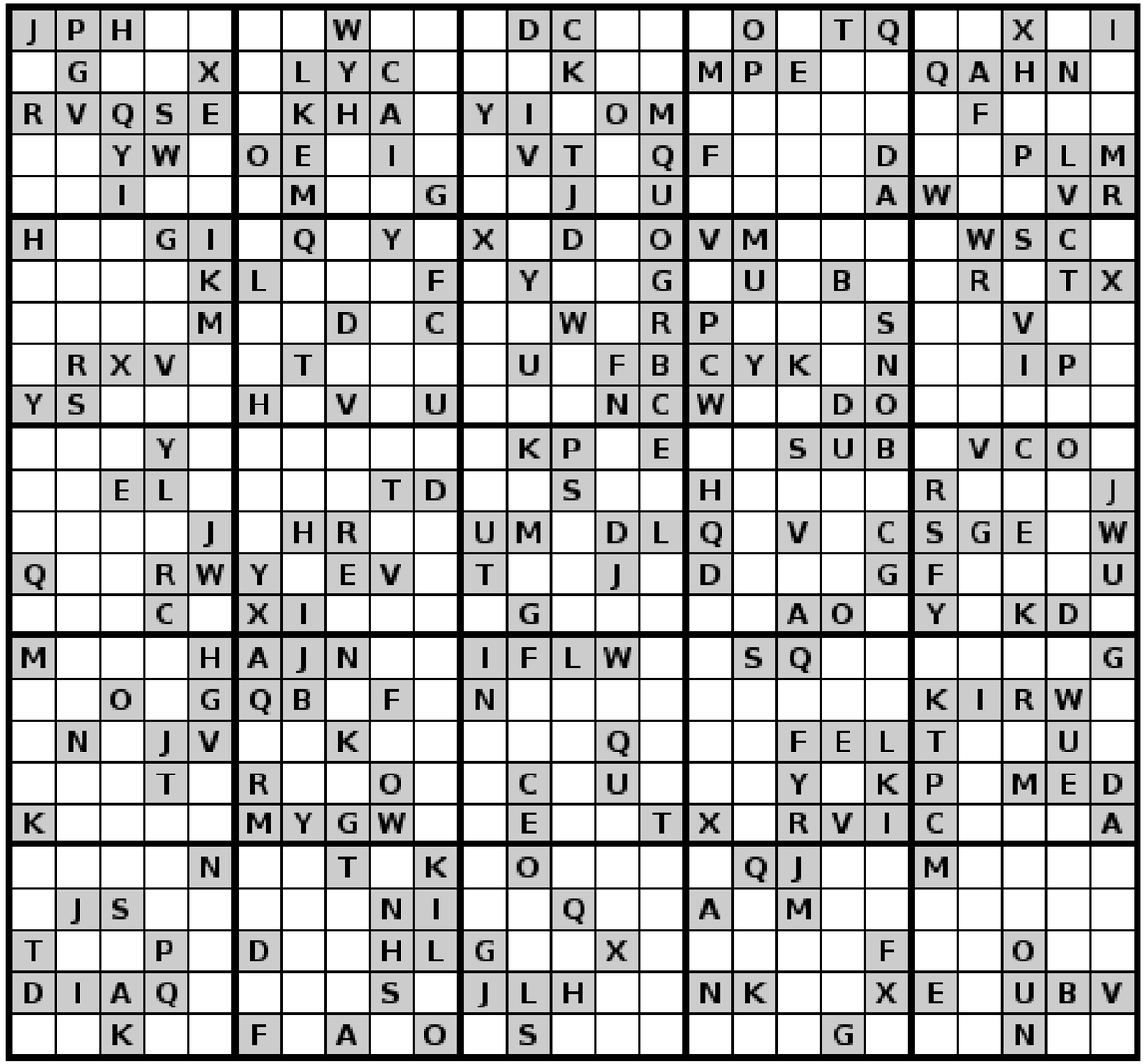
勝つ条件：

縦・横の各列および各ブロックに各数字（1～16）がちょうど１回ずつ表れる

解を簡単に検証できる問題

25 x 25 数独

入力サイズが増えても処理時間がどうなるか調べたい



J	P	H					W				D	C				O		T	Q			X		I
	G			X		L	Y	C				K			M	P	E			Q	A	H	N	
R	V	Q	S	E		K	H	A		Y	I		O	M							F			
		Y	W		O	E		I			V	T		Q	F				D			P	L	M
		I				M			G			J		U					A	W			V	R
H			G	I		Q		Y		X		D		O	V	M					W	S	C	
				K	L				F		Y			G		U		B			R		T	X
				M			D		C			W		R	P				S			V		
	R	X	V			T					U		F	B	C	Y	K		N			I	P	
Y	S				H		V		U				N	C	W			D	O					
			Y								K	P		E			S	U	B		V	C	O	
		E	L					T	D			S			H					R				J
				J		H	R			U	M		D	L	Q		V		C	S	G	E		W
Q			R	W	Y		E	V		T			J		D				G	F				U
			C		X	I					G						A	O		Y		K	D	
M				H	A	J	N			I	F	L	W			S	Q							G
		O		G	Q	B		F		N											K	I	R	W
	N		J	V			K						Q				F	E	L	T			U	
			T		R			O			C		U				Y		K	P		M	E	D
K					M	Y	G	W			E			T	X		R	V	I	C				A
				N			T		K		O					Q	J			M				
	J	S						N	I			Q			A		M							
T			P		D			H	L	G			X						F			O		
D	I	A	Q					S		J	L	H			N	K			X	E		U	B	V
		K			F		A		O		S							G				N		

勝つ条件：

縦・横の各列および各ブロックに各数字（1～25）がちょうど１回ずつ表れる

解を簡単に検証できる問題

N x N 数独（ただし、Nは平方数）

9 x 9, 16 x 16, 25 x 25, 36 x 36, 49 x 49, 64 x 64, 81 x 81, 100 x 100, ...

勝つ条件：

縦・横の各列および各ブロックに各数字（1～N）がちょうど１回ずつ表れる

解を簡単に検証できる問題

N x N 数独はNP問題である

最良のアルゴリズムを使えば

9 x 9: １秒未満

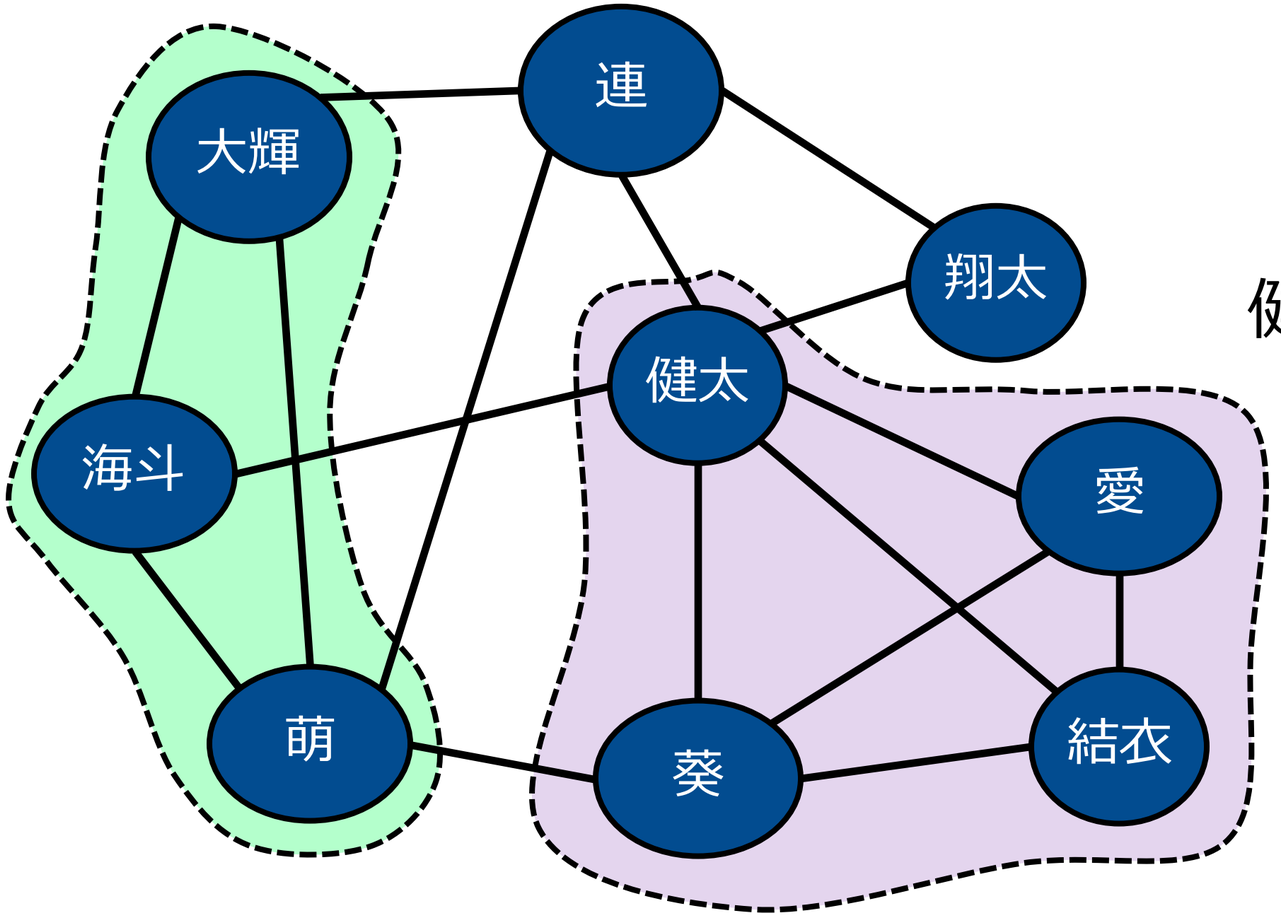
16 x 16: １分未満

25 x 25: １時間未満

...

100 x 100になると解ける見込みが全くない

社会的ネットワーク（例えばLINE友だち）



大輝が海斗と連とLINEで繋がっている
翔太が連と健太とLINEで繋がっている
…

健太、愛、結衣、葵は「グループ」になっている（互いに完全に繋がっている）

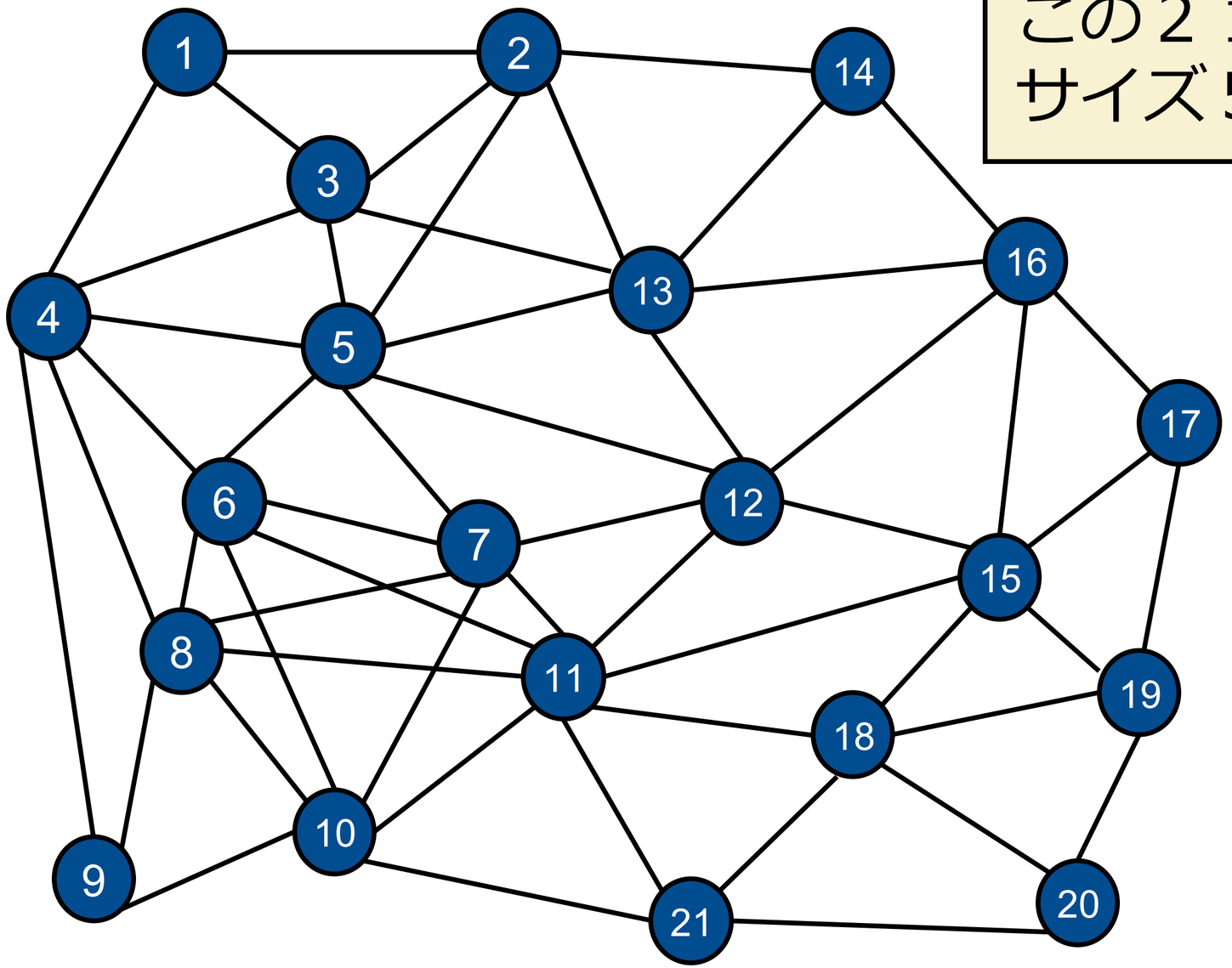
⇨ サイズ４のグループ

連、翔太、健太も「グループ」になっている（互いに完全に繋がっている）

⇨ サイズ 3 のグループ

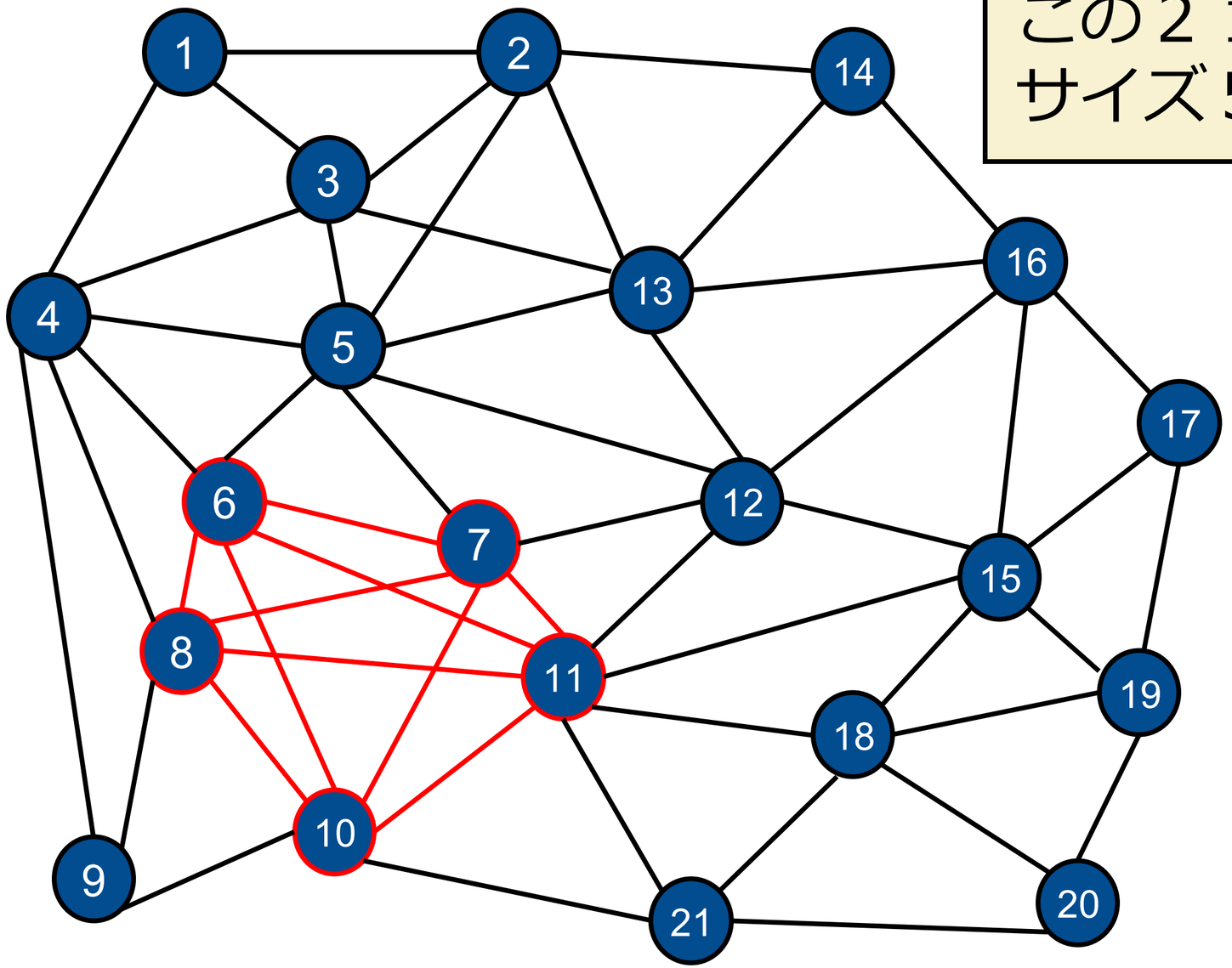
グラフ理論ではこのようなグループをクリークと呼ぶ

クリーク問題



この２１頂点のフラフにおいて、
サイズ５のクリークが存在するのか？

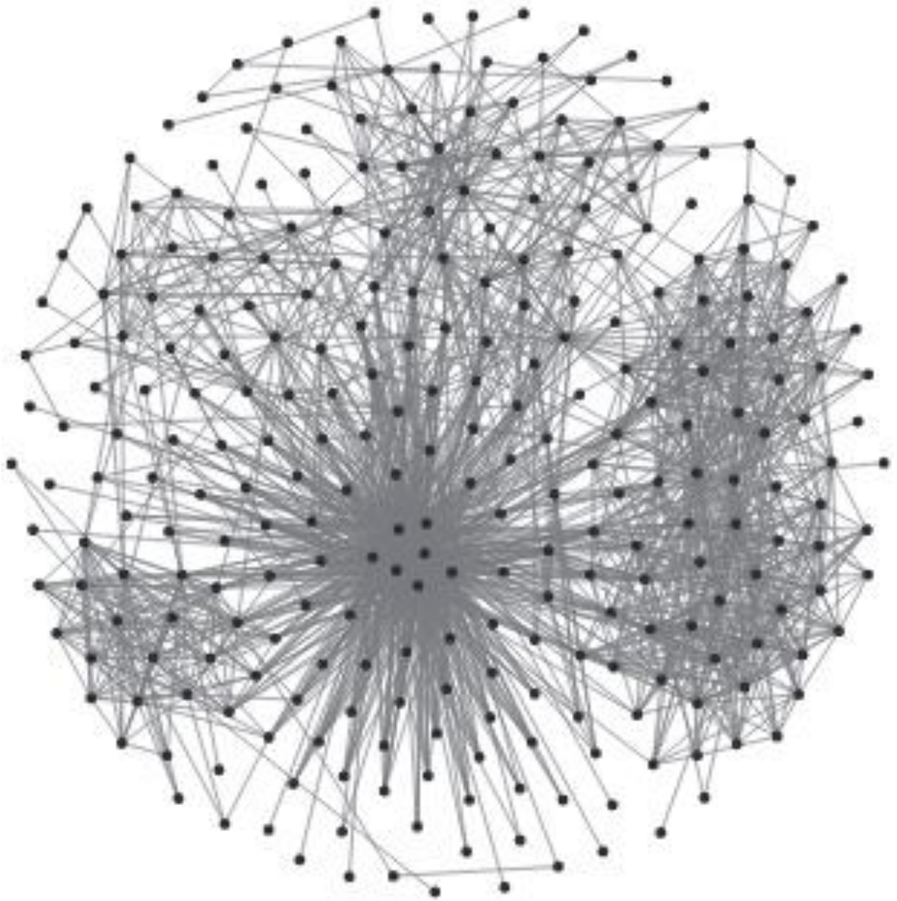
クリーク問題



この２１頂点のフラフにおいて、
サイズ５のクリークが存在するのか？

解を簡単に検証できる

クリーク問題：一般のグラフ



N 頂点のフラフにおいて、
サイズ M のクリークが存在するのか？

↑

解を簡単に検証できる

クリーク問題はNP問題である

M 頂点の選び方は ${}_{N}C_{M}$ 通り

N=1000, M=50 のとき、10^{85} 通り

(参考：宇宙に存在する原子の数が 10^{80} 程度)

部分和問題：例

以下の38個の数から、19個を選んで足し合わせたとき、
その和がちょうど 1,000,000 になるのか？

14,175	19,300	26,343	41,867	58,306	69,189	82,027	97,042
15,055	19,731	28,725	43,155	61,848	72,936	82,623	97,507
16,616	22,161	29,127	46,298	65,825	74,287	82,802	99,564
17,495	23,320	32,257	56,734	66,042	74,537	82,988	
18,072	23,717	40,020	57,176	68,634	81,942	90,467	

赤い数の和 = 1,000,000

解を簡単に検証できる

部分和問題：一般

与えらている N 個の数（N:偶数）から、N/2を選んで足し合わせたとき、
その和がちょうど 目標値 Tになるのか？

Tも入力として与えられる

解を簡単に検証できる

部分和問題はNP問題である

選び方は ${}_{N}C_{N/2}$ 通り

N = 300のとき、10^{89}通り以上！

講演の流れ

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独やLINE友だち問題
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

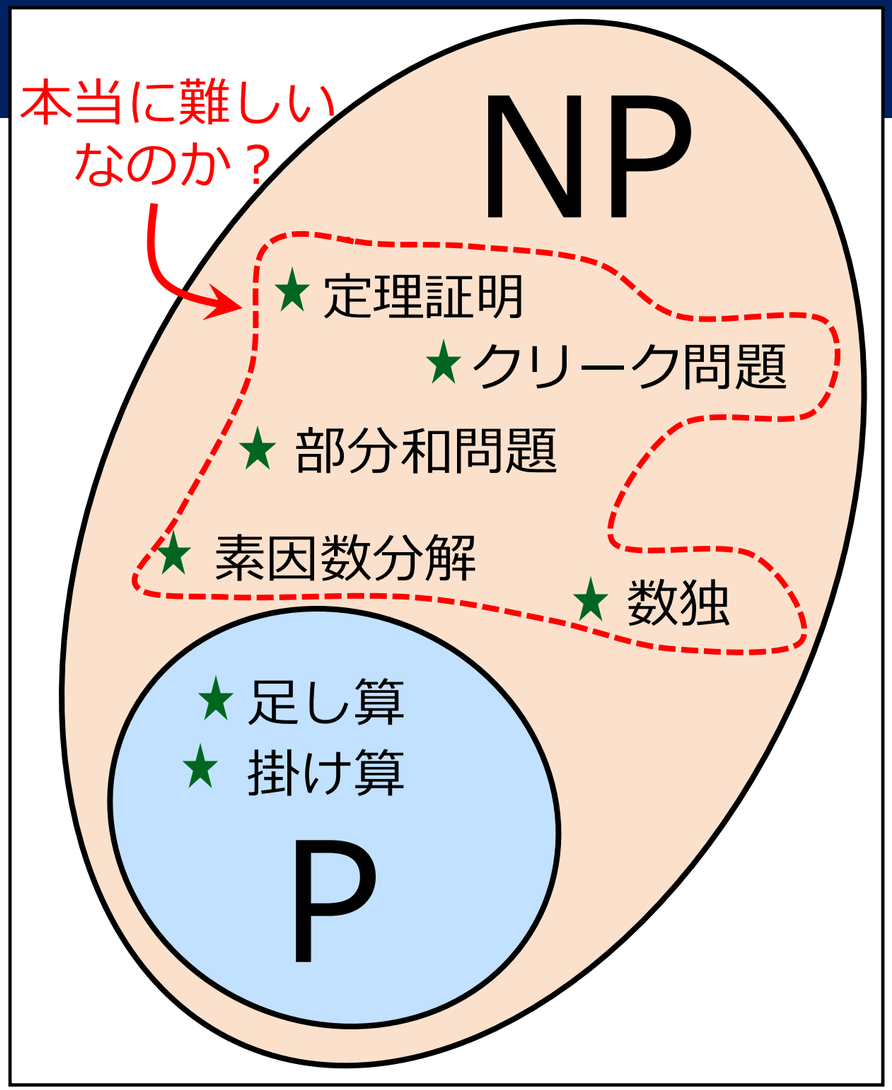
P と NP

P: 効率よく解ける問題（の集合）

例：足し算、掛け算

NP: 解を効率よく検証できる問題（の集合）

例：素因数分解、 数独、クリーク問題、
部分和問題、定理証明



P ≠ NP のときのイメージ図

例：数独の難しさの議論

- ✓ 「数独は効率よく解けない」と証明できれば
  P ≠ NP の証明になる　　絶対的な困難性
- ✓ 「数独は NP の問題の中で最も難しい
  問題である」なら証明できる　　相対的な困難性

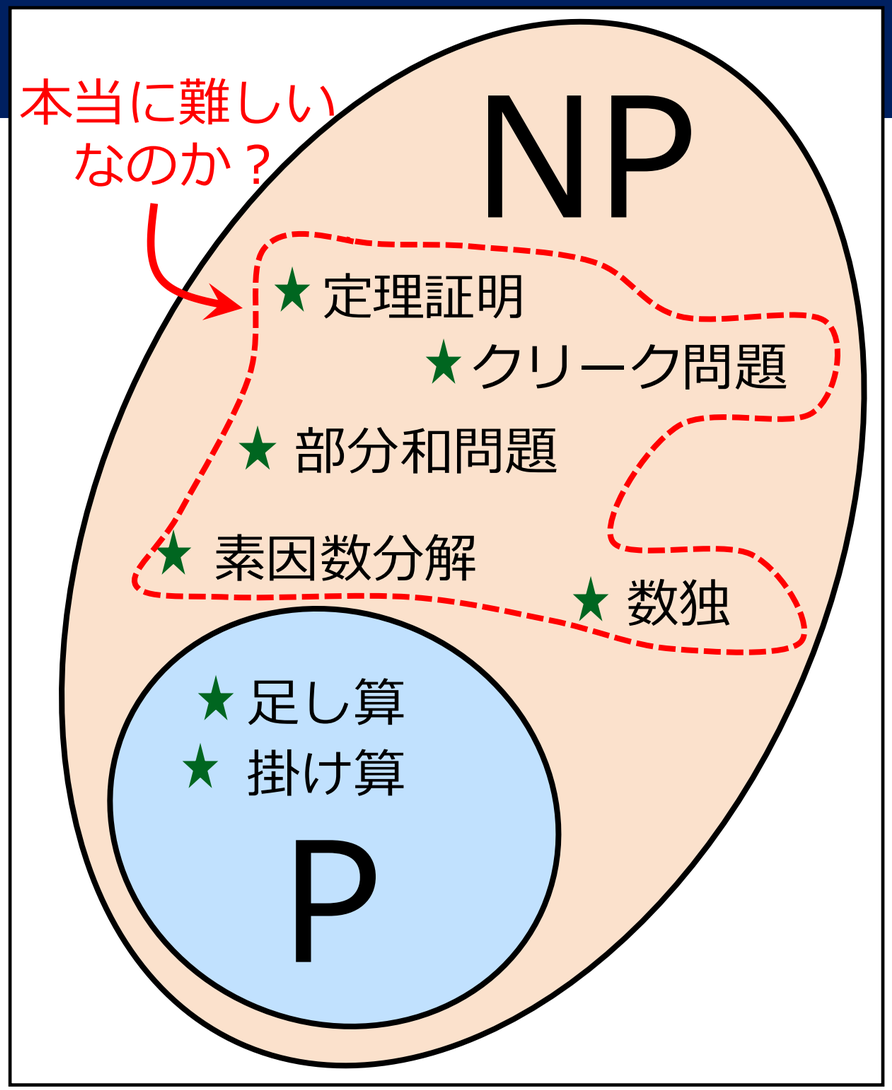
正確な主張（NP完全性）

> 数独が効率よく解けるならば、
> NP のすべての問題も効率よく解ける

数独が簡単ならば：

- クリーク問題も効率よく解ける
- 部分和問題も効率よく解ける
- 定理証明問題も効率よく解ける
- 素因数分解も効率よく解ける

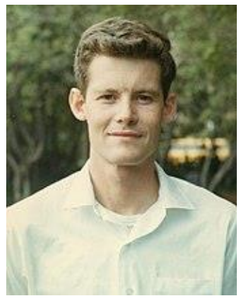
何万人の研究者が何十年も挑戦してきた



P ≠ NP のときのイメージ図

NP完全性の歴史

スティーブン・クックとレオニード・レヴィンは、
1971年に独立に、 P ≠ NP 問題を定式化した



スティーブン・
クック
1971

Appears in
Proceedings Third Annual
ACM Symposium on
Theory of Computing
May, 1971

The Complexity of Theorem-Proving Procedures

Stephen A. Cook
University of Toronto

Summary

It is shown that any recognition problem solved by a polynomial time-bounded nondeterministic Turing machine can be "reduced" to the problem of determining whether a given propositional formula is a tautology. Here "reduced" means, roughly speaking, that the first problem can be solved deterministically in polynomial time provided an oracle is available for solving the second. From this notion of reducible, polynomial degrees of difficulty are defined, and it is shown that the problem of determining tautologyhood has the same polynomial degree as the problem of determining whether the first of two given graphs is isomorphic to a subgraph of the second. Other examples are discussed. A method of measuring the complexity of proof procedures for the predicate calculus is introduced and discussed.

Throughout this paper, a set of strings means a set of strings on some fixed, large, finite alphabet Σ. This alphabet is large enough to include symbols for all sets described here. All Turing machines are deterministic recognition devices, unless the contrary is explicitly stated.

1. Tautologies and Polynomial Reducibility.

Let us fix a formalism for the propositional calculus in which formulas are written as strings on Σ. Since we will require infinitely many proposition symbols (atoms), each such symbol will consist of a member of Σ followed by a number in binary notation to distinguish that symbol. Thus a formula of length n can only have about n/logn distinct function and predicate symbols. The logical connectives are & (and), ∨ (or), and ¬(not).

The set of tautologies (denoted by {tautologies}) is a certain recursive set of strings on this alphabet, and we are interested in the problem of finding a good lower bound on its possible recognition times. We provide no such lower bound here, but theorem 1 will give evidence that {tautologies} is a difficult set to recognize, since many apparently difficult problems can be reduced to determining tautologyhood. By reduced we mean, roughly speaking, that if tautologyhood could be decided instantly (by an "oracle") then these problems could be decided in polynomial time. In order to make this notion precise, we introduce query machines, which are like Turing machines with oracles in [1].

A query machine is a multitape Turing machine with a distinguished tape called the query tape, and three distinguished states called the query state, yes state, and no state, respectively. If M is a query machine and T is a set of strings, then a T-computation of M is a computation of M in which initially M is in the initial state and has an input string w on its input tape, and each time M assumes the query state there is a string u on the
the next state M
yes state if u∈T
if u∉T. We thin
which knows T, pl
yes state or no s

Definition

A set S of strings is P-reducible (P for polynomial) to a set T of strings if
query machine M
Q(n) such that
w, the T-computa
put w halts wi
(|w| is the leng
in an accepting

It is not h
P-reducibility i
lation. Thus th

NP完全性の概念も導入

次の定理（クック・レヴィンの定理）も証明：

NP完全問題は存在する

ПРОБЛЕМЫ ПЕРЕДАЧИ ИНФОРМАЦИИ
Том IX 1973 Вып. 3

КРАТКИЕ СООБЩЕНИЯ

УДК 519.14

УНИВЕРСАЛЬНЫЕ ЗАДАЧИ ПЕРЕБОРА

Л. А. Левин

В статье рассматривается несколько известных массовых задач «переборного типа» и доказывается, что эти задачи можно решать лишь за такое время, за которое можно решать вообще любые задачи указанного типа.

После уточнения понятия алгоритма была доказана алгоритмическая неразрешимость ряда классических массовых проблем (например, проблем тождества элементов групп, гомеоморфности многообразий, разрешимости диофантовых уравнений и других). Тем самым был снят вопрос о нахождении практического способа их решения. Однако существование алгоритмов для решения других задач не снимает для них аналогичного вопроса из-за фантастически большого объема работы, предписываемого этими алгоритмами. Такова ситуация с так называемыми переборными задачами: минимизации булевых функций, поиска доказательств ограниченной длины, выяснения изоморфности графов и другими. Все эти задачи решаются тривиальными алгоритмами, состоящими в переборе всех возможностей. Однако эти алгоритмы требуют экспоненциального времени работы и у математиков сложилось убеждение, что более простые алгоритмы для них невозможны. Был получен ряд серьезных аргументов в пользу его справедливости (см.[1, 2]), однако доказать это утверждение не удалось никому. (Например, до сих пор не доказано, что для нахождения математических доказательств нужно больше времени, чем для их проверки.)

Однако если предположить, что вообще существует какая-нибудь (хотя бы искусственно построенная) массовая задача переборного типа, неразрешимая простыми (в смысле объема вычислений) алгоритмами, то можно показать, что этим же свойством обладают и многие «классические» переборные задачи (в том числе задача минимизации, задача поиска доказательств и др.). В этом и состоят основные результаты статьи.

Функции $f(n)$ и $g(n)$ будем называть сравнимыми, если при некотором k
$g(n) \leqslant (f(n)+2)^k$.
ин «меньше или сравнимо».
переборного типа (или просто переборной задачей)
данному x найти какое-нибудь y длины, сравнимой
ется $A(x, y)$», где $A(x, y)$ – какое-нибудь свойство,
работы которого сравнимо с длиной x. (Под алгоритмом здесь можно понимать, например, алгоритмы Колмогорова — Успенского или машины Тьюринга, или нормальные алгоритмы; x, y – двоичные слова). Квазипереборной задачей будем называть задачу выяснения, существует ли такое y.

Мы рассмотрим шесть задач этих типов. Рассматриваемые в них объекты кодируются естественным образом в виде двоичных слов. При этом выбор естественной

レオニード・
レヴィン
1973

NP完全性の歴史

リチャード・
カープ
1972

REDUCIBILITY AMONG COMBINATORIAL PROBLEMS†

Richard M. Karp

University of California at Berkeley

Abstract: A large class of computational problems involve the determination of properties of graphs, digraphs, integers, arrays of integers, finite families of finite sets, boolean formulas and elements of other countable domains. Through simple encodings from such domains into the set of words over a finite alphabet these problems can be converted into language recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily solved when an algorithm for its solution is found which terminates within a number of steps bounded by a polynomial in the length of the input. We show that a large number of classic unsolved problems of covering, matching, packing, routing, assignment and sequencing are equivalent, in the sense that either each of them possesses a polynomial-bounded algorithm or none of them does.

1. INTRODUCTION

All the general methods presently known for computing the chromatic number of a graph, deciding whether a graph has a Hamilton circuit, or solving a system of linear inequalities in which the variables are constrained to be 0 or 1, require a combinatorial search for which the worst case time requirement grows exponentially with the length of the input. In this paper we give theorems which strongly suggest, but do not imply, that these problems, as well as many others, will remain intractable perpetually.

†This research was partially supported by National Science Foundation Grant GJ-474.

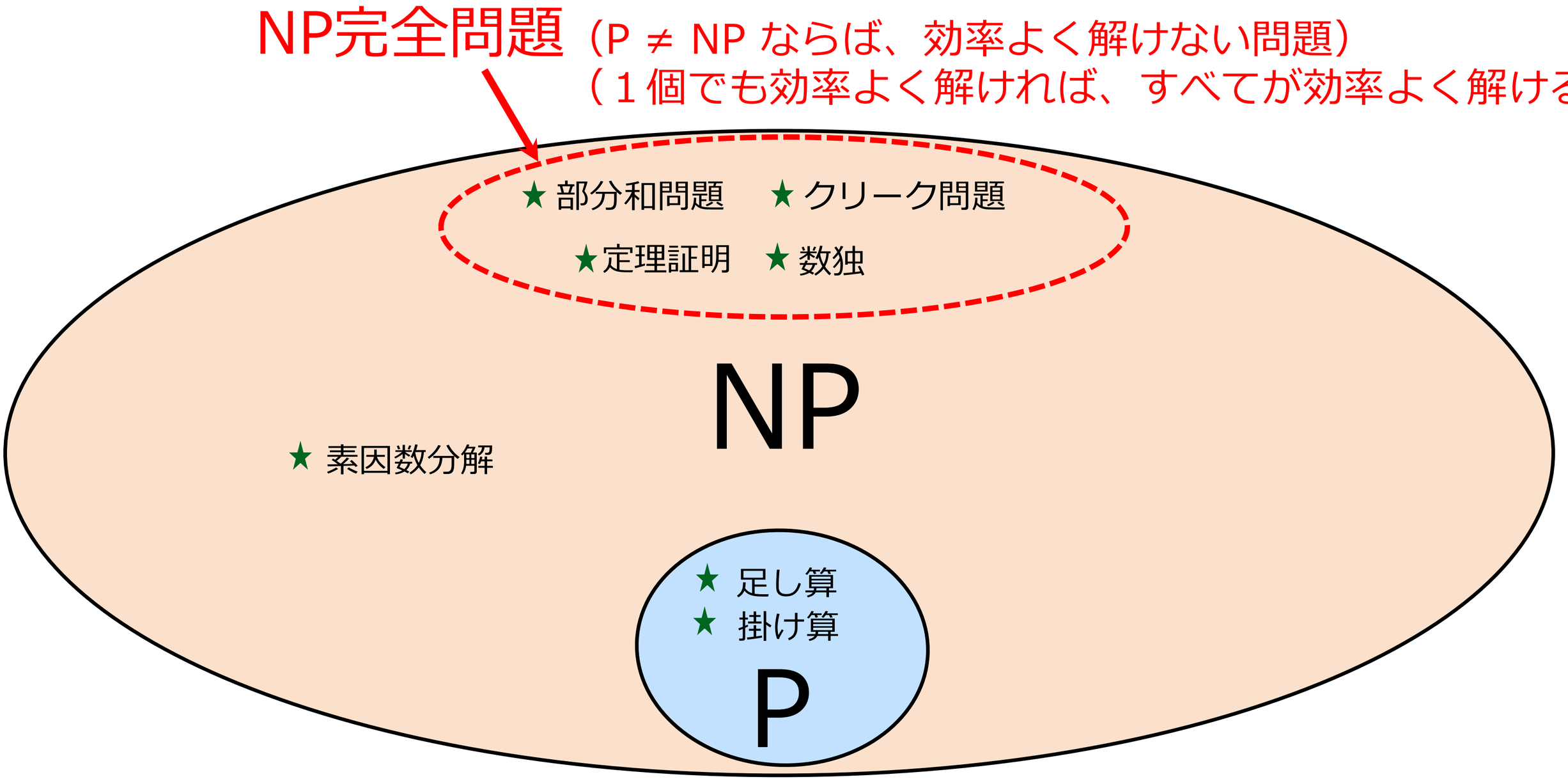
21個の重要な問題のNP完全性を証明

- ✓ 整数計画法
- ✓ グラフの彩色数の計算
- ✓ クリーク問題
- ✓ グラフの頂点被覆の計算
- ✓ ハミルトン閉路問題
- ✓ ナップサック問題
- ✓ 巡回セールスマン問題
- ✓ 部分和問題

…

数独のNP完全性は2003年に証明
（八登崇之さんの修士論文）

P ≠ NP のときのイメージ図



講演の流れ

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独やLINE友だち問題
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

巡回セールスマン問題

アメリカの 49 都市（各州から１都市＋ワシントンDC）を考える。
セールスマンは「すべての都市を1回ずつ訪れて、元の場所に戻る」必要がある。
最短ルートを探したい。

Reprinted from JOURNAL OF THE OPERATIONS RESEARCH SOCIETY OF AMERICA
Vol. 2, No. 4, November, 1954
Printed in U.S.A.

SOLUTION OF A LARGE-SCALE TRAVELING-SALESMAN PROBLEM*

G. DANTZIG, R. FULKERSON, AND S. JOHNSON
The Rand Corporation, Santa Monica, California

(Received August 9, 1954)

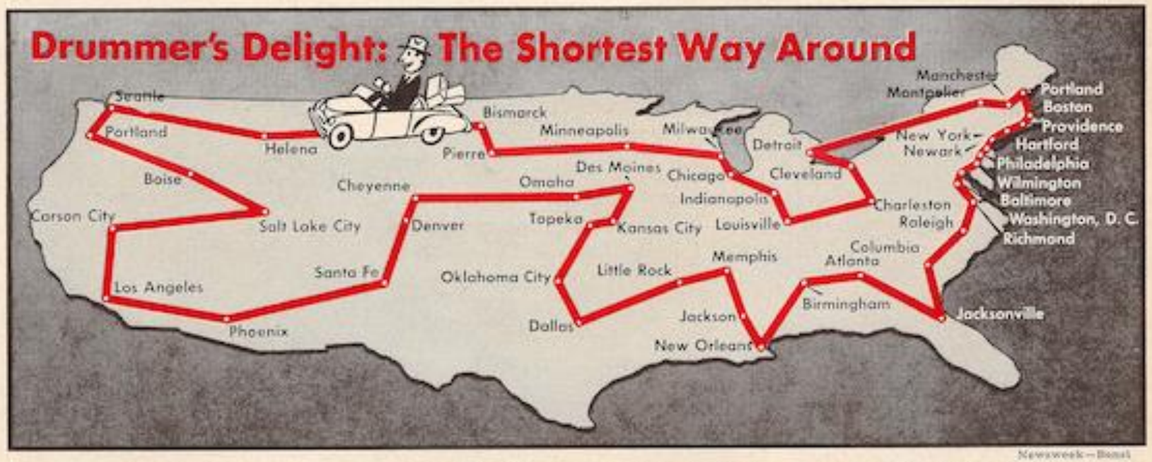
It is shown that a certain tour of 49 cities, one in each of the 48 states and Washington, D. C., has the shortest road distance.

1954年 ランド研究所の論文

$49!/(49 \times 2) \approx 10^{60}$ 通り

SCIENCE

Drummer's Delight: The Shortest Way Around



Newsweek—Bonsi

FINDING the shortest route for a traveling salesman—starting from a given city, visiting each of a series of other cities, and then returning to his original point of departure—is more than an after-dinner teaser. For years it has baffled not only goods- and salesmen-routing businessmen but mathematicians as well. If a drummer visits 50 cities, for example, he has 10^{62} (62 zeros) possible itineraries. No electronic computer in existence could sort out such a large number of routes and find the shortest.

Three Rand Corp. mathematicians, using Rand McNally road-map distances between the District of Columbia and major cities in each of the 48 states, have finally produced a solution (see above). By an ingenious application of linear programming—a mathematical tool recently used to solve production-scheduling problems—it took only a few weeks for the California experts to calculate "by hand" the shortest route to cover the 49 cities: 12,345 miles

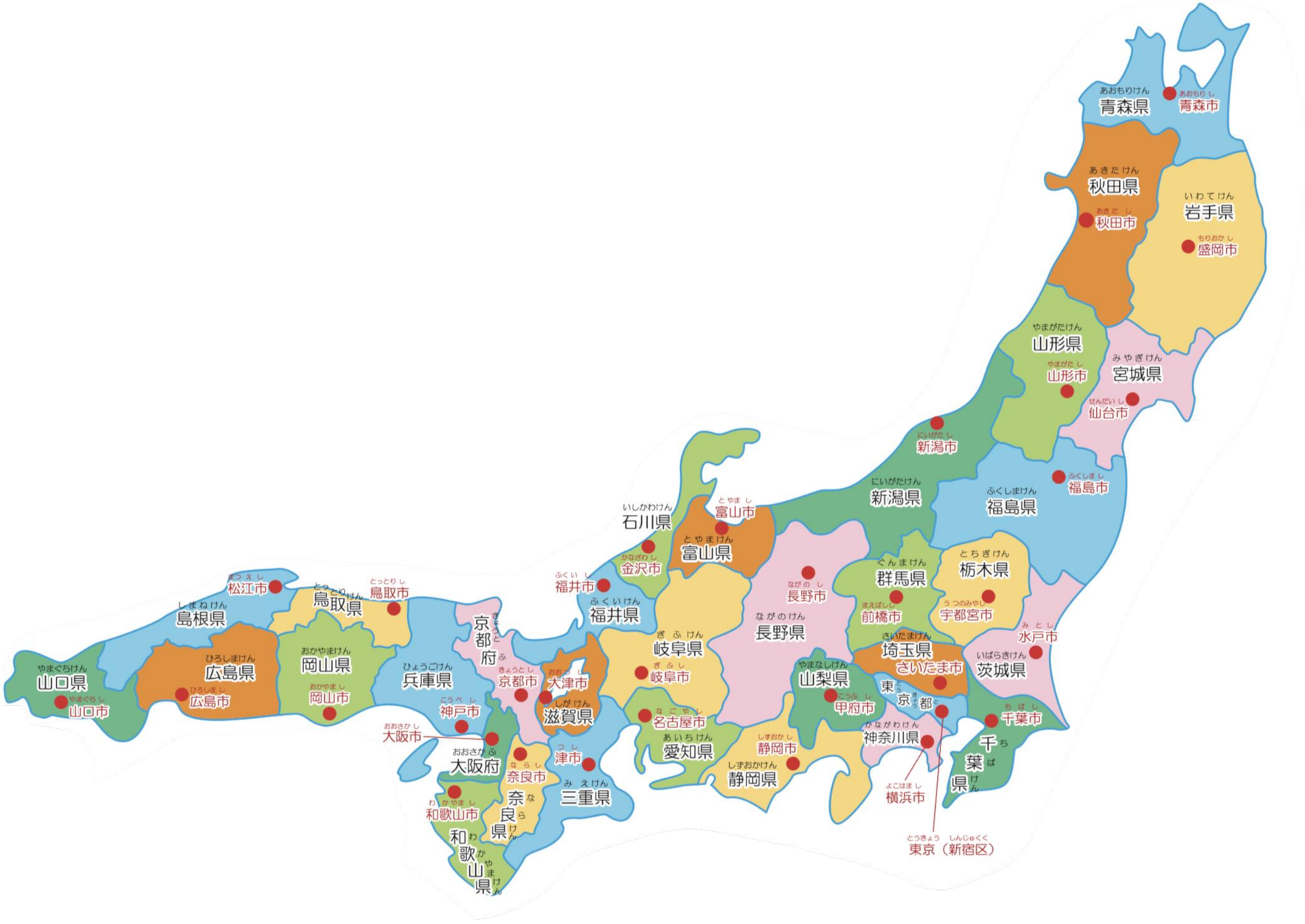
Newsweek, 1954年7月26日

計算機を使って数週間以内に求めた！

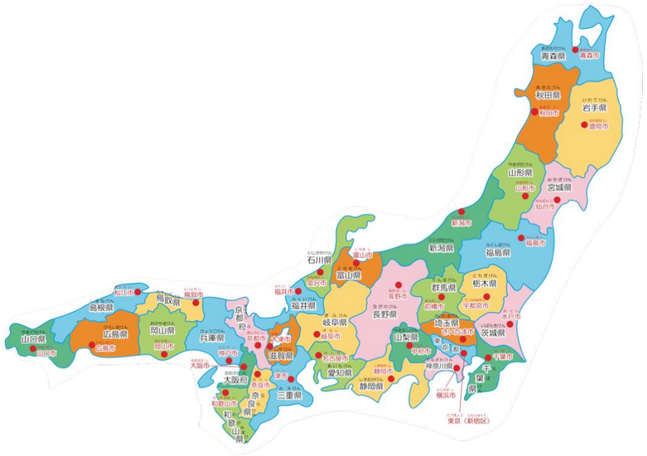
12,345 マイル（＝ 19,867 km）

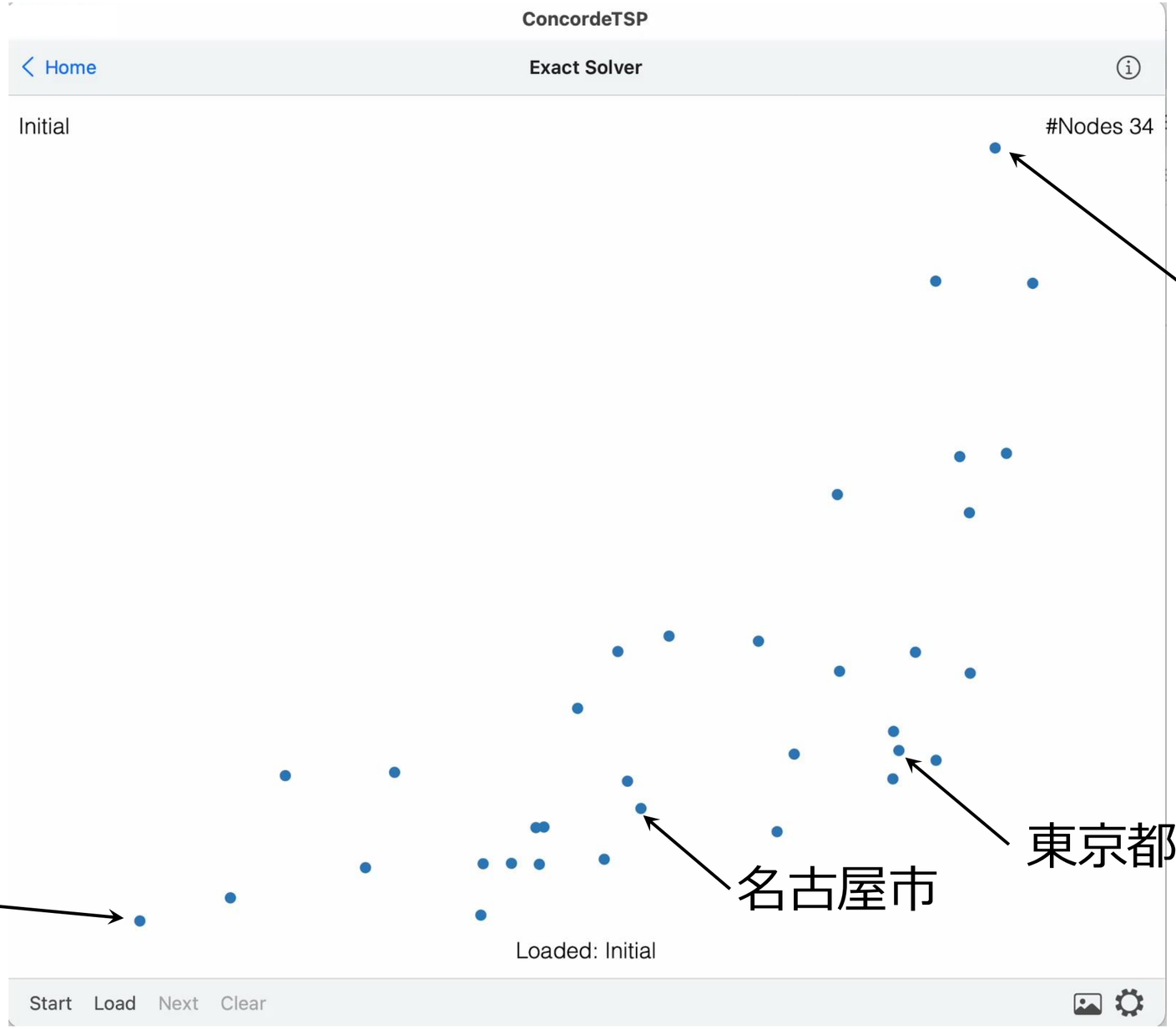
日本（本州）県庁所在地

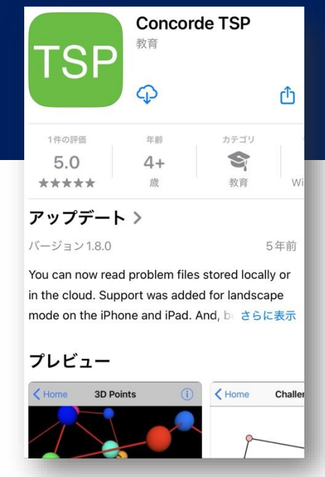
34 都市



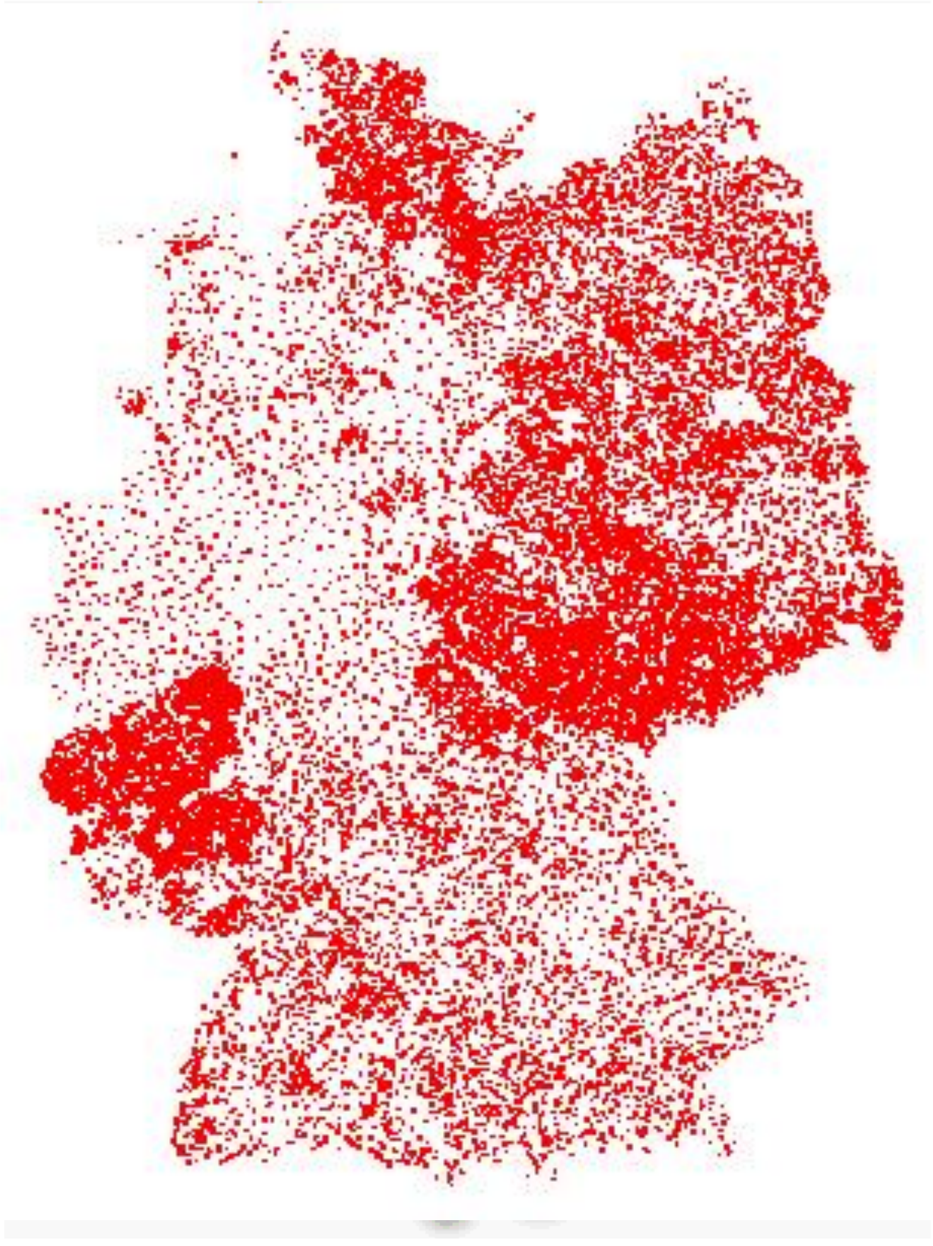
日本（本州）県庁所在地




ConcordeTSP
Home
Exact Solver
Initial
#Nodes 34
青森市
東京都（新宿区）
名古屋市
山口市
Loaded: Initial
Start
Load
Next
Clear


Concorde TSP
教育
TSP
1件の評価
5.0
年齢
4+
歳
カテゴリ
教育
アップデート
バージョン1.8.0
5年前
You can now read problem files stored locally or in the cloud. Support was added for landscape mode on the iPhone and iPad. And,
さらに表示
プレビュー
Home
3D Points
Home

ドイツの 15,112 都市問題



最短ルート：66,177 km

2001年に100台以上のスパコンを繋げて
計算された

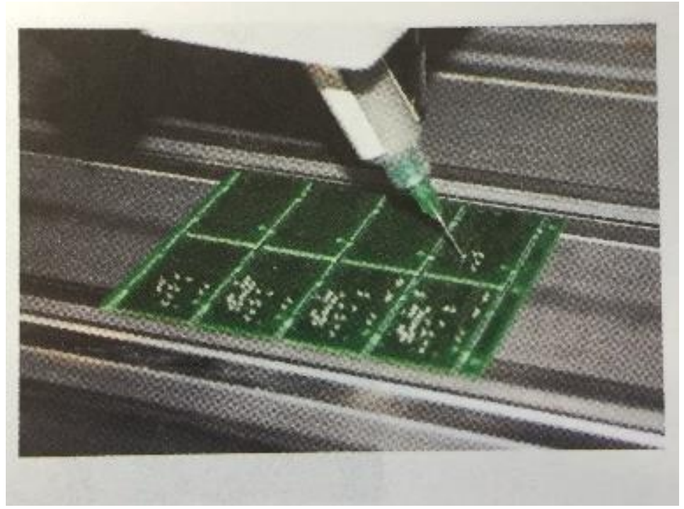
巡回セールスマン問題の応用の例

物流・配送計画

宅急便　大手なら１日500万個以上の小包

製造業（マイクロチップの製造など）

ヘッドは数十万か所の点に金属接合部を
配置しなければならない



時は金なり

巡回セールスマン問題のNP完全

リチャード・
カープ
1972

REDUCIBILITY AMONG COMBINATORIAL PROBLEMS†

Richard M. Karp

University of California at Berkeley

Abstract: A large class of computational problems involve the determination of properties of graphs, digraphs, integers, arrays of integers, finite families of finite sets, boolean formulas and elements of other countable domains. Through simple encodings from such domains into the set of words over a finite alphabet these problems can be converted into language recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily solved when an algorithm for its solution is found which terminates within a number of steps bounded by a polynomial in the length of the input. We show that a large number of classic unsolved problems of covering, matching, packing, routing, assignment and sequencing are equivalent, in the sense that either each of them possesses a polynomial-bounded algorithm or none of them does.

1. INTRODUCTION

All the general methods presently known for computing the chromatic number of a graph, deciding whether a graph has a Hamilton circuit, or solving a system of linear inequalities in which the variables are constrained to be 0 or 1, require a combinatorial search for which the worst case time requirement grows exponentially with the length of the input. In this paper we give theorems which strongly suggest, but do not imply, that these problems, as well as many others, will remain intractable perpetually.

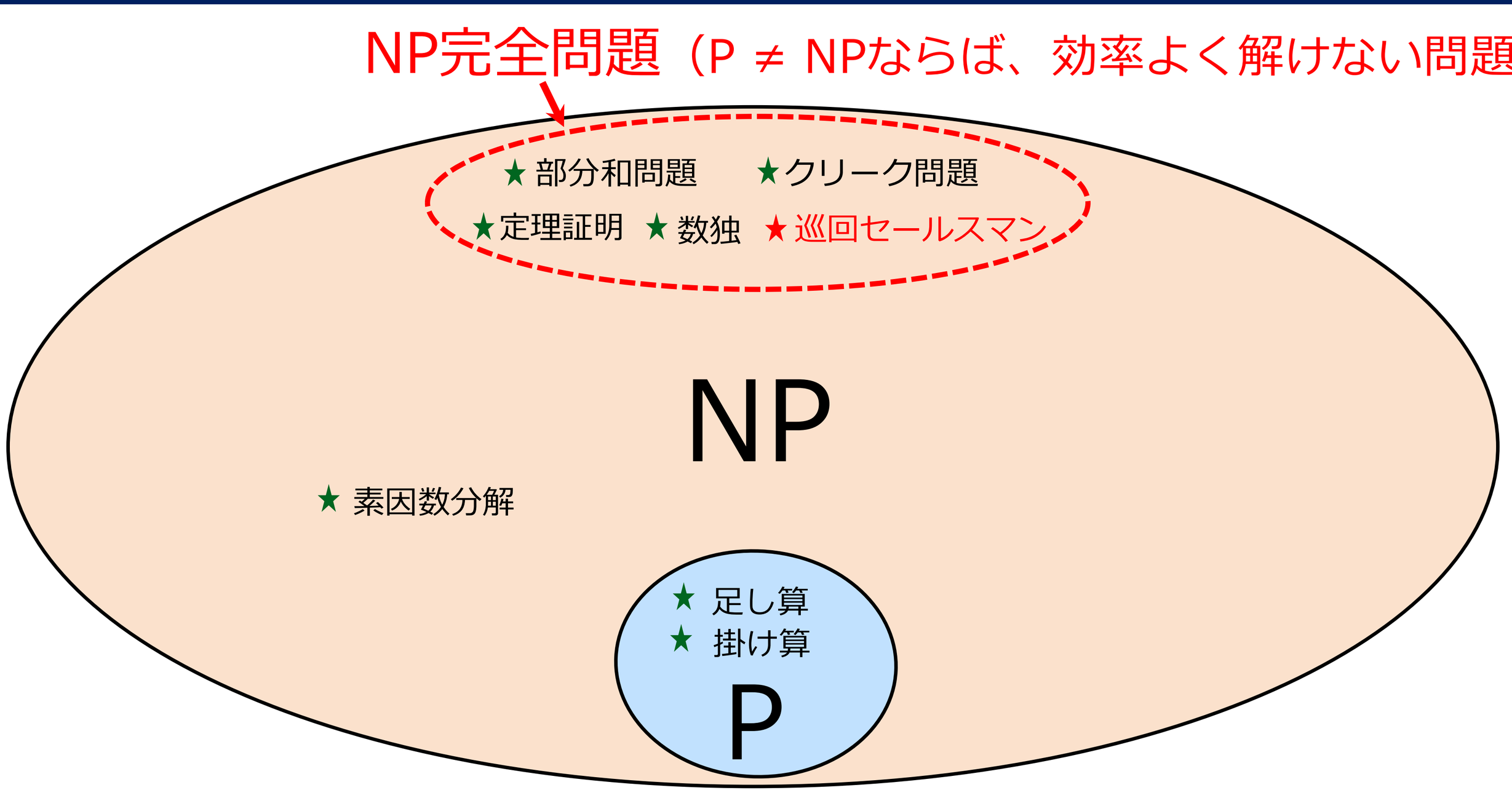
†This research was partially supported by National Science Foundation Grant GJ-474.

21個の重要な問題のNP完全性を証明

- ✓ 整数計画法
- ✓ グラフの彩色数の計算
- ✓ クリーク問題
- ✓ グラフの頂点被覆の計算
- ✓ ハミルトン閉路問題
- ✓ ナップサック問題
- ✓ 巡回セールスマン問題
- ✓ 部分集合和問題

…

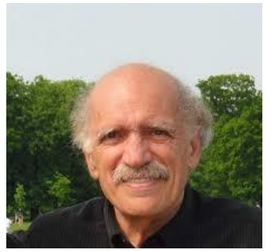
巡回セールスマン問題のNP完全



1990年代：巡回セールスマン問題の近似困難性

クリストフィデスのアルゴリズム (1976年)

最短ルートの 1.5 倍以内の長さのルートは効率よく計算できる



クリストフィデス

最短ルート：66,177 km

66,177 x 1.5 = 99,265 km 以内は簡単

66,177 x 1.007 = 66,640km 以内は難しい

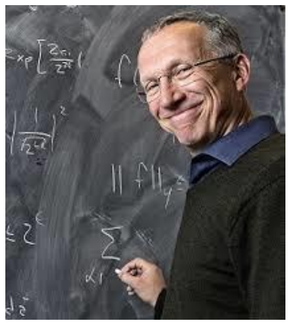
1.4倍以内は可能？

~~1.001は？~~

近似困難性定理 (2000年)

最短ルートの 1.007 倍以内の長さのルートの計算はNP完全である

数十人の研究者が10年間積み重ねてきた研究の成果



ホースタッド

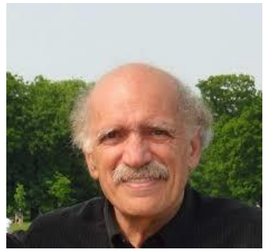
ヴェンパラ

パパディミトリウ

巡回セールスマン問題に対する近年のブレークスルー

クリストフィデスのアルゴリズム (1976年)

最短ルートの 1.5 倍以内の長さのルートは効率よく計算できる



クリストフィデス

カーリン・クライン・ガランのアルゴリズム (2021年)

最短ルートの 1.499999999999999999999999999999999999 倍以内の長さのルートは効率よく計算できる

9 が 35個

A (Slightly) Improved Approximation Algorithm for Metric TSP

Anna R. Karlin*, Nathan Klein†, and Shayan Oveis Gharan‡

For some $\epsilon > 10^{-36}$ we give a randomized $3/2 - \epsilon$ approximation algorithm for metric TSP.

カーリン

クライン

ガラン

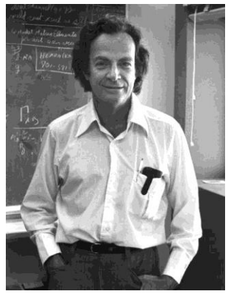
講演の流れ

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独やLINE友だち問題
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

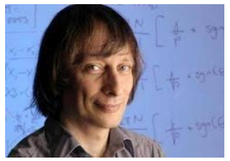
2000年代以降の計算量理論：量子計算

量子計算の提唱

Manin



Feynman



Deutsch

量子系の操作の実験

Wineland Haroche
ノーベル物理学賞（2012）



Aspect Clauser Zeilinger
ノーベル物理学賞（2022）

量子アルゴリズムの発見

Shor

Grover

1980 1982 1985 1994 1996

素因数分解

量子探索

素因数分解とショアのアルゴリズム

147,573,952,589,676,412,927 = ☐ x ☐

従来のコンピュータでは、
膨大な時間がかかる（数千桁の整数なら１兆年も）

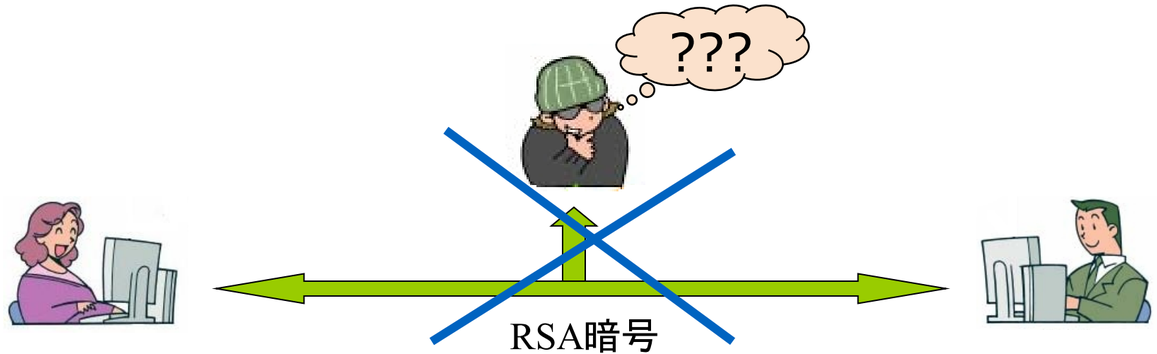
➡ 素因数分解は暗号方式などに使われている

素因数分解を効率よく求める量子アルゴリズムが存在する！

ピーター・ショアによって1994年に発見

ピーター・ショア
ネヴァンリンナ賞　1998年

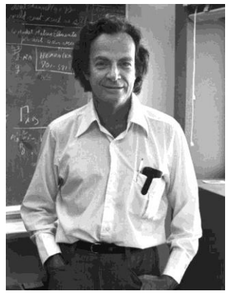
➡ もし量子コンピュータを開発できれば、
現在の暗号方式（RSAなど）を破ることができる！



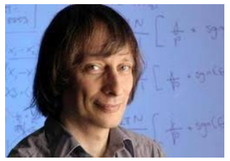
2010年代以降の計算量理論：量子計算

量子計算の提唱

Manin



Feynman



Deutsch

量子系の操作の実験

Wineland Haroche
ノーベル物理学賞（2012）



Aspect Clauser Zeilinger
ノーベル物理学賞（2022）

量子アルゴリズムの発見

Shor

Grover

第1次量子計算ブーム

素因数分解

量子探索

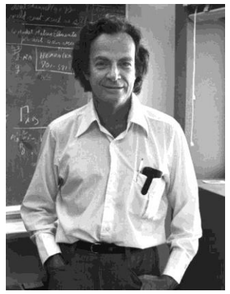
1980 1982 1985 1994 1996

暗黒時代

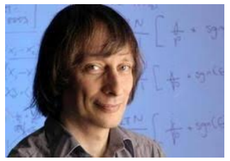
2010年代以降の計算量理論：量子計算

量子計算の提唱

Manin



Feynman



Deutsch

1980　1982　1985

量子系の操作の実験

Wineland　Haroche
ノーベル物理学賞（2012）

Aspect　Clauser Zeilinger
ノーベル物理学賞（2022）

量子アルゴリズムの発見

Shor

Grover

素因数分解

1994　1996

量子探索

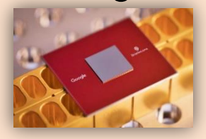
第1次量子計算ブーム

小規模の量子コンピュータの開発に成功
(~数百量子ビット)

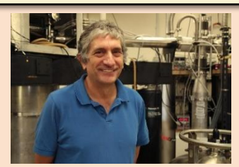
20量子ビット程度までなら無償で（クラウドで）利用できる
https://www.ibm.com/quantum-computing/tools/

理研「叡」

Google



IBM

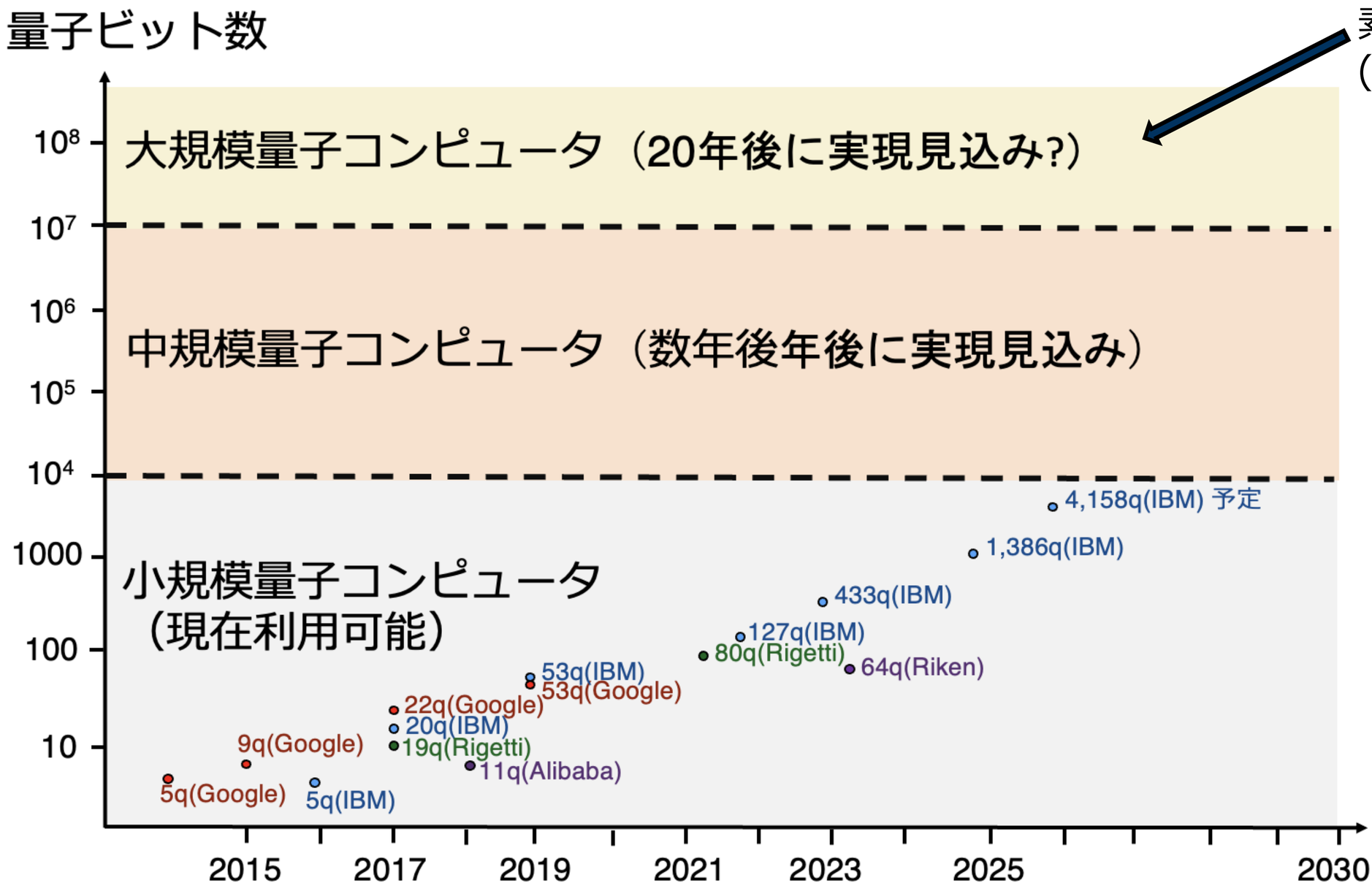


2023　2020　2018　2015　2010

暗黒時代

第2次量子計算ブーム

量子コンピュータ開発の現状と展望



理研 「叡」
（2023年、64量子ビット）

量子計算の喫緊の課題

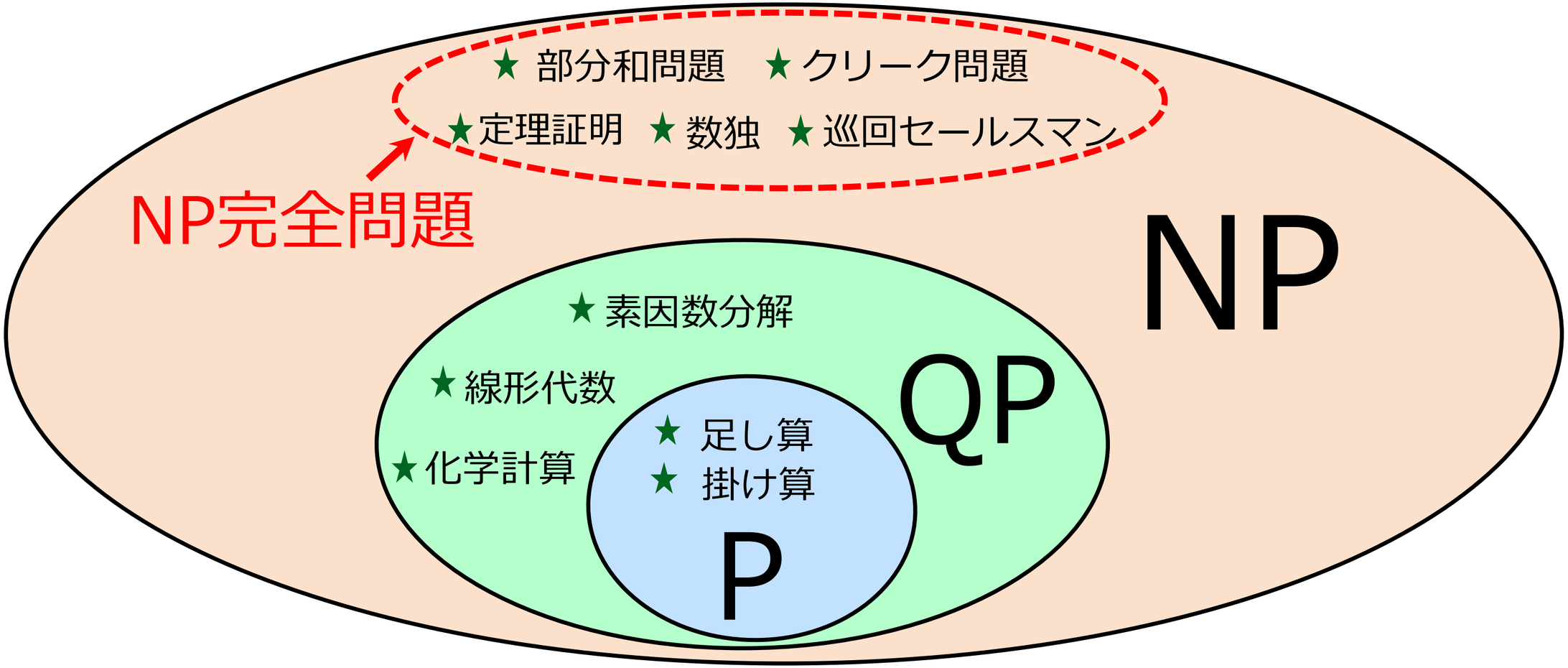
量子コンピュータの計算能力の究明

大規模量子コンピュータがまだ実現できていないので、
数学を使って理論的に研究

量子コンピュータの計算能力

QP: 量子コンピュータで効率よく解ける問題（の集合）

予想　量子コンピュータでNP完全問題は速く解けない



終わりに

1. P ≠ NP 予想の歴史
2. コンピュータの数学
3. 様々な問題：数独など
4. 1970年代の数学：計算の複雑さ
5. 1990年代の数学：近似の難しさ
6. 2000年代以降の数学：量子計算

「計算」を理解するため、
数学の新しい概念が数多く
作られてきた

> *"$\mathcal{P}$ versus $\mathcal{NP}$ – a gift to mathematics from Computer Science"*
>
> *Steve Smale.*

P ≠ NP 予想 ― 計算機科学から数学への贈り物

スティーブ・スメール
フィールズ賞 1996年