Unsolvable P versus NP

Hajime Mashima

October 4, 2020

Abstract

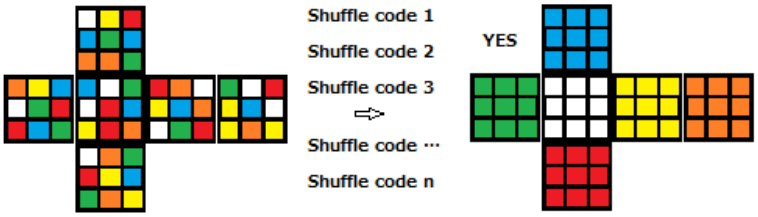
This question (P versus NP) asks whether or not there is an algorithm, and it is famous that it was formalized and submitted as a sweepstakes question at the Clay Mathematics Institute in 2000.

1 introduction

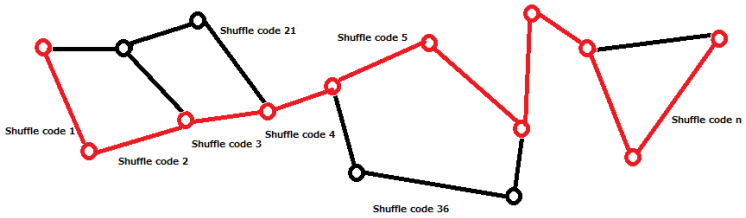
P versus NP は原始的な疑問から始まり定式化を経ても尚、概念の理解が要求される問題である。その定式化とは何か、自身の見解とした上で説明していく。P versus NP の原始的な疑問はおそらく「如何なる難解な問題でも容易に解けるようなアルゴリズムは存在するだろか？」である。しかし例えば問題に非決定論性や任意性が内在したり、既に最小手順であるにも関わらず不明なもの。あるいはアルゴリズム作成に関わる規則性が著しく乏しいか、逆に多様で複雑である条件では、アルゴリズムの作成は不可能に近いと考えられる。そもそも難解や容易など抽象的なものをどのように扱ったら良いだろうか？そこで用いられるのが計算量オーダー $\mathcal{O}$ である。これは計算回数の grade に相当するもので、計算量オーダーが大きいほどコンピューターの処理などに時間がかかることを意味している。また総当たりの計算量オーダーから、どのくらい手順を省けたら効率的なアルゴリズムと言えるのか定める必要があるだろう。そのための線引は計算量オーダーで多項式時間以内となるものを容易、それを超えるものは難解としている。
以上より「多項式時間を超える問題を多項式時間以内で解けるアルゴリズムは必ず存在するだろうか？」と定式化できた。それでもまだ十分とは言えない。先に挙げたような不可能と推定される諸条件、何より想定外の条件を想定することはできないため、問題を現実的と推定される対象範囲に制限する必要がある。P versus NP は、それを NP としたものであるという前提で証明できない説明をする。

Example 1

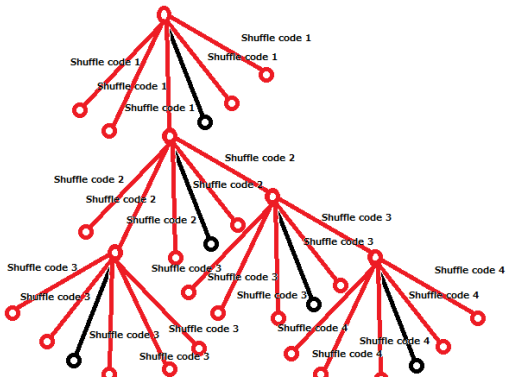
Rubik's Cube の一回の Shuffle を code に対応させたものを Shuffle code とする。
実際の Cell 数は十分多いものと仮定する。
Rubik's Cube を完成させる code は簡単のため n までの連番とする。
必要であれば Cell に数字を書き、対称性を排除したり都合よく設定を変更しても構わない。



次に点と線で構成される field を考える。やはり field の点と線の数も十分多いと仮定する。一つの線は一つの Shuffle code に対応しており、点から点へ線を伝って進むとき Rubik's Cube は対応する Shuffle が行われる。



次に、この field に無限近く存在する fake となる分岐を加えるが、実際には正解 Shuffle code を推測できるような偏りはないものとする。



Definition 2

- 判定者：yes(解) の正当性を判定する人
- 探索者：アルゴリズムを探索する人

Question 1 : 乱雑な Rubik's Cube を最短で完成させる Shuffle code が field に一つだけ存在している。その経路を多項式時間で見つけなさい。

上図において探索者が総当りで見つけた正解 Shuffle code の経路を判定者に提示した場合、判定者は正解となる経路から多項式時間で yes を検証できる。
しかし探索者が正解 Shuffle code のみを予め知っていたとしても、分岐を増やしただけの経路から正解 Shuffle code を多項式時間で見つけるアルゴリズムは存在しない。
Question 1 は探索者が正解 Shuffle code を知らない前提なので結論は同じである。
これは P≠NP となる問題を特定できたように思えるが、経路が Rubik's Cube を完成させるアルゴリズムと直接的には関係ないものである。
また判定者が偶然に多項式時間で yes を検証してしまう場合なども考えられるが「如何なる難解な問題でも容易に解けるようなアルゴリズムは存在するだろか？」を調べる目的にそぐわないものは除外される前置きをする必要がある。

1.1 単独判定問題

Definition 3

- 単独判定問題：1 通りの検証で yes を判定できる問題

Example 4

- ハミルトン閉路問題 etc(Hamiltonian path problem)

計算量オーダー $\mathcal{O}(1)$ は任意に設定できるが、総当たり計算量オーダーの単位ステップとするのが常套である。$\mathcal{O}(1)$ が例えば 0.1 秒かかるならば $\mathcal{O}(600)$ は 1 分かかると見積もることができる。つまり計算量オーダーは回数の概念であり時間の概念ではないので単独判定問題は検証する時間によらず $\mathcal{O}(1)$ となる。
仮に $\mathcal{O}(1)$ の次元を拡張して多項式時間 $\mathcal{O}_d(n^2)$ かかる

$$\mathcal{O}(1) = \mathcal{O}_d(n^2)$$

と定義しても $\mathcal{O}_d$ が $\mathcal{O}$ に影響を与えるわけでないため、多項式時間で検証可能ということが問題の対象範囲を制限している根拠とならないと考えられる。

1.2　比較判定問題

Definition 5

- 比較判定問題：2 通り以上の検証で yes を判定できる問題

Example 6

- 巡回セールスマン問題 (traveling salesman problem)
- ナップサック問題 (Knapsack problem)

A〜F の中で最も重いボールを探すアルゴリズムを考える。
このとき探索者は正解を知らない。判定者は正解を知っている優位性を持つ前提とする。

ball	weight(kg)
A	5
B	13
C	1
D	17
E	11
F	23

1.2.1　1:2 比較値

【探索者】

$$
\begin{aligned}
A \not> B &\quad A \textit{ is not the heaviest.} \\
C \not> D &\quad C \textit{ is not the heaviest.} \\
E \not> F &\quad E \textit{ is not the heaviest.} \\
B \not> D &\quad B \textit{ is not the heaviest.} \\
D \not> F &\quad D \textit{ is not the heaviest.}
\end{aligned}
$$

【判定者】

$$
\begin{aligned}
F > A &\quad F \textit{ is the heaviest.} \\
F > B &\quad F \textit{ is the heaviest.} \\
F > C &\quad F \textit{ is the heaviest.} \\
F > D &\quad F \textit{ is the heaviest.} \\
F > E &\quad F \textit{ is the heaviest.}
\end{aligned}
$$

1.2.2 1:3 比較値

【探索者】

$$A \not> B, C \quad A \text{ is not the heaviest.}$$
$$B \not> C, D \quad B \text{ is not the heaviest.}$$
$$C \not> D, E \quad C \text{ is not the heaviest.}$$
$$D \not> E, F \quad D \text{ is not the heaviest.}$$
$$E \not> F \quad E \text{ is not the heaviest.}$$

【判定者】

$$F > A, B \quad F \text{ is the heaviest.}$$
$$F > C, D \quad F \text{ is the heaviest.}$$
$$F > E \quad F \text{ is the heaviest.}$$

1.2.3 1:n 比較値

【探索者】

$$A \not> B, C, D, E, F \quad A \text{ is not the heaviest.}$$
$$B \not> C, D, E, F \quad B \text{ is not the heaviest.}$$
$$C \not> D, E, F \quad C \text{ is not the heaviest.}$$
$$D \not> E, F \quad D \text{ is not the heaviest.}$$
$$E \not> F \quad E \text{ is not the heaviest.}$$

【判定者】

$$F > A, B, C, D, E \quad F \text{ is the heaviest.}$$

1:n 比較可能なアルゴリズムでは、単独判定問題のように判定者は 1 回の検証で済むが、探索者は、このアルゴリズムでも全探索となる。
同様に多項式時間で検証可能ということが問題の対象範囲を制限している根拠とならないと考えられる。

1.2.4　2 グループの最大値

【探索者】

$$\begin{aligned} Max(A,B) &= B \quad B \textit{ is the heaviest.} \\ Max(B,C) &= B \quad B \textit{ is the heaviest.} \\ Max(B,D) &= D \quad D \textit{ is the heaviest.} \\ Max(D,E) &= D \quad D \textit{ is the heaviest.} \\ Max(D,F) &= F \quad F \textit{ is the heaviest.} \end{aligned}$$

【判定者】

$$\begin{aligned} Max(A,F) &= F \quad F \textit{ is the heaviest.} \\ Max(B,F) &= F \quad F \textit{ is the heaviest.} \\ Max(C,F) &= F \quad F \textit{ is the heaviest.} \\ Max(D,F) &= F \quad F \textit{ is the heaviest.} \\ Max(E,F) &= F \quad F \textit{ is the heaviest.} \end{aligned}$$

1.2.5　3 グループの最大値

【探索者】

$$\begin{aligned} Max(A,B,C) &= B \quad B \textit{ is the heaviest.} \\ Max(D,E,F) &= F \quad F \textit{ is the heaviest.} \\ Max(B,F) &= F \quad F \textit{ is the heaviest.} \end{aligned}$$

【判定者】

$$\begin{aligned} Max(A,B,F) &= F \quad F \textit{ is the heaviest.} \\ Max(C,D,F) &= F \quad F \textit{ is the heaviest.} \\ Max(E,F) &= F \quad F \textit{ is the heaviest.} \end{aligned}$$

当該条件で A_1～A_{99} の中で最も重いボールを探すアルゴリズムを考える。

ball	weight(kg)
A_1	9
A_2	24
A_3	13
A_4	53
A_5	61
A_6	13
～	～
A_93	1
A_94	62
A_95	56
A_96	88
A_97	74
A_98	44
A_99	heaviest

【探索者】

$$
\begin{aligned}
Max_1(A_1, A_2, A_3) &= Max_1 \quad Max_1 \textit{ is the heaviest.} \\
Max_2(A_4, A_5, A_6) &= Max_2 \quad Max_2 \textit{ is the heaviest.} \\
Max_3(A_7, A_8, A_9) &= Max_3 \quad Max_3 \textit{ is the heaviest.} \\
&\vdots \\
Max_{33}(A_{97}, A_{98}, A_{99}) &= Max_{33} \quad Max_{33} \textit{ is the heaviest.}
\end{aligned}
$$

$$
\begin{aligned}
Max'_1(Max_1, Max_2, Max_3) &= Max'_1 \quad Max'_1 \textit{ is the heaviest.} \\
Max'_2(Max_4, Max_5, Max_6) &= Max'_2 \quad Max'_2 \textit{ is the heaviest.} \\
Max'_3(Max_7, Max_8, Max_9) &= Max'_3 \quad Max'_3 \textit{ is the heaviest.} \\
&\vdots \\
Max'_{11}(Max_{31}, Max_{32}, Max_{33}) &= Max'_{11} \quad Max'_{11} \textit{ is the heaviest.}
\end{aligned}
$$

$$
\begin{aligned}
Max''_1(Max'_1, Max'_2, Max'_3) &= Max''_1 \quad Max''_1 \textit{ is the heaviest.} \\
Max''_2(Max'_4, Max'_5, Max'_6) &= Max''_2 \quad Max''_2 \textit{ is the heaviest.} \\
Max''_3(Max'_7, Max'_8, Max'_9) &= Max''_3 \quad Max''_3 \textit{ is the heaviest.} \\
Max''_4(Max'_{10}, Max'_{11}) &= Max''_4 \quad Max''_4 \textit{ is the heaviest.}
\end{aligned}
$$

$$
Max'''_1(Max''_1, Max''_2, Max''_3) = Max'''_1 \quad Max'''_1 \textit{ is the heaviest.}
$$

$$
Max''''_1(Max'''_1, Max''_4) = Max''''_1 \quad Max''''_1 \textit{ is the heaviest.}
$$

【判定者】

$$
\begin{aligned}
Max_1(A_1, A_2, A_{99}) &= A_{99} \quad A_{99}\ \textit{is the heaviest.} \\
Max_2(A_3, A_4, A_{99}) &= A_{99} \quad A_{99}\ \textit{is the heaviest.} \\
Max_3(A_5, A_6, A_{99}) &= A_{99} \quad A_{99}\ \textit{is the heaviest.} \\
&\vdots \\
Max_{49}(A_{97}, A_{98}, A_{99}) &= A_{99} \quad A_{99}\ \textit{is the heaviest.}
\end{aligned}
$$

探索者の処理回数は、$33 + 11 + 4 + 1 + 1 = 50$ なので n グループの最大値を知るアルゴリズムでは処理回数が概ね判定者と等しい。