A_001

「Ｐ＝ＮＰ？」問題の最終解決：Part 2
The Final Solution of "P=NP?" Problem : Part 2

山口人生
Jinsei Yamaguchi

１．まえがき

以下、計算量理論における"*P*"、"*NP*"、"*EXP*"等の概念は周知のここと仮定する。また、便宜上、従来の計算量理論のことを、「素朴 CCT（"the naïve Computational Complexity Theory"）」と呼ぶ。何故、" 素朴" という用語を使用するのかについては、後に言及する。

本発表の目的は、有名な難問「*P*＝*NP*？」問題を素朴 CCT 上で否定的に証明することである。同時に、「*NP*＝*EXP*？」問題についても、素朴 CCT 上で否定的に証明する。

２．ＮＰ≠ＥＸＰ

順序として、まず、「*NP*≠*EXP*」から証明しておこう。

定理２．１（素朴 CCT）
NP≠*EXP*。

証明：
以下、判り易さを重視して、証明を４段階に分ける。

段階１
各論理式 α に対し、*TT*(α)を α の真理表とし、|*TT*(α)| を *TT*(α)に出現する" *1*（真）" の個数とする。この時、例えば、次のような判定問題を考えてみる。

" $|TT(\alpha)| < (2^{n-1}+n)$？" …(1)

ここで、入力は α であり、n は α の次数（アトム数）である。以下、便宜上、問題(1)を *Q*(*1*, α) と呼び、(1)を実現するアルゴリズム全体の集合を *A(1)*とする。ところで、各 α に対し、*TT*(α) を構成する高々*EXP* のアルゴリズムが存在する。よって、*Q*(α) は最悪でも *EXP* に属する。 ■

ここで、(1)には" *NP* の意味で、より速いアルゴリズム" が存在しないことを示せば、「*NP*≠*EXP*」を証明したことになる。しかしながら、「*NP*≠*EXP*」を証明する新ルートが存在する。以下、我々は、この別ルートを登ることにする。

段階２
別の判定問題*Q*(*2*, α)、例えば、"$|TT(\alpha)| > (2^{n-1}-n^2)$?"を考える。そして、*Q*(*2*, α)を実現するアルゴリズム全体の集合を *A(2)*とする。この時、*Q*(*1*, α) と *Q*(*2*, α)から、次のような判定問題が得られる。

" *Q*(*x*, α), where $x \in \{1, 2\}$" …(2)

この(2)も *EXP* に属する。 ■

(2)を実現するアルゴリズム λ は *A(1)*∪*A(2)*を基盤にしている。つまり、λ は $\sigma \in A(1)$と $\tau \in A(2)$から構成される。そして、{*1*, *2*}から、例えば、パラメータ値２を指定すると、λ は τ 部分に制限されることになる。以下、この種の戦略を "ASBP(algorithm separation based on a parameter)"と呼ぶことにする。

段階３
A(*1*, *s*)＝ {$\sigma \in A(1)$ | *Q*(*1*, α)を判定するために σ は *TT*(α) 全体を生成し、検査する} …<#>
を用いて、*A(1)*＝*A*(*1*, *s*) ∪ *A*(*1*, *p*)と分離する。ここで、*A*(*1*, *p*) ＝ ∅ ならば、上の段階１で「*NP* ≠ *EXP*」の証明は終了。よって、以下、*A*(*1*, *p*) ≠ ∅ と仮定する。この分離により、新しい判定問題

" *R*(*x*, α), where $x \in \{s, p\}$" …(3)

を考えることが可能になる。ここで、*R*(*x*, α)の定義は、

"*R*(*x*, α) is *Yes* iff *Q*(*1*, α) is *Yes* based on an algorithm $\sigma \in A(1, x)$."

より詳しく定義すれば、次のようになる。

R(*x*, α)の出力は(*i*, *j*)の形をしている。ここで *i*, *j*∈ {*1*, *0*}であり、*i* は *x* のチェック、*j* は α のチェックの役割を果たす。例えば、*R*(*s*, α)が(*1*, *0*)を出力するのは、

「*A*(*1*,*s*)のアルゴリズムで *Q*(*1*, α)が *No*」

の場合のみである。同様に、*R*(*s*, α)が(*0*, *1*)を出力するのは、

「*A*(*1*,*s*)以外のアルゴリズムで *Q*(*1*, α)が *Yes*」

の場合のみである。つまり、各アルゴリズムに対し、*A*(*1*, *x*) への所属チェックは必ず実行される。

この定義の下、出力(*1*, *1*)を" 超真（メタ *1*）" と呼ぶ。この新概念は従来なかったものである。そして、*R*(*x*, α) が *Yes* とは *R*(*x*, α)が超真になることを言う。この判定には、必ず出力(*i*, *j*)の経由が必要という点がポイントである。■

(3)は ASBP 戦略で生成されている。段階２では、{*1*, *2*} の選択が{*A(1)*, *A(2)*}の選択に対応していた。同様に、ここでは、{*s*, *p*}の選択が{*A*(*1*, *s*), *A*(*1*, *p*)}の選択に対応している。

段階４
A(*1*, *s*)に着目して、ある性質 *T* を用いて、

A(*1*, *s*)＝*A*(*1*, *s*, *u*) ∪*A*(*1*, *s*, *v*)

と分離する。この *A*(*1*, *s*, *u*)と *A*(*1*, *s*, *v*)を用いて、新しい判定問題

" *S*(*x*, α), where $x \in \{u, v\}$" …(4)

を考えることが可能になる。ここで、(4)の定義は

"*S*(*x*, α) is *Yes* iff *Q*(*1*, α) is *Yes* based on an algorithm $\sigma \in A(1, s, x)$."

この場合も、詳しく言えば、超真で定義される。明らかに、

㈱Ｉ．Ｉ．Ｉ．代表取締役会長

(4)は *EXP* に属する。さらに、*A(1, s)*の定義により、(4)は *NP* には属さない。かくして、(4)が「*NP* ≠ *EXP*」の witness になる。　■　QED

３．Ｐ≠ＮＰ

集合論的には、“ 関数 *f*” なる概念は
「*y=f(x)*なる*(x, y)*の集合」
として定義される。同様に、“ 関係 *R*” なる概念は
「*R(x)*なる*x*の集合」
として定義される。一方、CCT の観点から見れば、関数 *f* や問題 *Q* は
「その関数（問題）を実現するアルゴリズムの集合」
で決定される。このアイデアこそ、CCT の基本思想である。何故ならば、*f* や *Q* の計算量の概念は、*f* や *Q* を実現する様々なアルゴリズムの計算量の中で“ 最小のもの” として定義されるからである。これによって、
「SAT は *P*？」
のような質問が可能になる。
(厳密に定義するには、“Turing Machine”及び“(入力)言語”の概念が必要になる。しかしながら、本質は同じである。)
以上の認識に基づいて、次のような重要な事実を提示しておこう。

定理３．１（素朴 CCT)
定理 2. 1 の A(1, s)は CCT の意味で正当である。つまり、ある決定問題を規定する。

証明：
*A(1, s)*が集合論の意味で定義可能なことは、すでに示しておいた。よって、確認すべきは、任意のアルゴリズム$\lambda \in A(1)$に対し、
「$\lambda \in A(1, s)$？」…(5)
が決定可能かどうかという点である。ここで、λ が条件<#>を満たすかどうかのチェックが複雑過ぎると思う研究者に対しては、<#>の代わりに、以下のような、より具体的な制約を準備すればよい。
「入力αに対し、*TT(α)*を生成する」
というアルゴリズムモジュールをρとする。そして、$\rho+\mu$ なる形の *A(1, s)*内のアルゴリズムを“ρ-extension in *A(1)*”と呼ぶ。そして、一般の *A(1, s)*の代わりに、
A(1, ρ*)*＝ {λ | λはρ-extension in *A(1)*}
を利用する。この場合、“$\lambda \in A(1, \rho)$” のチェックは容易になる。**QED.**

同様の議論を経て、定理２．１全体の正当性が示される。但し、段階３や段階４の実現には本質的な難しさがある。例えば、(4)は（パラメータ *x* を無視した）*Q(1, α)*の勝手なアルゴリズムでは代用実現できない。つまり、実行するアルゴリズムの特徴を指定する必要がある。これが“ 超” の意味である。この種の問題を“ 自己アルゴリズム言及問題” と呼ぼう。ここでの課題は、段階４の実現であるが、これは可能である。詳しい議論は、[３]を参照せよ。　全く同様にして、次の結果も得られる。

定理３．２　(素朴 CCT)
P ≠ *NP*。

証明：
FT(SAT)＝ {λ | λ は SAT の全解探索アルゴリズム} …<$>
とする。（各 *TT(α)*を、上から順に調べる。）更に、
FT(SAT) = FT(SAT)(a) ∪ *FT(SAT)(b)*
と分離する。この時、*FT(SAT)*を用いて、ASBP 戦略により“ FTS-SAT” なる２入力判定問題を次のように定義する。
“(*x, α*) is *Yes* in FTS-SAT iff α is satisfiable based on an algorithm in *FT(SAT)(x)*, where $x \in \{a, b\}$.”
ここでも、超真を用いている。定義により、FTS-SAT は *NP* に属するが *P* には属さない。よって、FTS-SAT が *P* ≠ *NP* の証拠になる。　**QED**

４．まとめ

上の証明が不適切だと思う研究者は、ASBP 戦略について、正当なものと、そうでないものとを区別する公理を定義する必要がある。例えば、段階２までは、誰しも容認する ASBP 戦略である。では、段階３は、どうなのか？段階３では、候補アルゴリズムを制限する方法論（AR 法）が採用されている。これに関し、次の驚くべき事実がある。

任意の論理式 α は、α の真理表で定義される関数を実現するアルゴリズムの内、特に、“（アルゴリズム）シェーマ α で表現された計算を実行するもの” に制限されたアルゴリズムだけに対応している。つまり、AR 法は、計算量理論において、普通に使用されている。そして、上では、決定問題としてＯＫだという点を確認した。

しかしながら、こういう方法は、受理言語の観点からも許容可能か？
「決定問題を言語認識問題に特化することで、上述のような自己アルゴリズム言及問題は排除できるのでは？」
これに対する解答は、一筋縄ではいかない。実は、この課題を追求していくと、究極的には、状況依存性が顕現してくるのである。つまり、普遍の枠外議論になってくる。この点が、「*P*=*NP*？」問題の最大の難関である。これに関しては、次回の発表で論じる。

いずれにせよ、２１世紀からは、CCT は“ 公理化” という新時代に入る。公理化問題抜きに、「*P*=*NP*？」問題は語れないのだ。

参考文献

[1]山口人生、*「P=NP?」問題の解決*、Proceeding of IPSJ64, Vol.1, 183-184, 2002.
[2]山口人生、*計算量理論の存亡（１）：「P=NP?」問題の解決*、I.I.I., 2002.
[3]山口人生、*「P=NP?」問題の解説*、I.I.I.社の情報サイト http://www.int2.info/news1.htm
[4]Yamaguchi, J., *A Proof of P≠NP: Toward the Axiomatic Computational Complexity Theory*, to appear.