計算できない問題について

山崎秀記

0　はじめに

コンピュータはいまや身近なものになり、いろいろな場面で利用されている。思い付くままに並べてみても、ワープロ、銀行での預金管理などのデータ処理、機械翻訳、テレビゲーム、など様々な話題が新聞に載っている。最近ではコンピュータ（パソコン）通信がはやり、株の売買をパソコン通信を利用して行いませんかという広告も見かける。これら外見上様々な仕事をするコンピュータもハード的には同一の原理に従って動作するものであり、適当なプログラムを与えることによりこれらの仕事をさせることができる、ということは御存知だろう。

コンピュータは最初は弾道の計算やロケットの軌道の計算など、数値の計算に利用されていた。つまりプログラムは数値計算のプログラムがほとんどだった。もちろん今でも数値計算プログラムの重要性は言うまでもないが、上にあげた例にも見られるように、その応用は数値計算だけに限られるものではなくなっている。そして、コンピュータは適当なプログラムさえ与えれば、どんな仕事もこなしてくれる、あるいはどんな問題でも計算してくれるようにも見える。そこで本当にコンピュータはなんでもできるのだろうか、という問を考えてみたい。この問に対する議論を通して、計算可能性理論というコンピュータ・サイエンスの基礎的分野の入門、紹介をしたいというのがこの小文の目的である。

この問に対する答えおよび問自身に対する批判として

いくつか考えられる。例えば、人間の創造的能力に関する事柄、音楽を作曲するとか、絵を描くとかいう仕事はコンピュータにはできない、つまり作曲プログラムはできないという議論もあろう。

しかし、これにはそもそも仕事の定義自体が、少なくとも数学的に、定式化できないという疑問がある。音楽を作曲するとは、何を持って作曲したというのか。ただ音を並べるだけでよいのなら、そのプログラムは簡単に作れるだろう。しかし、音の並びがどういう条件を満たしていれば、作曲をしたというのにふさわしいかを、きちんと述べることなど期待できないだろう。逆に言うと、もしその様なことが可能なら、つまり、どのような音の並びを我々は美しいと感じるかをきちんと定式化できるなら音楽を作曲するプログラムも可能になるかも知れない。そこで、数学的に定式化できる仕事だけを考えることにする。

次に、円周率を十億桁求める、という仕事を考えてみよう。これは、現在のコンピュータを使って、普通の時間内では行うことができない。実際、現在求められている円周率の最大桁数は約二百万桁である。しかし、これはコンピュータの計算スピードが発達すれば、あるいは計算時間やデータの記憶装置をいくら使ってもよいとすれば、原理的には計算できる。現在あるプログラムそのままか、あるいはごく簡単な変更で、いくらでも多くの桁数の円周率を計算するプログラムが作れるだろう。

同様のことはもっと簡単な例にも見られる。例えばかけ算がコンピュータで計算できるかと聞かれれば、だれでも計算できると答えるだろう。しかし、桁数が二の十兆乗桁位の整数同士のかけ算がやはり現在のコンピュータで許しえる時間内に計算可能とは思えないだろう。だからといって、かけ算の計算が（一般的には）できないというのでは議論にならない。そこで以後、計算時間や計算のための記憶領域はいくらでも使用してよいという（理想的な）コンピュータを仮定する。

また、その仕事あるいは問題を解決する方法をまだ知らないために、解くことができないという例はいくらでもある。そのような問題はその問題に対する我々の知識が足りないために解けないのであって、いつかはその問題を解く方法を手にいれ、その暁には、コンピュータにも解けるようになるだろう。ここでは、そのような問題

も、除いて考えることにしたい。

では問題を数学的にきちんと定式化でき、コンピュータがいくらでも記憶装置を用いてよく、また計算時間をいくらでもかけてよいとして、なおかつ、その問題に関する我々の知識がいくら増えたとしても、それでもなおコンピュータで解けない問題が存在するであろうか。言い替えると、その問題を解くプログラムの存在自身が矛盾を導くような問題があるかということである。

その答えは yes である。それについて、これから説明をしたい。なお、この小文では、(理想の)コンピュータで計算できない問題があるか、という形で問を設定したが、この問は実は人間に計算できない問題があるかという問と同じであると(少なくとも計算機科学者の間では)信じられている。ある問題に対し、その手順に従えばだれでもその問題が解けるような具体的な手順があることと、それを解くコンピュータのプログラムがあることとは同値である、というチャーチの仮説は現在広く受け入れられている。したがって、その問題を解くための具体的な手順の存在自体が矛盾を導くような問題が存在する。本題にはいる前に、この問は、問がはっきりと定式化される以前は、即ち、計算できるとはどういうことかが定義される以前は、答えは no であろうと予想されていたことに注意しておきたい。

1 プログラムの停止性判定

本論に入る前に、いくつかの点をもうすこし厳密に定めておきたい。(といっても、この小文の性格上、完全に厳密というわけにはいかない。)まずコンピュータが解く(あるいは解かせたい)問題を数学的に定式化するとは、一体何を定めればよいのだろうか。最初にも述べたように、コンピュータは数値の計算をするだけの機械ではない。もっと一般的に、何等かの入力が(いくつか)あって、それに対し何等かの出力を出す機械と考える。では、入力や出力は何かといえば、それは適当に定められた文字(記号)を並べた文字列であると考える。入力や出力を構成する有限個の文字の集合は、プログラムしようとする仕事によって適当に定めればよい。考えてみれば、数値計算も、入出力は数字の列であると考えれば上記の範疇にはいる。

そして簡単のために、コンピュータが停止したときだ

けその出力を考えることにし、コンピュータが計算を続けている間はなにも出力されないものとする（あるいはされたとしても無視する）。また、コンピュータが停止するのは停止命令に出会ったときだけとする（プログラムの最後には必ず停止命令があるものとする）。

次に、コンピュータにさせたいと思う仕事は次のようにして、定式化する。それは与えられた個数の与えられた条件に合った文字列をその入力として定める（これを入力条件という）。そして、与えられた入力条件に合う入力に対し、どの様な出力を出せばよいかを定める（入出力関係）。したがって、問題は与えられた入力条件を満たす任意の入力に対し、与えられた入出力関係を満たす出力を求めるプログラムが存在するか、ということである。

では、コンピュータに解けない問題を説明しよう。以下の議論では適当なプログラム言語を固定しよう。BASICでもFORTRANでもまた機械語であってもかまわない。与えられたプログラム（考えているプログラム言語の文であるから、ある記号列）とそのプログラムの入力に対し、プログラムがその入力のもとで、停止するか否かを判定せよという**停止性判定問題**を考える。この停止性判定問題を解くプログラムは存在しない。それを証明しよう。問題を解くプログラムと、問題の入力としてのプログラムとがあるので、読者は混乱しないように気を付けて欲しい。

定理1　次のような停止性判定プログラムHは存在しない。

Hの入力：記号列XとW

Hの出力：次の条件が満たされていれば1、さもなければ0。

Xは正しいプログラムであり、Xが入力Wのもとで停止する。

証明　このプログラムHが書けたとしよう。このプログラムはあらゆる入力に対して停止しなければならないことに注意しよう。Hを変形して次のようなプログラムGを作る。

Gの入力：記号列X

Gの出力：入力X、XをプログラムHに入れたとき、Hが（停止して）1を出力するなら

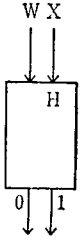
Gは停止しない。
Hが0を出力するならはGは停止する。

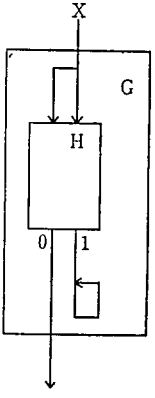
このプログラムはプログラムHを次のように変形すればよい。出力命令があったらそれを特別の変数aに出力すべき値を代入する命令に置き換える。プログラムが停止するのは停止命令にであったときだから、停止命令をaの値が1なら例えば次のような無限にループする命令とそれにつづく停止命令に置き換えればよい。

```
（無限ループ）　10: if a=1 then go to 10
                stop
```

さて、Gに入力として自分自身Gを入れてみよう。Gは停止するだろうか。Gが停止すれば、それはHに入力としてG、Gを入れたとき出力が0であることを意味するから、Hの性質からGは入力Gにたいし停止しない。これは矛盾である。一方、Gが停止しないときは、入力G、Gに対するHの出力は1である。これはGが入力Gのもとで停止することを意味し、これも矛盾である。

どちらにしろ矛盾であり、この矛盾が起きた理由は停止判定プログラムHの存在を仮定したことにあったので、そもそもHが存在しないことが証明された。□





ここで、問題をよりハッキリさせるために、停止性判定問題を二つに分けてみよう。上で存在しないことが証明されたプログラムは与えられたプログラムが与えられた入力に対し停止するか否かをそのどちらの場合にも数えてくれるプログラムである。そこで、これを、プログラムが停止することを教えてくれるプログラム（停止しないことは教えてくれなくてもよい）と、プログラムが停止しないことを教えてくれる（停止することは教えてくれなくてもよい）プログラムとに分けて考えてみよう。

定理2　次のようなプログラムH′は存在する。

H′の入力：記号列XとW

H′の出力：次の条件が満たされていれば（停止して）1を出力する。
Xは正しいプログラムであり、Xが入力Wのもとで停止する。

証明　記号列Xがあらわすプログラムの動作を入力Wのもとで模倣するプログラムを書けばよい。これはいわゆるインタプリタとよばれるもので、一般にコンピュータがプログラムを解釈し、そのプログラムの意味通りに実行するさいに必要とされるプログラムである。このようなプログラムが書けることは明かだろう。　□

定理3　次のようなプログラムH″は存在しない。
H″の入力：記号列XとW
H″の出力：次の条件が満たされていれば（停止して）0を出力し、さもなければ停止しない。
Xが入力Wのもとで停止するプログラムではない。

証明　このプログラムH″が書けたとしよう。H″を変形して次のようなプログラムGを作る。

Gの入力：記号列X
Gの出力：X、XをプログラムH″に入れたときのH″の出力。

さて、Gに入力として自分自身Gを入れてみよう。Gは停止するだろうか。Gが停止すれば、それはH″に入力として、G、Gを入れたとき出力が0であることを意味するから、H″の性質からGは入力Gにたいし停止するプログラムでないから矛盾である。一方、Gが停止しないときは、入力G、Gに対するH″の出力は0である。これはGが入力Gのもとで停止することを意味し、これも矛盾である。

どちらにしろ矛盾であり、この矛盾が起きた理由は停止判定プログラムH″の存在を仮定したことにあったので、そもそもH″は存在しない。　□

これらにより、停止性判定問題が計算可能でないのは、結局非停止性を調べることができないためであることがわかる。また、後半のような意味でのプログラムの存在は否定に関して対称的でないこともわかる。上の証明の中で、入力としてのプログラム（を表す文字列）とその

入力を処理するプログラムとが巧みに二重に使われていることに注意して欲しい。

繰り返しになるかも知れないが、プログラムHとプログラムH′の違いを見てみよう。プログラムHに要求されていたことは、与えられたプログラムがその入力にたいし停止するしないどちらの場合にもそのことを教えてくれることであり、そのようなプログラムHは、停止しないことを教えてくれるプログラムH″が存在しないので、存在しない。

一方、プログラムH′では、与えられたプログラムがその入力に対し停止する場合にだけそのことを教えてくれればそれでよい。停止しない場合には、それを教えてくれる必要はない。そしてそのようなプログラムH′は存在する。

与えられたデータにたいし、それがある性質を満たすか否か判定せよという判定問題は、答えがｙｅｓ、ｎｏどちらの場合にも答えてくれるようなプログラムが存在するとき、**計算可能**であるという。したがって、与えられたプログラムが与えられた入力に対して停止するかという停止性判定問題は、計算可能でない。

一方、判定問題に対し、その答えがｙｅｓであるようなデータにだけそのことを答えてくれるプログラムが存在するとき、その判定問題は**半計算可能**であるという。したがって、停止性判定問題は半計算可能である。そして、非停止性（停止しないことの）判定問題は半計算可能でさえない。

このように、半計算可能性というのは、否定に関して対称的でない。ところで、ある判定問題とその否定の判定問題がともに半計算可能なら、その判定問題は計算可能である。

定理4 判定問題は、それ自身とその否定がともに半計算可能のときそのときのみ、計算可能である。

証明 判定問題をPとする。Pが計算可能のとき、Pとその否定はともに計算可能であり、したがってともに半計算可能である。逆に、Pを半計算するプログラムFとPの否定を半計算するプログラムGがあるとき、Pを計算するプログラムHを次のように構成することができる。すなわち、Hはその入力に対し、FとGを平行して動かし、どちらかから返ってきた答えを答えとする。Fから

返ってくればｙｅｓ、Ｇから返ってくればｎｏである。入力に対しＦ、Ｇのどちらかは必ず停止するので、Ｈは必ずｙｅｓかｎｏの答えを返すプログラムであり、Ｐは計算可能である。　□

2　計算可能という言葉について

1節では、(理想的な)計算機上である問題を解くプログラムが存在するときに、その問題は(半)計算可能であると定義した。この言い方がふさわしいかについて、すこし議論してみたい。

まず、計算可能性を定義する土俵をどこにするかによって、実は大きく分けて二つの流儀がある。一つは自然数の上で定義するもの(いわゆる帰納的関数論)と、この小文でも説明したように記号列の上で定義するやり方である。最初にも注意したように、自然数は適当な表記法を仮定すれば記号列として扱える。(例えば10進法、そして表記法を仮定せずに自然数を書き表すことはできない。)逆に、記号例は自然数で表せるだろうか。まず、記号をそれぞれある奇数と対応させる。例えばａは1、ｂは3、ｃは5、……。すると文字列に対し、対応する数列ができる。そして数列ｉ、ｊ、ｋ、……を数

$$2^i3^j5^k\cdots\cdots$$

で表す。ここで、2、3、5、……は素数を順に持ってきたものである。したがって文字列ｂｃａｂは、数

$$2^33^55^17^3$$

で表される。ここで重要なのは、数を記号列に変換したり、逆に記号列を数に変換したりする計算が、ともに可能であるということである。こうして数と記号列は互いに変換可能であり、どちらの土俵で計算可能性を議論しても同じになる。

さてコンピュータプログラムが存在するという意味での計算可能性はどの様な性質を持っているだろうか。

第一に、プログラムが存在するという概念は、強い不変性を持つ。すなわち、元になる計算機がどの様な能力を持つか、基本的なデータとして何を扱い(自然数か記号列かなど)、演算としてなにが許されているかに関わらず、定義される概念は(適当な符号化によって)同値なものになる。(実はこのことに関しては暗黙のうちに仮定にしてきた。)

第二に、計算可能と呼べる候補として考えられた数多

くのシステムによる定義はすべて、上に述べた定義と一致することが知られている。具体的な説明をする余裕はここではないが、それには、計算機の数学的モデルであるテューリング機械や、論理学者達によるλ－計算、ポストのシステム、帰納的（部分）関数などがあり、さらにはチョムスキーの句構造文法、マルコフのアルゴリズムなどもある。

以上のことから、計算機のプログラムが存在することをもって、あるいはそれと同値な他の定義でもって、（半）計算可能の定義としてよいというのがチャーチの仮説であり、現在広く受け入れられている。

3 ライスの定理

2節で、プログラムが停止するか否かを判定するプログラムは存在しないことを証明した。プログラムに関する他の性質はどうだろうか。例えば、そのプログラムが10個以上の変数を使っているか否かという問題はどうだろうか。これはプログラムの字面を眺めてみればわかることなので、実際にそのようなプログラムを書くのは面倒だとしても、そのプログラムが存在することは納得できるだろう。では、どういう問題が計算可能で、どういう問題が計算可能でないのだろうか。

プログラムは何かある計算をするのだけれど、一般に同じ計算結果を持つプログラムは数多くある。そこで、プログラムの計算結果、あるいはその入出力関係が（プログラムが停止するしないを含めて）どういう性質を満たすかを判定する問題を考える。これをプログラムの計算結果に関する判定問題ということにする。この問題では、プログラムの字面は問題ではなく、字面が異なっていても同じ計算をするプログラムに対しては、同じ答えが返らなければならない。

プログラムの計算結果に関する判定問題は、それが自明な問題でない限り、計算可能ではない。プログラムの判定問題が自明であるというのは、すべてのプログラムに対してその問題の答えがｙｅｓであるか、またはすべてのプログラムに対してその問題の答えがｎｏのときである。自明な判定問題が計算可能であるのは明かだろう。

定理5（ライスの定理）

プログラムの計算結果に関する判定問題はそれが自明

でなければ計算可能でない。

証明　Pをプログラムの計算結果に関する自明でない判定問題とし、Pを解くプログラムQがあるとする。決して停止しないプログラムOを用意する。さらに、Oに対するPの答えとは違う答えになるプログラムを持ってくる。それをNとしよう。さて、Pを解くプログラムQが存在したとして、それを用いて、停止性判定プログラムHが作れることを示そう。それは次のような二つのステップからなるプログラムである。

第1ステップ

入力X、Wに対し、次のようなプログラムRを（Hの計算の中で）構成する。

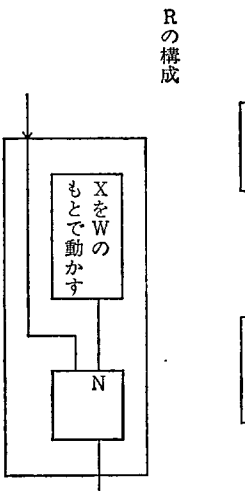
XがWのもとで停止すれば、以後RはNと同じに振舞う。

XがWのもとで停止しなければ、Rは何も出力しない（Oと同じ振舞い）。

第2ステップ

RをQに入力する。

RはX、Wによって、外見上の振舞いはNかOのどちらかのプログラムと等しい。そして、プログラムHはそのどちらかであるかを判定できるのだから、答えによって、QはXがWのもとで停止するか否か判定できることになる。これは前の定理に矛盾し、したがって上記のようなプログラムQは存在しない。　□



したがって、プログラムの計算結果について何等かの性質を判定せよという問題は、いつでも成り立つかいつでも成り立たないかの場合を除いて、計算可能でない。もちろん、停止性判定問題のように半計算可能である場合もある。しかし半計算可能である問題の場合、その否定の問題は半計算可能でさえない。

4 終わりに

以上、この小文では、原理的に計算可能でない問題が存在すること、また数多く存在することをしめした。これは今日、計算の理論と呼ばれる計算機科学の一分野の始まるもととなった研究をコンピュータの言葉を借りて説明したものである。この結果は、実は世の中にコンピュータが存在する以前に、主に数理論理学の分野の人たちによって得られていたことは注意しておく必要がある。その歴史的な経緯や数理論理学における結果について、(筆者の力不足もあって)述べることができなかったのは残念である。

またプログラムに関する判定問題以外にも、計算可能でない問題の例は色々知られている。以下にその様な問題(のうち簡単に説明できそうなもの)をいくつか上げてみよう。

ポストの対応問題

上下二つの文字列からなる対がいくつか与えられる。この対を重複を許して並べて、上の列と下の列が等しくなるようにできるか否か、判定せよという問題。

ヒルベルトの第十問題。

与えられた整数係数方程式が整数解を持つか否か判定せよという問題

群(半群)の語の問題

いくつかの等式で定義される有限生成群(または半群)上で、生成元の積で与えられた二つの元が等しいか否か判定せよという問題。

これらの問題に関する詳しい説明や歴史的な説明をする余裕はないが、この小文を読んで興味を持たれた方には以下に上げるようなより進んだ教科書等を参照していただくことで許してもらいたいと思う。

M・デービス、計算の理論、(渡辺、赤訳、岩波書店、1966)

A・サローマ、計算論とオートマトン理論
(野崎、町田、山崎、横森訳、サイエンス社、1988)

(一橋大学助教授)