サーベイ論文

# チューリングの歴史的位置づけを巡って

河西 棟馬*

## Concerning the historical position of Alan Turing

Toma KAWANISHI

## §1　導入

アラン・チューリング（1912–1954）は数理論理学や暗号理論，人工知能論における業績で知られるイギリスの数学者であるが，近年では計算機科学ないしはコンピュータの父として言及されることも多くなっている．彼の仕事は初期の統計学や概周期関数の研究から，数学基礎論における研究，戦時の暗号解読者としての活動，その後の国立物理学研究所（NPL）における計算機開発，更にはチューリングテストの提唱など初期の人工知能論における仕事や晩年の形態発生学，数値生物学についての研究など多岐に渡っており[1]，彼が残した業績の重要性はすでに広く認められているが，彼をコンピューティングの歴史の中にどう位置づけるべきか，という問題を巡っては近年活発な議論が行われている．

これまでチューリングの歴史的位置付けに関して特に議論の焦点となってきたのは，Turing（1936）において提示された万能チューリング機械の概念といわゆる「プログラム内蔵」計算機との間の関係性であり，本サーベイは特にこの議論を整理することを目的とする[2]．先に結論を述べれば，チューリングが万能チューリング機械の概念を提唱したのは数学基礎論研究の文脈においてであり，一方の電子計算機の製作はあくまで数値計算ないし電子工学の伝統の中で行われたものである．両者は極めて離れた知的伝統の中で行われた活動であって，これらの間になんらかの因果関係を求めよう

* 京都大学大学院文学研究科博士後期課程，日本学術振興会特別研究員（DC）．

[1] チューリングの評伝としては Hodges（[2012]2015）が決定版であり，本論も Hodges のこの研究に多くを負っている．もう一人活発にチューリング研究を行っている人物としてジャック・コープランドがいるが，彼のチューリング評価はヒロイックにすぎるきらいがあり，少々割り引いて読む必要がある．

[2] Haigh（2013）や Daylight（2015）も指摘しているように，「プログラム内蔵」概念そのものが多義的であり議論の混乱を招きがちであるが，便宜上ここでは括弧付きでこの用語を用いる．

とするのは無理がある．むしろ近年では，両者の間には直接的な影響関係はほぼ存在せず，工学的所産としての計算機と理論的計算モデルであるチューリング機械との結び付きが認識されるようになるのは 50 年代以降であることが定説となりつつあり，筆者もこの立場に与するものである[3]．

以下，第 2 章でチューリングの 1936 年の論文「計算可能な数について」（Turing 1936）において定式化された万能チューリング機械について，第 3 章で von Neumann（1945）においてフォン・ノイマンが 1945 年に提示したとされる「プログラム内蔵」概念について検討したのち，第 4 章で両者の間には大きな隔たりが存在することを確認する．

## §2　万能チューリング機械

今日チューリング機械はコンピュータの理論的なモデルとして広く研究に応用されているが，もともとはあくまで数学基礎論研究，特に一階述語論理の決定問題を否定的に解決するための理論的道具立てとして考案されたものである[4]．また，現在一般的に研究に利用されているチューリング機械は 1936 年に提示されたオリジナルのチューリング機械とは定式化の仕方が異なっているため，ここでは 1936 年時点のチューリング機械についてその基本的な特徴を確認したのち，1945 年の「第一草稿」で提示された諸々の発想との比較対照を行う．

### 2.1　チューリング機械導入の文脈

以下で見ていくチューリングの論文『計算可能な数について，その決定問題への応用』（1936）は，研究の伝統としては数学基礎論／数理論理学研究の流れに位置づけられるものである．彼がこの計算モデルを考案したのは「機械的 mechanical（ないし実効的 effective ）手続き」とは何かを考察する過程においてであった[5]．少々乱暴な言い

[3] cf. Mounier-Kuhn 2012; Daylight 2012; Haigh 2014．これについては別稿にて扱うこととしたい．

[4] 本論では数学基礎論史には立ち入らず，1936 年に提示されたチューリング機械がどのようなものであったかを説明するに留める．彼の仕事が行われた歴史的文脈の詳細については Adams（2011）や Petzold（[2008]2012），Priestley（2011, ch. 4），佐野・杉本（2014, ch. 2）などが参考になる．なお，「チューリング機械」という命名を与えたのはアロンゾ・チャーチであって，チューリング自身は Turing（1936）においてこの計算モデルを単に機械 machine ないし計算機械 computing machine としか呼んでいない．しかしここでは物理的な計算機械との混同を避けるために便宜上「チューリング機械」の呼称を用いる．

[5] 佐野・杉本（2014, pp. 62–63），Copeland（2015）によれば，手続き $M$ が実効的ないし機械的と呼ばれるのは次のような場合である．

方をすれば，1936 年以前の研究において形式系によって定義された基本操作が実効性 effectiveness という非形式的概念に当てはまるかどうかは，どの程度「そう感じられるかどうか」に依存していた[6]．そこで，1936 年にエミール・ポストとチューリングの二人は独立に実効的計算可能性概念の分析を行い，ポストの言い方を借りればより大きな「心理的迫真性 psychological fidelity」[7]をこの概念に与えることを目指した．

ポストとチューリングはどちらも自らの計算モデルを「特定の知的問題を機械的な方法で実行する人間」という比喩を検討することによって基礎づけようとした．ポストは計算を実行する存在を「問題解決者 problem solver」及び「作業者 worker」と呼び，チューリングは 1936 年論文の第 9 節で「計算者 computer 」[8]について分析を行っている．ポストとチューリングは共に人間の行う計算という活動から二つの特徴を抽出している．一つは計算に使われるデータが保存される外部媒体，もう一つは計算者ないし作業者が従うところの命令である．

Post（1936）の提示した計算モデルは以下のようなものである．計算作業は「2 方向に無限に続く箱ないしスペースの列 a two way infinite sequence of spaces or boxes」として表現される「記号空間 symbolic space」において，作業者により実行される[9]．記号空間を構成する箱はそれぞれが印（ポストの原論文では垂直線 vertical stroke が使われる）の有無によって二つの状態を持ち，作業者は一つの時点においては一つの箱に対してのみ操作を行うことができる．特定の箱を出発点として選び，有限数の箱に印を付けてから作業者に指示を与え，作業者が作業を完了すると，作業者による操作列の実行結果が印のついた箱の配置によって与えられる．作業者に可能な操作は以下の 5 つのみである．

(a) 箱が空の場合に印を付ける
(b) 箱に印がついている場合にその箱の印を消す

1. $M$ は有限個の精確な命令（有限個の記号で表現される）で提示されている
2. $M$ は間違いなく実行される，有限ステップで求められた結果を生じる
3. $M$ は原理的には紙と鉛筆以外の如何なる道具も用いない人間によって実行できる
4. $M$ は人間がそれを実行するのに何ら洞察も工夫も必要としない

ただし，これが数学的な定義ではない点には注意が必要である．ここでいう「洞察や工夫」は数学的な定式化をされず，日常的な意味で理解されている．

[6] Priestley 2011, p. 75.

[7] Post 1936, p. 105.

[8] 1936 年時点で “computer” と言えば「計算を行う人間」の意味であったことに注意されたい．

[9] Post 1936, p. 103.

(c) 右側の箱に移る
(d) 左側の箱に移る
(e) 箱が印付けられているか否かを判断する

問題の解法は自然数で番号が振られた指示 direction の有限列によって与えられる．指示には次の 3 種類があり，$j_i$ を $i$ 番目の指示とすれば

(A) 操作 (a)(b)(c)(d) のいずれかを実行し，次に指示 $j_i$ に従え
(B) 操作 (e) を実行してその答えに応じて指示 $j_i'$ ないし $j_i''$ に従え
(C) 停止せよ

のいずれかの形式を取る．このように．指示は「作業者が行う操作」と「次に従う指示の番号」の組によって構成されている．なお，Post（1936）は 3 ページ程度の短い論文であり，この計算モデルが「ゲーデル-チャーチの意味での再帰性と論理的に等価である」(p. 105) ことを予想はしているが証明は与えていない（しかしこれは後に事実であることが証明された）．

データを記録するための媒体として，チューリングはポストの「記号空間」と類似した 1 次元の無限長のテープを採用し，計算を行う主体をはっきりと「機械 machine」と呼んだ．一階述語論理の決定問題に対する否定的解決に関してはアロンゾ・チャーチの方が成果の公表は早く，それゆえチューリングの仕事の意義は「機械的手続き」を明確に捉える手段をもたらした点にあるということもできる．

## 2.2 チューリング機械の概要

ではチューリング機械の概要の紹介に移ろう[10]．チューリング機械は人間の計算という行為を抽象化することから得られた架空の計算機械である．チューリング機械は一つ一つのマス目が区切られた 1 次元・無限長のテープ上で動作する．一つ一つのマス目のそれぞれには一つだけ記号を書き込む事ができ，また記号の種類は有限であることが前提とされる[11]．機械は $m$ 配置 $m$-configuration と呼ばれる内部状態をもつ[12]．

[10] 以下の記述は基本的に Turing（1936, pp.231–232）および Petzold（[2008]2012, pp. 107–113），Priestley（2011, ch. 4）を参照している．チューリングがこの着想を得た経緯については Hodges（[2012]2015, 上巻 pp.167–188.）を参照されたい．

[11] Turing 1936, pp. 231, 250.

[12] 第 9 節の記述から，これは計算者の「心の状態 state of mind」をモデル化したものであることが分かる．（Turing 1936, pp. 250–251）

機械は任意の一時点において，*m* 配置の有限集合のうちの一つの状態をその時点の状態として持つ．また，どの時点においても「機械の中」にはテープのマス目が一つだけ存在し，このマス目は「走査されたマス目 scanned square」と呼ばれ，このマス目に書かれた記号は「走査された記号 scanned symbol」と呼ばれる．機械が直接認識することができるのはこの「走査された記号」だけである．つまりこの機械は一度にテープ全体を見渡すことはできず，一度に一つのマス目に注目することしかできない[13]．機械はテープ上の記号を走査すると *m* 配置を変更することができ，これにより記号の幾つかを実質的に記憶することができる．各時点で機械に可能な動きは *m* 配置とその時点で走査されている記号の組み合わせによって決まり，この組み合わせは「配置 configuration」と呼ばれる．機械はその時々の配置に応じて，走査されたマス目に新しい記号を書き込んだり，走査された記号を消去したりする．走査されるマス目は右か左に 1 マスずつしか移動しない．

チューリング機械はその振る舞いによって無数に存在し，正しく定義された一つのチューリング機械は一つの実数に対応している．原論文でのチューリングの関心は計算可能数の定義にあり，個々のチューリング機械はテープ上で動作を開始すると一つの実数（厳密に言うと 2 進実数の小数点以下）を永遠に計算しつづける．チューリングはチューリング機械の動作を記述するために，機械動作表 machine tables という記法を使った．例えば空白のテープを受け取って，このテープ上に無限に “01010101...” と印字し続けるチューリング機械は次のように記述される[14]．

| 配置 | | 動作 | |
|---|---|---|---|
| *m*-配置 | 記号 | 操作 | 最終 *m* 配置 |
| 𝔟 | None | P0, R | 𝔠 |
| 𝔠 | None | R | 𝔢 |
| 𝔢 | None | P1, R | 𝔣 |
| 𝔣 | None | R | 𝔟 |

一行目は「m-配置が 𝔟 で走査したマス目が空白ならば 0 を印字（P0）してヘッドを

[13] Petzold [2008]2012, p. 110．この点についてもチューリング自身が原論文第 9 節で解説を加えている．つまり「人は 999999999999999 と 999999999999999 が同じ数字だとはすぐには判別できない」ため，「計算者がある瞬間に認識できる記号の数ないしマス目の数にはある上限 *B* があると仮定する．彼がより多くを認識したいのなら，彼は認識を何度も行う必要がある」(Turing 1936, p. 250.).

[14] Turing 1936, p. 233．なお，これは二進法表記の $\frac{1}{3}$ を計算しているものとして解釈される．“None” は空白のマス目を，“P0”, “P1” の P は Print（書き込み）を表す．

右（R）に移動し，機械の m-配置を $\mathfrak{c}$ に変える」と読める．以下は同様である．ここに挙げたのは単純な例であるが，チューリングは骨組み表 skeleton tables という略記法を用いてチューリング機械の記述を簡略化することにより万能チューリング機械の記述を与えている．チューリングはテープ利用上の規約として，数字を印字するマス目（F マス，F-squares）とそれ以外の補助的な記号を印字するためのマス目（E マス，E-squares）が交互に 1 マスずつ互い違いになるように機械の記述を与えている．そのため，この記述に従って機械を走らせた場合にテープに書き込まれるのは 0 と 1 の間に空白のマス目が挟まった記号列になる[15]．

## 2.3 万能チューリング機械の概要

万能チューリング機械 $\mathscr{U}$ の概説に移ろう．$\mathscr{U}$ もまたチューリング機械の一つであるが，$\mathscr{U}$ は，他のチューリング機械 $\mathscr{T}$ の機械動作表（を $\mathscr{U}$ が読み込めるように変換したもの）を受け取ると，$\mathscr{T}$ が計算するのと同じ列を計算する．つまり $\mathscr{U}$ は単一の機械でありながら，適切に定義された任意のチューリング機械が行う計算を実行することができる．万能 universal と呼ばれるのはそのためである．

この際に，$\mathscr{U}$ は $\mathscr{T}$ が計算の過程で通過するあらゆる配置をすべてテープ上に記録する．$\mathscr{U}$ は $\mathscr{T}$ が（$\mathscr{T}$ のテープ上に）書き込むであろう記号を $\mathscr{U}$ のテープ上に正確に書き込むことによって，いわば $\mathscr{T}$ の動作をエミュレートする[16]．$\mathscr{U}$ はそれゆえ「任意の計算可能列を計算することの出来る機械」ということができ，$\mathscr{U}$ の存在はまた，「明示的に与えられた命令に従う」というプロセスそれ自体が機械的（実効的）手続きになっているということを示している[17]．

万能チューリング機械の記述それ自体も上に挙げたような一行一行の記述によって

[15] この表記ですべての行を書き下すと行数が大きくなるため，チューリングは一行で複数の操作列を記述することや，同じ $m$ 配置を共有する行をまとめて記述することを許容している（Turing 1936, pp. 234–235）．ただしこのような拡張をほどこしても，そのようにして記述された機械動作表は常に上記の最も単純な形に書き下すことが可能であるため，これは単なる表記上の簡略化にすぎない．チューリングは他にもテープ使用上の規約や機械動作表における変数や関数，再帰の使用など，様々なテクニックを導入しているが，紙幅の都合上これらについては省略する．

[16] ただし $\mathscr{U}$ によって実行される計算の内容は $\mathscr{T}$ を単純に実行した場合の計算の内容とは異なり，$\mathscr{U}$ は機械 $\mathscr{M}$ が計算する列以外にも多くの余分な出力を印字する．

[17] 興味深い事に，チューリングは通常のチューリング機械の動作を機械動作表によって与える際にはこれを機械の「記述」と呼ぶのに対し，以下の変換を通じて得られた（そして万能機械 $\mathscr{U}$ のテープ上に書き込まれる）記号列のことを「（万能機械への）命令 instruction」と呼んでいる，つまり一つの計算作業のみを行う（専用）チューリング機械は，以上の変換を通じて万能チューリング機械への命令となる．

構成されるが，万能チューリング機械は空白のテープではなく，個別のチューリング機械の記述を変換したものが書き込まれたテープを受取り，その上で動作（与えられたチューリング機械のエミュレート）を行う．個々のチューリング機械は次のような手順で自然数へと変換される[18]．

1. $q_1, q_2, \ldots, q_i, \ldots, q_R$ （$i \in [1, R]$）のようにして，m 配置に 1 から始まる自然数の番号を割り当てる．
2. $S_1, S_2, \ldots, S_j, \ldots, S_m$ （$j \in [1, m]$）のようにして，記号にも 1 から始まる自然数の番号を割り当てる．このとき特に空白には $S_0$，0 には $S_1$，1 には $S_2$ を割り当てる．

このとき，書き込みを行わずただヘッドを移動する場合には元々書かれていた文字を再度書き込むものとし（つまり，どの行においても何らかの書き込みが行われるものとし），ヘッドが移動しない場合を記号 $N$ で表わすこととすれば，表の各行は “$q_i S_j S_k L\ q_m$”, “$q_i S_j S_k R\ q_m$”, “$q_i S_j S_k N\ q_m$” のいずれかの形式に書き直すことが出来る．各業の区切りを “ ; ” で表わすこととすれば，先に挙げた 0 と 1 を交互に印字し続ける機械は

$$q_1 S_0 S_1 R q_2\ ;\ q_2 S_0 S_0 R q_3\ ;\ q_3 S_0 S_2 R q_4\ ;\ q_4 S_0 S_0 R q_1$$

という文字列へと変換される．次に

3. $q_i$ を文字 “*D*” の後ろに文字 “A” を $i$ 回繰り返したものに置き換え， $S_j$ を “D” の後ろに “C” を $j$ 回繰り返したものに置き換える．

すると先程の機械は

$$DADDCRDAA\ ;\ DAADDRDAAA\ ;\ DAAADDCCRDAAAA\ ;\ DAAAADDRDA$$

と書き直すことが出来る．こうして得られた機械の記述を Turing（1936）は「標準記述 standard description」 と呼んでいる[19]．

---

[18] 以下の手順は Turing（1936, pp. 239–241）に基づく．

[19] チューリングはさらに，この標準記述について．“A” を “1” に，“C” を “2” に，“D” を “3” に，“L” を “4” に，“R” を “5” に，“N” を “6” に，“;” を “7” に置き換えるという操作を行うことによって，機械の記述を巨大なアラビア数字へと変換するという操作を行っている．こうして得られた数は記述数 description number と呼ばれ，その一つ一つがもとの機械が計算する計算可能数と対応している．記述数は単なる整数であり全候補を数え上げることが出来るため，すべての計算可能数が原理的には列挙することができることになる．チューリングはこの操作により，計算可能数（チューリング機械に

チューリングは原論文第 6–7 節において万能機械 $\mathscr{U}$ の完全な機械動作表を与えているが，この動作表は（再帰の利用や変数の多用など）多くの技巧を凝らした大変複雑なものであり，ここでは詳細に立ち入ることはしない[20]．

簡単に概要だけを説明すると，$\mathscr{T}$ が行う計算はその機械動作表とテープ上の記号列，走査されているマス目の位置によって決定される．テープ上の記号列と走査されているマス目位置は計算のステップが進む毎に変化していく．$\mathscr{U}$ はそれゆえ $\mathscr{T}$ のテープの完全な内容を $\mathscr{U}$ のテープ上に記録して置かなければならない．それゆえに $\mathscr{U}$ のテープは 3 つの部分に分かれる．つまり，

(1) $\mathscr{T}$ の標準記述（$\mathscr{U}$ への命令）が書き込まれた部分[21]
(2) $\mathscr{T}$ のテープの完全な内容[22]を記録した部分
(3) $\mathscr{T}$ が計算した記号を印字する部分[23]

の 3 つである．万能チューリング機械は現代的に言えばインタプリタにあたり，(1) 上に与えられた標準記述と (2) 上に印字された $\mathscr{T}$ の完全記述とを参照しながら (3) 上に記号を印字し，その操作によって生じた次の $\mathscr{T}$ の完全配置がコロンを挟んで右側に記録される．要点は $\mathscr{U}$ のテープ上に $\mathscr{T}$ のテープが表現される，という点である．

以上がチューリングが 1936 年に提示した万能機械の概要である．先に進む前に諸点確認をしておきたい．第一に，以上の「機械」はあくまでも抽象的な数学的対象であり，チューリングはこの論文では物理的な計算機械の機構には一切の言及を行っていない[24]．さらに，主にテープすなわちメモリが無限の容量を持つという問題により，チューリング機械の物理的実装は不可能である．第二に，この研究が行われたのは数学基礎論研究の文脈においてである．議論の都合で省略したが，万能チューリング機

よって生成することの出来る数）が実数全体の可算部分集合にすぎないこと，それゆえ計算可能でない実数が多数存在することを示している．

[20] それゆえ原論文のこの部分にはいくつか誤りがあることが知られている．詳細については Petzold（[2008]2012, ch. 9）を参照されたい．

[21] テープ使用上の規約として，この部分はテープ使用域の左端に置かれることになっている．この部分と残りの部分は特殊記号 “;;” によって区切られる．

[22] これは完全記述 complete configuration と呼ばれる．完全記述には，その時点のテープ上の記号全体と，テープヘッドの位置とが表現される．この部分も標準記述の表記法によって符号化されたものが用いられる．

[23] (2) と (3) の領域はコロン “:” で区切られる．(2) と (3) の領域は $\mathscr{U}$ のテープ上に交互に出現する．

[24] レトロスペクティブな言い方を許容するならば，ここで言及されているのは徹頭徹尾ソフトウェアないしプログラムに関わる側面であって，これを実行する物理的な計算機ハードウェアについてではない．

械はあくまで「決定問題」の否定的解決のための準備として導入されたのであって，1936 年論文の主眼はむしろ後半部，つまり「機械の標準記述が与えられたときにその機械が 0 を印字することがあるか」を決定する機械が存在しないことを証明している第 8 節と，この問題が一階述語論理の決定問題と同値であることを証明している第 11 節にある．こうした点については再度第 4 章で検討することとして，「プログラム内蔵」計算機の検討に移ることとしたい．

## §3 「プログラム内蔵」計算機と「第一草稿」

今日，コンピュータをチューリングの万能機械概念の物理的実現とみなすことはほとんど日常的に行われている．例えばチューリング研究者として知られるジャック・コープランドはこう述べている．

> コンピュータ科学の教科書や書籍や記事がその歴史について書くとき，チューリングのプログラム内蔵方式をフォン・ノイマンが考えたものとしている．確かにフォン・ノイマンはアメリカの科学者の中でも目立った存在で，この考えを世に広めたが，それが自分のアイデアだと主張したことは決してなかった[25]．

このような，万能チューリング機械といわゆる「プログラム内蔵」コンピュータを同一視し，チューリングこそがコンピュータの父であるとする言及はそれほど珍しくはない．もう少し穏当な立場として「第一世代の計算機開発の現場において，論理学と工学が合流し，その結果現代的なコンピュータが登場した」とする歴史記述も一般的に行われているように思われる．例えば数学史家マイケル・マホーニーは次のようなコメントを残している．

> 二つの全く別の歴史的系統，すなわち高速かつ高精度な自動計算を達成しようという努力と，推論を行うことができる論理学的機械を設計する努力とが合流したのは，まさしくフォン・ノイマンと ENIAC 開発チームとの協働においてであった[26]．

[25] Copeland [2012]2013, p. 175.

[26] Mahoney and Haigh 2011, p. 26. 但しこの論集はマホーニーの死後に編者の Haigh がまとめたものであり，この引用の初出は 1988 年の文書である．

「プログラム内蔵」という工学的概念の導入に際して，チューリングの万能機械概念が大きな役割を果たしたとする見方は Hodges にも見られる[27]．

確かに，万能チューリング機械概念といわゆる「プログラム内蔵」計算機の間には類似性があるように思われる．つまり，どちらも一つの記憶領域の中にデータと命令を同居させることによって高い汎用性を獲得している．しかし，1945 年頃に「プログラム内蔵計算機」について記述した文書を読む限り，この発想が導入されたのは工学的な応用と数値計算の実践の文脈においてであり，抽象的な論理学への言及はほとんど存在しない．以下この点を確認するために，まずこの発想が提示された文脈について概説し，次に「プログラム内蔵」という発想を広める上で大きな役割を果たした文書「EDVAC に関する報告書第一草稿」（von Neumann 1945, 以下「第一草稿」と略）を主に参照しつつ，構想当時の「プログラム内蔵」計算機について，その内容を検討していく．

## 3.1 EDVAC 開発の経緯

EDVAC はペンシルヴァニア大学ムーアスクールにおいて 1943–46 年に開発が行われた電子計算機 ENIAC の後継機として開発が計画されていた計算機である．このグループは ENIAC の設計に幾つか難点があることを早い時期から認識しており，1944 年頃から後継機 EDVAC の開発プロジェクトを開始した．それゆえ，EDVAC について述べる前に ENIAC について少々触れておく必要がある．

ENIAC は弾道計算をその主な用途として開発された[28]．ENIAC はプログラム可能ではあったが機械の配線を差し替えることによってプログラミングが行われたため，プログラミングの作業は極めて厄介であった．しかし ENIAC は長期に渡って同じ問題を解くことを意図して開発されたものであり，その利用が意図されていた文脈においてこの問題は許容可能なものであった[29]．とはいえプログラムの困難さはやはり問題であり，後継機の目標の一つは電子計算機のプログラミングに最適な方法を考案することとなる．

1940 年代初頭において，計算のプログラム制御という発想はすでに実用化が始まっていたが，これらはすべて機械式ないし継電器を利用した電気機械式であった．プロ

[27] Hodges（[2012]2015, 下巻 pp. 91–95）を参照．

[28] ENIAC 開発の経緯については既に多くの記述があるが，例えば Campbell-Kelly et al.(2014, ch. 4) 等を参照されたい．

[29] Priestley 2011, p. 126.

グラム制御計算機にはパンチカードや紙テープによって命令を供給するという方法が一般的であったが，電子計算機において従来の「外部媒体から命令を読み取って実行する」という方法は実践的ではなかった．プログラムをカードないしテープから外部供給する場合，物理的な読み込み動作がネックとなってせっかくの演算の高速性を活かすことができなくなるからである[30]．

この問題への対処として提案されたのは，プログラムを保持するためのストレージを計算機の内部機構へと組み込む，という方策であった．ENIAC 開発においてエンジニア・チームを率いたプレスパー・エッカートは 1944 年 1 月に記述したタイプスクリプトのなかで，計画中の計算機械の貯蔵部について次のように言及している．

まず型 A と呼ばれる円盤ないしドラムがあり，これは少なくとも表面が「繰り返し高速に磁化・消磁することができる」磁気合金で出来ている．そして「この磁化される部分の位置および／または位相が，なんらかの便利なコードにより，あとで使用・参照される，文字あるいは記号を貯蔵する方法をもたらす」．これをエッカートは「一時的なタイプ temporary type」の円盤ないしドラムと呼んでいる，次に型 B と呼ばれる同じく円盤ないしドラムがあり，これは「円盤近くに置かれたコイルにおいて電圧を生じさせるように表面に刻みが入れられ」ており，これは「上で言及した合金円盤によってもたらされるよりも永続的な，必要とされる基本信号の貯蔵手段をもたらす」．こちらをエッカートは「永続的なタイプ permanent type」と呼んでいる．さらに彼は「一時的なタイプ」の合金円盤は数を無制限に貯蔵することを可能にし，かつ「複合的なシャフト系を用いれば，利用可能な機械設備において，そして設備とこれに関わるプロセスの自動的なプログラミング programming に関して，顕著な向上がなされるかもしれない」と述べたうえで，「電気的な制御は高速な演算を可能にするだけでなく，設計の単純化とより容易な拡張およびほかの器具との接続を可能にする」としている[31]．この文書は 3 ページ程度と短く，またその内容もわかりやすく書かれているわけではないが，少なくともここから読み取れるのは，エッカートは 1944 年 1 月の段階で，計算を制御する機構を計算機の内部に組み込むことにより「電気的な制御による高速な演算」を行おうという発想を持っていた，ということである．

---

[30] この点は Ceruzzi(2003, p. 22) が既に指摘している．

[31] Eckert 1944. この資料はコンピュータ歴史博物館の Web アーカイブに公開されており “Disclosure of Magnetic Calculating Machine” という題がついている．(http://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-8-pdf/k-8-u2775-Disclosure.pdf, 2018 年 2 月 28 日閲覧)

このように 1944 年までの段階で，ENIAC の後継機に関する検討はすでにある程度進んでいた．フォン・ノイマンが登場するのは，既にある程度 ENIAC の後継機についての議論が進んでからのことである．彼は 1944 年夏の終わり頃からこのグループと関わりを持ち始め，翌 1945 年 6 月に「第一草稿」を公表する[32]．

## 3.2 「第一草稿」の概要

では「第一草稿」の内容に移ろう．「第一草稿」はペンシルヴァニア大学ムーア・スクールのチームが開発計画を進めていた計算機 EDVAC について，フォン・ノイマンがそのアイデアを整理・公表したものである[33]．「第一草稿」はコンピューティング史研究において最も重要視されてきた文献の一つであり，すでに多数の解説が存在する．以下では適宜それらを参照しつつ，「第一草稿」に描かれた EDVAC について見ていくこととしたい．

「第一草稿」は全 15 節の構成で，ここに盛り込まれている発想は数多いが，Haigh（2013）および Haigh et al.（2014）によればこれらは大きく以下の 3 つに分類することができる[34]．

(1) プログラミングに関わるもの

---

[32] 「第一草稿」の表紙には 1945 年 6 月 30 日という日付がある．フォン・ノイマンと ENIAC 及び EDVAC 開発との関わりについては Goldstine（[1972]1980, ch. 7）や Aspray（[1990]1995, ch. 2–3）に詳しい．フォン・ノイマンは 1930 年代に純粋数学ないし数理物理学の世界から応用数学の世界，より工学的な世界へと身を転じ，第 2 次大戦勃発以降は合衆国政府の科学コンサルタントとしてマンハッタン計画を始めとする国家プロジェクトで中心的な役割を果たすことになる．彼はこの仕事の過程でしばしば（特に流体力学の問題を解く際に発生する膨大な計算量という）問題に直面し，その過程で自動計算技術に関心を抱いた．1943 年から 44 年の時期には当時の計算技術の状況に関して包括的な調査を行っており，フォン・ノイマンが ENIAC を知ったのもその過程においてである．

[33] ここで参照したテクストはスミソニアン・ライブラリが Web 上で公開しているもの（`http://archive.org/stream/firstdraftofrepo00vonn#page/n0/mode/2up`, 2018 年 2 月 28 日閲覧）である．「第一草稿」については，フォン・ノイマンが単独の著者となっていたことや他の開発者（プレスパー・エッカートやジョン・モークリー等）の了承を得ずにそのアイデアを公表したことによって，後に特許権・先取権を巡って激しい議論や人間関係の軋轢をもたらした事がよく知られている．実際，ここに提示されている発想をフォン・ノイマン一人に帰することは決して出来ない．これについては例えば Ceruzzi（2003, pp. 21–23.）を参照されたい．

[34] ヘイグは「第一草稿」をその後の計算機開発においてパラダイムの役割を果たした研究であるとし，クーンを参照しつつ，これらのそれぞれを (1) モダン・コード・パラダイム，(2) フォン・ノイマン・アーキテクチャ・パラダイム，(3) EDVAC ハードウェア・パラダイムと呼んでいる．しかしここで彼はパラダイム概念の多義性をも引継いでしまっており（例えば「パラダイム」を「模範例」の意味で用いているのか「その共同体において支配的な役割を果たす概念的枠組み」といった意味で用いているのか明瞭でない），問題が多い．それゆえここでは元々の用語法は踏襲せず，彼が導入した分類法のみを借用した．

(2) コンピュータ・アーキテクチャに関わるもの
(3) ハードウェアの物理的構成に関わるもの

本論において関心があるのは万能チューリング機械と「プログラム内蔵」計算機の関係性であるので，(3) については解説を省略し，(1)(2) の部分に話を絞る．議論の都合上，先に (2) のアーキテクチャについて紹介し，その上で (1) のプログラミングに関わる側面を見ていくことにする[35]．

コンピュータ・アーキテクチャについて解説しているのは「第一草稿」第 2 節の部分であり，「第一草稿」に記載されている計算機の基本構成は以下の通りである．

CA　中央算術部 central arithmetical part
CC　中央制御器官 central control organ
M　記憶 memory
R　外部記録媒体 outside recording medium
I　入力 input
O　出力 output

これら諸部分はそれぞれに独立した機能を有している．四則演算を始めとする計算を行うのは CA のみであり，計算は逐次的に行われる（並列計算は行わない）．CC は装置の「論理制御」を行う．論理制御とは「演算を適切に配列する」ことと説明される[36]．フォン・ノイマンによれば「もし装置が柔軟 elastic であるなら，つまり可能な限り汎用 all purpose に近いものであるならば，与えられて特定の問題を定義する個別的 specific な命令と，命令が—その命令が何であれ—実行されるのを監視する一般的な制御器官の間に区別が設けられねばならない．前者はなんらかの仕方で貯蔵 store されなければならず［中略］，後者は明確な装置の部分によって表現される．CC で意味するのは後者の機能だけであり，この機能を果たす器官が第二の部分 CC を形成する．」[37]

次に記憶 M について概説しよう．M はプログラム内蔵概念との関わりにおいて特

[35] Priestley（2011）も指摘している通り，「第一草稿」は，生物や人間の頭脳の比喩を全面的に用いて計算機について述べたサイバネティクス的色彩の強い文書である．ここで計算機開発とサイバネティクスの関係に立ち入る余裕はないが，計算機の諸部分を器官 Organ と呼んだり，ストレージを記憶 Memory と呼ぶのは当時それほど一般的なことではなかったことは注意しておく必要がある．

[36] von Neumann 1945, 2.3 節.

[37] *ibid*.

に重要である．フォン・ノイマンはメモリの用途を次のように解説している．

(a) 演算の過程で生じる中間結果の記憶
(b) 命令の内部への貯蔵
(c) 関数表の記憶内への貯蔵
(d) 偏微分方程式の初期条件と境界条件の記憶
(e) 双曲型ないし放物型の偏微分方程式で変数 $t$ で積分されるものについて，周期 $t$ に属する中間結果を $t + dt$ の計算中に記憶
(f) 全微分方程式における (d)(e) のような条件，中間結果の記憶
(g) 逐次近似法 succesive approximations によって解かれる問題において，各近似の中間結果の記憶．
(h) ソート問題やいくつかの統計的実験 certain statistical experiments において扱われる素材の記憶

上に挙げた (b) の項目及び次の引用が，この文書がいわゆる「プログラム内蔵」概念を提示したことの根拠とされる．

> この装置は相当の大きさの記憶を必要とする．記憶の各部分はその性質においては幾分，その目的においては大いに異なる機能を果たさねばならないように思われるが，それでもなお記憶全体を一つの器官として扱い，それらの諸部分を上に列挙した様々な機能に対して可能な限り交換可能なものとすることは魅力的である．

こうした記述がもつ含意についてはあとで再度検討するが，この概念が極めて実践的ないし工学的な文脈の中で導入されていることは確認しておきたい．

「第一草稿」に描かれた計算機は今日的に言えばランダムアクセスメモリである．一つ一つの水銀遅延線には多数の小サイクル minor cycle が記憶されている [38]．個々の小サイクルは 32 のユニット（ビットに相当する）から成り，一つの小サイクルは単一の数ないし命令を保持することが出来る．個々の小サイクルは特定の一つの水銀遅延線を識別するための数 $\mu$（これは大サイクル major cycle と呼ばれる）と，その遅延線の中の個々の小サイクルを指定する数 $\rho$ という二つの数によって指定することができ

[38] この記述は von Neumann（1945, 12 節）及び Godfrey and Hendry（1993）を参照している．記憶は複数の水銀遅延線を使って構成されることが想定されており，それゆえ記憶へのアクセス方法も遅延線の特性に合わせた方法になっている．

る．小サイクルは 32 のユニットから成るが，その小サイクルが命令であるか数であるかはサイクルを構成する最初のユニットである $i_0$ の値によって識別される．$i_0 = 0$ ならばその小サイクルは数を表し，$i_0 = 1$ ならば命令を表わす[39]．

CA は 3 つのレジスタ $I_{CA}$, $J_{CA}$ 及び $O_{CA}$ を備えており，そのそれぞれが一つの小サイクルを保持できる．$I_{CA}$ と $J_{CA}$ の 2 つは被演算子を保持するのに，$O_{CA}$ は計算結果を保持するのに用いられる．算術演算を行う際には，その引数が $I_{CA}$ と $J_{CA}$ へと転送され，計算結果が $O_{CA}$ に出力される．CA に対し数を転送するとき，M から CA に送られる数はすべて $I_{CA}$ に送られ，もともと $I_{CA}$ にあった数は $J_{CA}$ に移動される．この際もともと $J_{CA}$ にあった数は消去される．

CA が実行できる操作は (1) 加減乗除及び平方根計算，(2)$I_{CA}$ と $J_{CA}$ から $O_{CA}$ へとデータを移動させる操作 $i$ および $j$，(3)$O_{CA}$ にあった数が 0 より大きいか小さいかに応じて $I_{CA}$ か $J_{CA}$ の内容を $O_{CA}$ に移す操作 $s$，(4) 入出力用に二進十進変換を行う操作（以下の表では省略）の 4 種類がある．演算命令は次の表にまとめられる．

| 省略記号 | 意味 |
|---|---|
| $wh$ | 操作 $w$（加減乗除および平方根計算，操作 $i, j, s$ のいずれか）を実行，$O_{CA}$ に結果を保持 |
| $w \to \mu\rho$ | 操作 $w$ を実行，結果を小サイクル $\mu\rho$ へ転送 |
| $wh \to \mu\rho$ | 操作 $w$ を実行，結果を $O_{CA}$ に保持するとともに小サイクル $\mu\rho$ へ転送 |
| $w \to f$ | 操作 $w$ を実行，結果をこの命令の次の小サイクルへ転送 |
| $wh \to f$ | 操作 $w$ を実行，結果を $O_{CA}$ に保持するとともに次の小サイクルへ転送 |
| $w \to A$ | 操作 $w$ を実行，結果を $I_{CA}$ に転送 |
| $wh \to A$ | 操作 $w$ を実行，結果を $I_{CA}$ および $O_{CA}$ に転送 |
| $A \leftarrow \rho\mu$ | 小サイクル $\rho\mu$ に保持されている数を $I_{CA}$ へ転送 |
| $C \leftarrow \rho\mu$ | CC を小サイクル $\rho\mu$ と接続 |

これらの操作により，M と CA の間でデータのやり取りが可能になり，CA 上で行われる演算が指定される．

プログラムがデータと同じ記憶の中に保持されているという性質により，これらのコードは命令修飾，つまり命令を計算の対象として扱い，命令の内容を書き換えるという操作が可能である．しかし，「第一草稿」はアドレス（$\rho\mu$）を指定する 13 桁にし

[39] von Neumann 1945, 15.1 節．残りの 31 ビットで数ないし命令の内容が示される．数表現は 2 進固定小数点方式である．詳細は Godfrey and Hendry（1993）等を参照．

かこの操作を許容していない．$O_{CA}$ から数を M の小サイクルへと転送するかどうかは，そのとき CC が扱っている小サイクルが数であるか命令であるかに依存している．もしその小サイクルが数を保持しているならばその全体が $O_{CA}$ の内容によって置き換えられるが，もし命令を保持しているならば，その小サイクルの内アドレスを指定している最後の 13 桁のみがコピーされる．

基本的に CC は命令を保持している小サイクルを記憶上の並べられた順番に逐次的に実行していく．制御の転送，つまりジャンプ命令は $C \leftarrow \rho\mu$ によって CC の読み取り位置を変更することにより行われる[40]．これだけでは無条件ジャンプ命令しか可能でないが，操作 $s$ とアドレス修飾の仕組みにより，条件ジャンプ命令すなわち条件分岐が可能になる．例えば特定の数が 0 以上か否かによって命令 $c_1$ ないし $c_2$ に制御を転送したい場合を考える．その場合，まずテストする数を計算して結果を $O_{CA}$ に保持し，次に $c_1$ および $c_2$ のアドレスを CA へと転送する．その上で操作 $s$ を実行すれば，$O_{CA}$ の符号により，$O_{CA}$ は $c_1$ あるいは $c_2$ で置き換えられる．あとは新しい $O_{CA}$ の内容（転送先アドレス）を C へと転送すればよい．これにより，条件分岐が実現されることになる．

## §4 検討

以上，万能チューリング機械と EDVAC 草稿に書かれた機械について，特にプログラミングに関わる側面に注目して概説してきた．チューリングが 1936 年に提示したアイデアを再度整理すれば次のようになるだろう．チューリング機械は無限長のテープ上で動作し，有限の状態を持ち，有限個の記号を扱うことができる．その時々の状態とテープヘッドが走査している記号によって，テープへの書き込み動作と次の状態は定まる．行う事の出来る操作はテープへの 0,1 の書き込みとヘッドの左右への移動のみである．計算機械の記述は 2.1 節で紹介した手続きによって万能チューリング機械に対する命令列（標準記述）へと変換され，万能チューリング機械は与えられた標準記述を読み取ってその動作をエミュレートすることができる．

一方の「第一草稿」に記述された計算機は，演算部，制御部，記憶，そして入出力を備えており，記憶内に保持された命令を制御部が読み出して演算を制御することにより計算が進行する．チューリング機械との対照でいえば，EDVAC の記憶はテープになぞらえることができる．記憶上には命令とデータが同居し，また命令のアドレス

[40] C はプログラムカウンタに相当する．

部を演算の対象とすることにより条件分岐を用いた柔軟な処理を行うことができる．確かに，両者の間には類似性が認められるように思われるが，概念的モデルとしてのチューリング機械と現実の物理的機械の設計との間には多くの相違がある．

まず，チューリング機械のテープはランダムアクセスメモリではない．チューリング機械のモデルにおいても，何らかの記号を用いてテープに印を付けることにより，印付けられた位置へとテープヘッドを移動するような機械動作表を与えることは出来るが，テープ上に番地のようなものが存在しているわけではない．次に，計算処理の単位も異なる．「第一草稿」の計算機に用意されている命令は四則演算および平方根の計算のみであり，論理和や論理積，ビット反転のような論理演算は命令としては組み込まれていない．あくまで計算は小サイクル，つまり 32 ビットで表現された一つの数を対象として行われるものである．他方チューリング機械はテープ上のマス目一つに書かれる数，つまり 1 ビットを個々の操作の対象としている．更に，チューリングの 1936 年論文で提示されたのは抽象的計算モデルであって，この概念モデルに物理的肉付けをどう与えるかといったことは原論文では全く話題になっていないが，フォン・ノイマン 1945 年の「第一草稿」において物理的実装は主要な話題の一つとして扱われている．

ここで問題となるのはフォン・ノイマンの位置付けである．不完全性定理の証明においてゲーデルに先を越されて以降，フォン・ノイマンが数理論理学の研究から距離をおいたことはよく知られているが，彼が 1937–38 年頃には既にチューリングを数学者として高く評価し，チューリング機械という計算モデルとその能力について正確に理解していたことはほぼ確実であるとされる[41]．また，彼は計算機開発に携わっていた実地のエンジニア達にチューリングの論文を読むように勧めて回ったという証言も伝わっている（例えば IAS コンピュータの技術責任者を務めたジュリアン・ビゲローの証言がある[42]）．それゆえフォン・ノイマンがいつから既に開発が進んでいた計算機とチューリング機械とを結びつけて考えるようになっていたか，というのは興味深い問題であり検討の余地がある．しかしフォン・ノイマンが現実の計算機開発プロジェクトに携わるようになったのは 1944 年夏ごろからであり，彼が 1945 年の「第一草

[41] Petzold [2008]2012, pp. 247–248. チューリングとフォン・ノイマンとの交流については Hodges（[2012]2015）全編に詳しく触れられている．チューリングの方もフォン・ノイマンのことを早くから認識しており，1933 年にはフォン・ノイマンの『量子力学の数学的基礎』に取り組み，チューリングが最初に公表した数学論文はフォン・ノイマンの概周期関数に関する成果を拡張したものであった（*ibid*., pp. 121, 165–166 ）．

[42] Dyson [2012]2017, ch. 13.

稿」で提示したアイデアは彼のみに帰せられるものでは決してない．「第一草稿」の意義は 1945 年時点における最先端の知見を整理して広めたことにある．そして，そこにはチューリングにかぎらず数理論理学への明示的な言及はほとんどない．以上を勘案すると，仮に早い段階でフォン・ノイマンの中で両者が結び付けられていたとしても「チューリングが 1936 年に提示した万能チューリング機械に具体的な形を与えたのがフォン・ノイマンが 1945 年の『第一草稿』で提示したプログラム内蔵計算機である」とする主張は基本的に憶測の域を出ない[43]．

3 章で確認したように，プログラム内蔵概念の導入は極めて実践的な内容について述べている文脈の中で行われたものである．ここでもう一度エッカートを参照してみよう．彼は 1946 年夏のムーア・スクール講義において，命令とデータを同じ領域に記憶することに言及しつつ「計算の進展を阻害することのないように，命令は高速に利用可能でなければならない」と述べた上で，計算機に解かせる問題の種類に応じて，これら数ないし命令を記憶するために必要な空間の大きさが大きく異なることを指摘し，機械を構成する際に最大限の柔軟性と経済性を得るためには，命令とメモリで別々のメモリ・コンポーネントをそれぞれに用意するのではなく，両者を単一の貯蔵部として扱うのが良いとする結論を引き出している[44]．つまり，計算機内部に命令専用のメモリとデータ専用のメモリを置いた場合，問題の性質によって一部のメモリが活用出来ずに無駄になる可能性がある．しかし両者を同じ領域上において柔軟に対処すればそのような問題は生じない．このように捉えた場合，「プログラム内蔵」概念はあくまで工学的要請の結果生じたものと考えた方が整合的であるように思われる．

マーク・プリーストリーによれば「計算機製作の世界を除けば，プログラム内蔵という原則はほとんど即時的な関心を引くことはなかった．1950 年以前の資料において，新しい計算機とまだ開発中の計算機は一緒くたに扱われていたし，そうした計算機はまず第一に自動的に計算を実行する能力によって，第二に電子技術によって得ら

[43] このような主張は例えば Davis（[2011]2016）や Dyson（[2012]2017），Copeland（[2012]2013）などに見られる．Davis はヒルベルトの第十問題解決に貢献したことを始めとして，多くの業績を残している数学者であるが，彼は例えば次のように述べている「[注：ECVAC 報告書には] チューリングの名前には一言も言及されていないのだが，眼識のある人が見ればその影響は明らかである．チューリングと同様にフォン・ノイマンも，人間の頭脳の驚くべき知力は万能計算機として働く能力を備えているからではないか，と推察していた」（Davis [2011]2016, p. 177）．一方，歴史研究者の Thomas Haigh は「コンピューティングの歴史家の間では，チューリングがフォン・ノイマンに対して及ぼしたかもしれない影響について議論するのは parlor game になっている」（Haigh 2013）と述べ，このような問題は歴史研究者にとって真剣な検討に値しないという判定を下している．このように，チューリングの評価を巡っては計算機科学の実践家と歴史家との間で著しい意見の対立がある．

[44] Eckert 1985.

れた**高速な計算速度**によって特徴づけられていた.」[45]「プログラム内蔵」は後世においては新しい計算機械の決定的な特徴とみなされるようになったが，同時代的にそのような理解があったわけではないという指摘は近年多くの歴史家が行っている[46]．この指摘は「第一草稿」冒頭の記述からも裏付けることができる．

1.1 以下の検討は極めて高速な自動デジタル計算システムの構造を，とりわけその論理的制御を取扱う．[後略]

1.2 自動計算システムは（通常高度に複合的な）装置で有り，かなりの程度複雑な計算を行うための，例えば 2,3 の独立変数を持つ非線形偏微分方程式を数値的に解くための命令を実行することができる.[47]

また，ここで検討されている計算機は数値計算を主な目的としていたこともまたこの記述から伺うことができる．フォン・ノイマンがここで計算機が解くべきものとして想定しているのは応用数学の問題，特に「非線形偏微分方程式を数値的に解く」ことである．EDVAC は ENIAC の後継機として開発プロジェクトが進められていた計算機であるが，ENIAC は元々弾道計算を目的として開発されたものであったし，当時のフォン・ノイマンは爆縮や衝撃波といった問題に取り組む中で高速計算機械に関心を抱いたという経緯がある．こうした事を鑑みれば，EDVAC が「数値計算機械」として記述されているのはむしろ自然である．

また，チューリング機械という計算モデルに実用的な利用法が存在するということは 1930–40 年代において全く明らかでなかったことも確認しておきたい，例えば，ジョージ・ダイソンは Dyson（[2012]2017）の中で自身の父親であるフリーマン・ダイソンの次のようなコメントを紹介している．

> 「1942 年にトリニティ・カレッジの図書館でチューリングの論文を読んだのを覚えている.『なんて素晴らしい数学の研究だ』と思ったよ．でもこういう結果を実用化する人がいるとは思いもしなかった.」[48]

1930–40 年代にかけて，数学の一分野としての数学基礎論と当時最先端の技術ないし工学の産物であった自動計算機械を結びつけて考える人間は，仮にいたとしてもごく

[45] Priestley 2011, p. 154.

[46] cf. Daylight 2016; Bullynck et al. 2015.

[47] von Neumann 1945, p. 1. 強調は原文.

[48] Dyson [2012]2017, 下巻 p. 173.

少数であった[49]．数学の世界においてすら，数理論理学は解析学や代数学のような他の分野とはやや離れた位置にあったとされる[50]が，そうであるならば電子技術ないし通信技術を専門とするエンジニア達にとって，数理論理学の文脈の中にあったチューリングの仕事はいっそう未知の世界であったものと推察される[51]．もちろんフォン・ノイマンがチューリングの仕事に知悉していたことが，この発想を明確に表現する上で役に立った可能性は否定できないが，仮にそうであったとしてもこの着想が EDVAC の設計の中に取り入れられたのは実践的・工学的な議論の結果であったと考えた方が自然である．それゆえ「チューリングの仕事が先行して存在していなければデータと命令を同じ記憶領域に記憶する発想は登場しなかった」とする主張を支持することはできないし，同様に，「論理学と計算機工学が合流したことによって現代的な計算機が誕生した」とする主張も支持することはできない．1945 年時点において，（一部の例外的な人物の頭の中を除けば）両者は別の文脈にあって没交渉であった．

## §5 結論

本論で検討したのは，チューリングを「コンピュータの父」とする見方は妥当か，という問題であった．以上検討した通り，この見方は妥当ではない．チューリングが考案したのは「機械的手続き」を数学的に理解するための抽象的計算モデルであり，一方「第一草稿」に描かれた計算機は「高速自動計算機械」という工学ないし技術の産物であった．1940 年代において両者はまだ別々の流れにあり，これが合流するのはもう少し後のことである．

工学的産物としての巨大な計算機械と抽象的計算モデルとしてのチューリング機械との対応関係は当初から自明なものとして受けいれられたわけではなく，むしろ後世に至って発見され，徐々に認識されていったものと考えられる．そもそも，計算機開発の初期において，ほとんどの専門家は真空管回路や磁気ドラムの技術的問題と格闘していたか，あるいは数値解析を発展させることに集中していたのであって，彼らにとって計算可能性の理論などほとんど意味を持たなかった．計算機という当時の技術

[49] Priestley（2011, p. 125）などにも同様の指摘がある．但し，チューリング本人は万能チューリング機械の物理的実現に当初から関心を持っていたとされる（Hodges [2012]2015, p. vii）．

[50] Dyson [2012]2017, 下巻 p. 174.

[51] Daylight（2015）はチューリングの仕事の意義は 1940 年代のハードウェアの構築へ与えた影響にあるのではなく，むしろ後のプログラミング言語や計算量の理論などを始めとするアルゴリズムの理論的研究に理論的基盤を提供したことの方にあると指摘している．

の粋を集めた機械と抽象理論を対応付けるためには，計算機全体の振る舞いを工学的な詳細を捨象して理解する必要があった．そしてそのような抽象的な視点から計算機を理解することが出来るまでには，一定の時間が必要だったのである．

## 参考文献


Adams, Rod. 2011. *An early history of recursive functions and computability: from Gödel to Turing*. Docent Press.

Aspray, William. [1990] 1995 年.『ノイマンとコンピュータの起源』杉山滋郎・吉田晴代訳．産業図書．［原書：*John von Neumann and the origins of modern computing* (MIT Press)］

Bullynck, Maarten, Daylight, Edgar G, and De Mol, Liesbeth. 2015. Why did computer science make a hero out of Turing? *Communications of the ACM* 58: pp. 37–39.

Campbell-Kelly, Martin, Aspray, William, Ensmenger, Nathan, and Yost, Jeffrey R. 2014. *Computer: A history of the information machine*. Westview Press.

Ceruzzi, Paul E. 2003. *A history of modern computing*. MIT press.

Copeland, B Jack. [2012] 2013 年.『チューリング：情報時代のパイオニア』．服部桂訳．NTT 出版．［原書：*Turing: Pioneer of the information age* (Oxford University Press)］

Copeland, B. Jack. 2015. The Church-Turing thesis. In *The Stanford Encyclopedia of Philosophy*, Summer 2015 ed., ed. Zalta, Edward N. Metaphysics Research Lab., Stanford University.

Davis, Martin. [2011] 2016 年.『万能コンピュータ』．沼田寛訳．近代科学社．［原書：*The universal computer: The road from Leibnitz to Turing*(Taylor & Francis)］

Daylight, Edgar G. 2015. Towards a historical notion of turing — the father of computer science. *History and Philosophy of Logic* 36: pp. 205–228.

Daylight, Edgar G., Wirth, Niklaus, Hoare, Tony, Liskov, Barbara, and Naur, Peter. 2012. *The dawn of software engineering: From Turing to Dijkstra*. Ed. Grave, Kurt De. Lonely Scholar.

Dyson, George. [2012] 2017 年.『チューリングの大聖堂（上・下）』吉田三知世訳．早川書房．［原書：*Turing's cathedral: The origins of the digital computer.* (Vintage Books))］

Eckert, Presper. 1944. Disclosure of magnetic calculating machine. ［http://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-8-pdf/k-8-u2775-Disclosure.pdf（2018 年 2 月 28 日閲覧）］

———. [1946] 1985. A preview of a digital computing machine. In *The Moore School lectures: theory and techniques for design of electronic digital computers*, ed. Campbell-Kelly, Martin and Williams, Michael Roy. MIT Press.

Godfrey, Michael D and Hendry, David F. 1993. The computer as von Neumann planned it. *IEEE Annals of the History of Computing* 15: pp. 11–21.

Goldstine, Herman H. [1972] 1980 年.『計算機の歴史 —パスカルからフォン・ノイマンまで—』. 末包良太・米口肇・犬伏茂之訳. 共立出版.［原書：*The computer from Pascal to von Neumann* (Princeton University Press)］

Haigh, Thomas. 2013. ‘Stored program concept’ considered harmful: History and historiography. In *Conference on Computability in Europe*: pp. 241–251.

———. 2014. Actually, Turing did not invent the computer. *Communications of the ACM* 57: pp. 36–41.

Haigh, Thomas, Priestley, Mark, and Rope, Crispin. 2014. Reconsidering the Stored-Program Concept. *IEEE Annals of the History of Computing* 36: pp. 4–17.

Hodges, Andrew. [2012] 2015 年.『エニグマ：アラン・チューリング伝』. 上巻：土屋俊・土屋希和子訳，下巻：土屋俊・土屋希和子・村上裕子訳. 勁草書房.［原書：*Alan Turing: The Enigma* (Random House)］

Mahoney, Michael Sean and Haigh, Thomas. 2011. *Histories of computing*. Harvard University Press.

Mounier-Kuhn, Pierre. 2012. Logic and computing in France: A late convergence. In *Proceedings of the Symposium on the History and Philosophy of Programming*. ［https://s3.amazonaws.com/academia.edu.documents/32537797/Mounier-Kuhn_longabstract_HAPOP_2012.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1520397726&Signature=zDVNe8Kn%2BZVHP0ezCQ2FKZulc4o%3D&response-content-disposition=inline%3B%20filename%3DLogic_and_Computing_in_France_A_Late_Con.pdf（2018 年 2 月 28 日閲覧）］

Petzold, Charles. [2008] 2012 年.『チューリングを読む』. 井田哲雄・鈴木大郎・

奥居哲・浜名誠・山田俊行訳．日経 BP 社．［原書：*The annotated Turing: a guided tour through Alan Turing's historic paper on computability and the Turing machine*(Wiley Publishing)］

Post, Emil Leon. 1936. Finite combinatory processes — Formulation 1. *The Journal of Symbolic Logic*, 1(3). pp. 103-105.

Priestley, Mark. 2011. *A science of operations: machines, logic and the invention of programming*. Springer Science & Business Media.

Turing, Alan Mathison. 1936. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society* 2: pp. 230–265.

von Neumann, John. 1945. First draft of a report on the EDVAC.［`https://library.si.edu/digital-library/book/firstdraftofrepo00vonn`（2018 年 2 月 28 日閲覧）］

小山俊士．2014 年．「フォン・ノイマンの EDVAC 草稿と二進法」『津田塾大学数学・計算機科学研究所報』第 36 号，1–22 頁．

佐野勝彦・杉本舞．2014 年．『コンピュータ理論の起源 第 1 巻 チューリング』伊藤和行編．近代科学社．