(2) The Role of Logic in Computer Science and Artificial Intelligence
(論理，計算機科学および人工知能：歴史的視点)

J.A.Robinson(シラキュース大学教授)

［略歴］

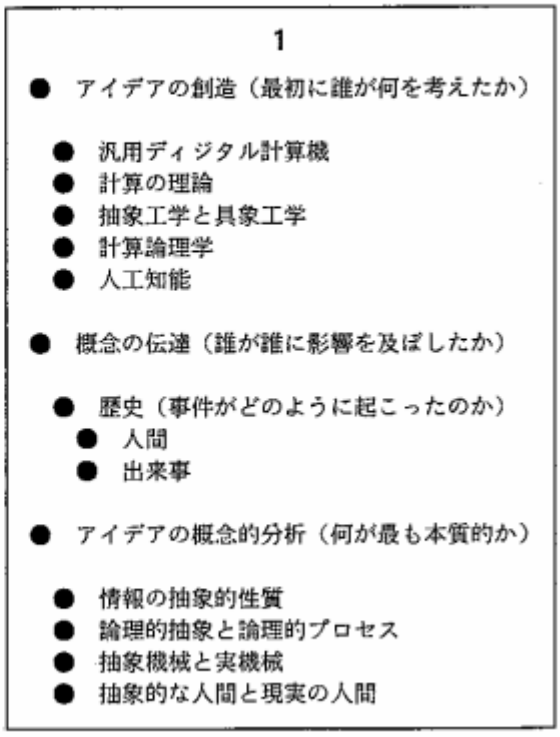
J. Alan Robinson。1952年ケンブリッジ大学卒業。1956年，プリンストン大学において哲学のPh. D. 取得。Robinson氏による融合原理の研究は，論理プログラミングのベースとなり，後に第5世代コンピュータ・プロジェクトの基盤となる。昨年，東京大学で1年間教鞭をとり，現在はシラキュース大学教授。

おはようございます。この会議で講演の依頼を受けたことは，非常に光栄で喜ばしいことです。講演の議題は，古川康一教授と田中英彦教授から提案されたものです。計算機科学と人工知能の発展において，過去と現在において論理と論理学者が果たしている役割について講演する機会が与えられたことを心から喜んでいます。

私の見解では，計算機の概念的な原点は，特定の先駆者的な論理学者たちの研究に見いだすことができます。論理は，最初から計算機科学の主な構成要素でしたし，またこれからもその事実は変わらないとおもいます。論理は，過去においては人工知能の分野でも重要な役割を果たしていましたが，将来は，それが計算機科学の一分野である人工知能において果たす役割は比較的小さくなるでしょう。

1980年代の初めに第五世代コンピュータシステムプロジェクトが発表されたとき，ご存知のように，組織委員会ではプロジェクトのテーマとして論理，特に論理プログラミングを選択しました。論理は，計算機とプログラミング言語の設計を統一する基準および原則の役割を果たすと強調されました。ご存知のように，この選択は多くの人を驚かせました。多くの人は，この選択を，計算機と計算を扱う上で，驚くべきかつ危険な方向転換であると考えました。なかには，一種の変革の始まりと見なす人さえいました。このような反応は確かに理解できるものの，それは間違っていたと思います。論理を強調することは実際に根元的なことですが，それは文字どおりの意味しか持っていません。この決定によって，計算機と計算機科学の原点に戻ることができたのです。私は，論理は，当然の選択であったと考えます。

本日の議題は，スライド1に示すように構成されています。次のような，質問を行います。誰がいつ何を行ったのか。一般的な目的に使用されるディジタル計算機，いわゆる汎用機械を最初に考えたのは一体誰なのか。現在我々が計算の理論と呼んでいるものは，どこで誕生したのか。計算論理学と考えているものはどこで誕



生したのか。人工知能の場合はどうなのか。このような概念がどのように生まれたのかについてだけでなく，それが現在までどのように受け継がれてきたのか，つまり誰がその概念をどのように伝えたかについても考えるべきです。

計算機の発展に関わってきた人々やそれに関連する出来事だけでなく，概念そのものについても議論しなければなりません。抽象と具象の違いが，議論の中で何度も取り上げられます。我々は，常にプロセスや構造から，最も重要な特徴と思われるものを抽象化します。つまり，分析や理解に関係ない枝葉末節な事柄は破棄します。たとえば，抽象機械と実機械についても議論します。また，抽象的な人間と現実の人間についても議論します。なぜならば，この議論は，人工知能の分野での我々の研究を見るための良い方法だと思うからです。我々は，知能に関する最も本質的な事柄だけを取り扱うことができるように，知的な生成物から枝葉末節をすべて破棄しようと試みます。

計算機を理解しようとする場合には，抽象と具象の違いを明確にしておくことが最も重要です。この違いの抽象的な側面を強調するからといって，計算機の工学的な側面をないがしろにするつもりは全くないことをとりあえず言っておきます。本当は，全くその反対なのです。実際に計算機を築くために過去数十年にわたってエンジニアたちが生み出した驚嘆すべき業績を正しく判断するには，否応なしに前述の違いを明確にしなければなりません。そして，最も高度な抽象化のいくつかを現実の機械で実際に実現できることは，明らかに非常にすばらしいことです。

これから抽象計算機と呼ぼうとしているものをもっと詳しく見てみると（スライド2），それは実際には形式的論理体系であることがわかります。抽象計算機は，論理学者や抽象数学者が長い間研究開発に取り組んできたもので，定義，推論ルール，および公理の集合です。また，現実の計算機は，電子回路と電気機械の構成要素から成る物理的な機械です。その動作は，物理的な法則に基づいて，機械の設計によって，また外界から機械に受信される入力によって因果関係的に決定されます。どの現実の機械，特にどの現実の計算機も，率直に言えば，抽象機械を物理的に実現したものです。現実の計算機を理解することは，単に計算機が実現する抽象を把握することなのです。すなわち，知的なプログラミングを行うためには，物理的な機械の「因果関係論理」に従う必要はありません。

私自身を含めて多くの研究者は，抽象チューリング機械でプログラムの作成を覚えました。

2　抽象計算機と現実の計算機

- この講演では，計算機を抽象的な視点から取り上げます。そして，抽象の実用化に役立つ本当にすばらしい技術を生み出した，すべてのエンジニアに，賞賛と畏敬の念を捧げます。
- 抽象計算機は，定義，公理，およびルールから成る形式的論理体系です。
- 現実の計算機は，電子回路と電気機械の構成要素から成る機械です。この計算機は，物理的な法則と，機械が外界から受け取る入力に完全に支配されます。
- どの現実の計算機も，抽象計算機を実現したものです。
- 私がプログラミングを始めた40年前に初めて使用した計算機は，抽象チューリング機械でした。その機械を机上でシミュレートするのは，時間のかかる，単調で退屈な作業でした。論理の視点から考えると，「現実の」機械を扱うのと同じことでした。

3 ALAN M.TURING 1912 – 1954

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM,1936

- この論文の万能チューリング機械は，すべての汎用ディジタル計算機のプロトタイプです。

SYSTEMS OF LOGIC BASED ON ORDINALS,1938

- Turingのプリンストン大学での博士号論文

ACE REPORT,1945

- スタック，プログラム記憶方式，サブルーチンの概念

CHECKING A LARGE ROUTINE,1949

- プログラムの最初の形式的な正しさの証明

INTELLIGENT MACHINERY,1948

- 人工知能技術に関する詳細な論文

COMPUTING MACHINERY AND INTELLIGENCE,1950

- 有名なチューリングテストを人工知能に導入

もちろん，現実の機械はなく，抽象機械だけでした。その定義と操作ルールは，論理の教科書やTuringの1936年の論文に記載されています。そして，その純粋に論理的な記述に基づいてプログラムを作成しました。

プログラムを「実行」するには，紙と鉛筆で遷移ルールを適用し，機械の動作を「シミュレート」しなければなりませんでした。論理の視点，計算のプロセスを理解するという点では，見かけ上は電子回路の構成要素から成る「現実の」チューリング機械を扱うのとちょうど同じことでした。ちょうど，無関係な枝葉末節な事柄の問題と同じで，状況の本質は変わりありませんでした。

Turingという名前は，「チューリング機械」というフレーズの単なるレッテルとしてお目にかかるのが普通です。しかし，この講義では，Alan Turingは現代の計算機の歴史に貢献した，２人の傑出した人物の１人（もう１人はJohn von Neumann）であると主張したいと思います（すべての人にご賛同いただけるとは思いませんが)。どちらの研究者も並外れた人間でした。どちらも能力的に優れていましたが，その他の点では両者は両極端に位置する人間でした。年は，Turingの方がvon Neumannより10才ぐらい若かったのですが，ある意味ではTuringの方が知能的には年上だったと言えましょう。Turingは，我々がvon Neumannに関連付けている概念を実際に考え出しました。

Turingの一生はあまり長くありませんでした（スライド３)。1912年に生まれ，1954年の42回目の誕生日を迎える直前に亡くなりました。とはいえ，それは多忙で充実した一生でした。1936年に発表した論文はすぐに優秀な論文として認められました。それは数学基礎論における長年にわたる問題である，いわゆるHilbertの

4　JOHN von NEUMANN　1903 − 1957

- 1920年代の最も優秀な若手の数学者かつ論理学者
- ゲッティンゲンで，公理集合理論とHilbertの形式主義プログラムを主導，1921 − 1930
- Godelが1930年に世間の注目を集めたあと，物理学と応用数学に専念，1930 − 1944
- 1935年ケンブリッジ大学で初めてTuringと出会う
- プリンストン大学でTuringと定期的に接触，1936 − 1937
- 1943年にTuringと再会していたと思われる
- 計算機設計の基礎として，Turingの1936年発表の論文を研究するように同僚に勧める
- Eckert−Mauchlyプロジェクト（ENIAC,EDVAC）で主要な役割を果たす，1944 − 1946
- IAS計算機の設計と構築，1945 − 1952
- 自己増殖オートマタ：理論的神経科学，生物学的情報処理，1943 − 1957

決定問題の一部の解決に寄与しました。決定問題を解決するために，Turingは計算機を発明しました。その論文で定義されかつ十分に開発された，いわゆる万能チューリング機械は，実際には，概念的にも歴史的にも，その後に続くすべての汎用ディジタル計算機のプロトタイプとなりました。

第二次世界大戦直後の1945年に，TuringはイギリスでACE計算機を設計しました。その設計に関する，非常に理解しやすく読みやすい説明の中で，Turingはスタック，プログラム記憶方式，およびサブルーチンの概念を提唱しました。現在では非常に馴染み深い，この概念は，このTuringの論文で初めて取り上げられました。数年後の1949年，Turingは，『大きなルーチンのチェック』という講義において，プログラムに関する形式的な推論の方法論，特にプログラムがその仕様に適合することを証明するための方法論を導入しました。当然のことながら，Turingは現在では人工知能(AI)を発案したことで広く知られてします。Turingは人工知能の創始者としてよく引き合いに出されますが，それはごく当然であると思います。Turingに関してそれほど知られていないこととしては，1950年に有名な「チューリングテスト」という哲学的な論文が発表される２年前に，人工知能のより技術的な側面をかなり徹底的に議論し，当時Turingが考えていた人工知能の概念や困難な問題点を検討したことが挙げられます。Turingが当時の人工知能研究のレベルをはるかに上回っていたことは，疑う余地がありません。

von Neumannに目を向けてみましょう（スライド４）。Turingは生まれながらの才能に恵まれた，途方もない天才ではあったものの，非常に風変わりで，大体においてかなりの異端者でした。したがって，「アウトサイダー」と呼んでもいいでしょう。一方，von Neumannは正反対で，文字どおり「インサイダー」でした。並外れた知能を，生まれながらに持ち合わせていました。その点では，Turingを含め誰よりも優秀でした。von Neumannほどの人は他にはいなかったでしょう。

von Neumannは，きわめて優秀でした。しか

も政治的手腕に優れ，保守主義を支持し，絶大な影響力を持つ現実的な人物でした。産業界と政府のトップレベルの研究に科学者としてリーダーシップを発揮し，一生を通じて国策を正しい方向へ導くことに貢献しました。von Neumannも短命でした。von NeumannとTuringは，どちらも若くして悲劇的な最期を遂げてしまう。しかし，von Neumannが計算機の歴史において並外れた影響力を持つ研究者であったとしても，彼の役割はTuringの果たした役割に次ぐものであり，Turingから派生したものであると思います。

歴史学者は，熱心に，この２人の関係を解明してきました。von Neumannは，Turing，特に彼が1936年に発表した論文に影響を受けたこと，そしてvon Neumannの力強い個性と精神でTuringの概念を押し進め，それを実現させたことが現在明らかになっています。いわゆるvon Neumann計算機は，本当はTuring計算機と称すべきです。

von Neumannは，おそらく1920年代の最も優秀な若手論理学者です。von Neumannが初めてTuringに会ったときには，Turingは23才のケンブリッジ大学の大学院生でしたが，このときvon Neumannはすでに有名でした。その後も２人はプリンストン大学でかなり定期的に接触を持っていました。プリンストン大学では，Turingは1936年から1937年まで大学院生として過ごし，Alonzo Churchと共同で研究を行っていました。TuringとvonNeumannの２人の研究室は，数学図書館とプリンストン大学数学科の研究室が入っている，ファインホールと呼ばれる小さな建物の中にありました。ファインホールは，数学者たちが交流しやすいように設計されており，理想的な研究環境を提供していました。当時，Institute for Advanced Studyと呼ばれる研究組織がプリンストン大学構内に新たに設立されました。そして，von Neumannもその組織の一員に名を連ねていました。ただし，それが当時ファインホールに入っていたという点を除けば，大学とは一切関係ありませんでした。その組織で，von NeumannとTuringは偶然知り合うことになりました。２人がお互いに影響を与えたという証拠が数多くあります。この２人の議論の主なテーマの１つは，Turingの1936年の論文に基づく概念と成果であったと思われます。

1939年に第二次世界大戦が始まると，２人は非常に重要な軍事研究に従事することとなり，Turingは暗号解読に，von Neumannはほとんどすべての分野の研究に携わりました。1942年11月から1943年３月までTuringはアメリカに留まり，多くの研究者と接触していたことがわかっています。この期間に，Turingは，ベル電話研究所のClaude Shannonを２ヶ月間にわたって訪問しています。TuringとvonNeumannがこの期間に一緒だったかどうかはまだ明らかになっていませんが，２人は会っていたに違いないと推測する人が多数います。私も同じように考えており，２人が実際に会っていて，1943年当時の最先端技術を代表する自動計算に関する概念を交換し合っていたと思います。

1943年，アメリカとイギリスでは，ディジタル計算機のための真空管技術がちょうど開発されようとしていました。アメリカでは，主としてペンシルバニア大学Moore SchoolのENIACとEDVACという計算機プロジェクトでした。イギリスでは，主にBletchley Parkコード解読機械でした。当時，米英両国が軍事研究の科学部門において，あらゆる関連技術をお互いに共有していたことはかなり明白です。したがって，von Neumannに対するTuringの影響は戦時中もずっと続いていたと思います。von Neumannが同僚に対してTuringの1936年発表の論文を研究するように勧め，基本的に

は，その論文は（汎用ディジタル計算機の設計で）これから行おうとしている研究にとって必要不可欠であると述べたという証言を数人から得てします。ご存知のように，von Neumannは，最初の汎用機械であるEDVACの設計と構築に主要な役割を果たすようになりました。プログラム記憶方式などを採用している現実の計算機の中で，どれが本当の最初の万能機械であったかについては，いくつか疑問があります。おそらくEDSAC（Moore SchoolのEDVACに基づくケンブリッジ大学の機械）か，またはマンチェスター大学のプロトタイプ機械でしょう。しかし，単にどの機械が最初に機能したかということは，ここではそれほど興味ある点ではありません。結局，ここで話しているテーマは，抽象を実現する最初の物理的な機械が実際に動くよりも，10年も前から存在していた抽象化というテーマだからです。

von Neumannは，物理的な機械，特にInstitute for the Advanced Studyで開発したIASシリーズの機械（von Neumannが基本設計を監修したという意味で彼が開発した機械）に実際に大きく関わっていました。その後すぐに，人工知能にもっと直接的に影響する研究に移りました。von Neumannは，今日我々がオートマタの理論と考えている概念，特に細胞的な疑似生物学的オートマタの理論を本質的に構築しました。特に，オートマタが自動的にどのように自己増殖するかという，不可能と思われるような問題を解決しました。驚くべきことに，von Neumannの理論構造は，抽象的には自然界に当てはまるものと同じです。しかし，von Neumannは，CrickとWatsonが1953年にDNA分子の構造を発見するよりも前に，その構造を発見し理論体系を記述しています。なんという偉大な業績でしょう。

次に，計算機の歴史において，Turingが抽象以外の分野で行った研究活動を見てみましょう（スライド５）。第二次大戦直後のイギリスにおいて，Turingは，ACEと呼ばれる現実の機械の設計の責任者となりました。最初に動いた機械がACEでなかったのは，Turingがどちらかと言えば単純で，管理能力が劣り，政治的手腕に長けていなかったからにすぎません。自分自身を計算機分野の第一人者に仕立てあげるだけの早さで，現実の機械の構築を押し進めることができなかったにすぎません。他の研究者の方が現実の機械の構築では先んじたかもしれませ

5　TuringのACEの設計，1945

PROPOSAL FOR DEVELOPMENT IN THE MATHEMATICS DIVISION OF AN AUTOMATIC COMPUTING ENGINE,1945

- この提案で述べられている設計に基づいて，後に，Pilot ACEとDEUCEという現実のプログラム記憶方式のディジタル計算機が，イギリスの国立物理学研究所で構築され，かつ正しく運転された
  - スタック
  - サブルーチン
  - プログラム記憶方式
- ACEは万能チューリング機械を概念的に受け継いだ機械
- 1947年Turingは以下のように記述している：

数年前，私はディジタル計算機械の理論的可能性とその限界について研究「を行った。」私は，中心的なメカニズムと無限のメモリを持つ機械を考案した。ACEなどの機械は，私が考案したのと同じ種類の機械を実用化したものと見なすことができる。

6　Turingの論理と計算

- ディジタルという特性は電子工学という特性よりも興味深いはずである
- ディジタル計算機は，最終的には記号論理に対する興味をかなり刺激すると期待される
- 言語が厳密であれば，どの言語を使用してもディジタル計算機と通信可能である
- 原則的には，どの記号論理でも通信可能でなければならない
- これまでに比べて，論理体系を実用化できる範囲が一層拡大する
- 機械に数学的な公式を実際に処理させるために，何らかの試みが行われる

んが，それはこの論文のテーマではありません。ここで言っておかなければならないことは，実際に最初に動いた機械がどれであっても，その概念はTuringの概念であるということです。ロンドン数学学会が1947年に発行したサーベイで，Turingは，第二次大戦後に開発された現実の機械を自分の1936年発表の抽象機械に直接結び付けています。

スライド6のTuringの記述からの引用を見ると，ディジタル計算と論理をどのように関連付けたかわかります。たとえば，最後の引用を見ると，形式的な記号処理の概念を明らかに予見していることがわかります。これらの引用全体から受ける印象では，Turingは計算機械の基礎に横たわる基本的な概念の重要性は，その本質的に抽象的かつ論理的な特徴にあることを認識していました。計算機械が電子装置という形，または実際にその他の物理的な装置として実現また具体化され得たという事実は，興味深くかつ実際に重要ではありますが，純粋に科学的な視点から考えるとそれほど興味をそそられるものではありません。このような概念が偉大な力を発揮しているのは，その概念が急激に移り変わる技術の不安定な性質に左右されていないからです。

Michie教授とGood教授は，第二次大戦が終結する頃にはTuringがすでに自分の考案した万能機械を構築する計画を立てていたと証言しています。このTuringの計画の動機は，人工知能の構築にありました。ACE機械を設計した理由もまさにそれです。Turingの言葉を借りれば，脳を作り出したかったのです。その他は些細なことでした。人工知能の研究を開始し，有効なツールを作り出したかっただけです。Turingは，普遍性(universality)の概念に基づけば，万能機械以外の機械を構築する必要はないと考えていました。万能機械を構築してしまえば，あとはプログラミングと万能機械に対応するソフトウェアの作成の問題だけであると考えていたのです。

スライド7は，人工知能に対するTuringの考え方の一端を示してします。人間の場合について，現実と抽象の違いを真剣に考えていることは明白です。Turingの人工知能に対するアプローチは，現実の物理的な人間の些細な細部をすべて捨て去り，計算機ソフトウェア，つまりTuring考案の機械に対応する計算機プログラムという形式で，人間の本質のみの再生を考えることでした。最も重要なのは脳が何をしているのかということであり，これこそTuringの考えていたことでした。適切なプログラムを作成することによって，脳の活動を万能機械で行わせることも可能であると考えていました。これが，当時のTuringの人工知能に対する考

え方の要でした。Turingは，考える機械を構築するには，人間を全体として把握して，その人間を部分単位で人工的なサブシステムに置き換えるだけであると主張しています。これは，人工物による置換と言えるかもしれません。現在の技術では，すでに義歯，義足，人工骨，人工関節，人工心臓などを人体に適用することができます。この傾向はどこまで行くのでしょうか。(技術の進歩により）人体の各部を置き換え続けて，最後には脳のまさに神経細胞まで置き換えるところにたどり着くと考えると，わくわくします。

スライド8では，前述のTuringの概念がvon Neumannを通して伝達，拡大，かつ普及されたことを裏付ける証拠を，いくつか取り上げています。この２人の研究者に関して，現在，Andrew Hodges著　Alan Turing: the Enigmaと William　Asprey著　John　von Neumann and the Origins of Modern Computingというすばらしい伝記が出版され

7　Turingの機械の知能

- 考える機械を構築できると信ずる大きな理由は，人間のどんな細部でも機械に模倣させることが可能だからである
- 「考える機械」を構築する方法の１つは，人間を全体として把握し，人間のあらゆる部分を機械に置き換えようとすること
- 我々に必要なのは，経験を通して自ら学習できる機械である。機械に自己の命令を変更させることができれば，そのメカニズムが提供される
- 機械が全く間違いを犯さないと期待するとしたら，同時に機械は知能を持つこともできない
- 人間は誰も知識そのものに非常に大きな貢献を行ったわけではないのに，なぜ機械に多くを期待するのか

8　von NeumannによるTuringの概念の伝達

- 万能チューリング機械はプログラム記憶方式計算機のモデルである（８年後に開発される）
  William Aspray, 1990
- von NeumannのTuringに対する支持の手紙，1937
- von NeumannがTuringに研究職を提供，1938
- von Neumannは，Turingの研究を他の研究者の前で賞賛した。IAS主任設計者およびエンジニアであるJulian Bigelowに，Turingの1936年発表の論文を研究させた。
- Stanislaus Ulamによると，1939年にvon Neumannは，Turingとその「すばらしいアイデア」を絶賛したということである
- von Neumannを計算機の父と称する人が多いが，von Neumann自身は間違ってもそのように自称することはなかったであろう。von Neumannは，基本的な概念はTuringによるものであることを私に強調した。Turingの1936年発表の論文が基本的に重要であることに，十分気づいていた。(von Neumannは）Turingの論文を私に紹介し，私は彼の勧めでそれを丹念に研究した
  Stanley Frankel,1972

9　CLAUDE SHANNON

- A SYMBOLIC ANALYSIS OF RELAY AND SWITCHING CIRCUITS,1938

- 電子回路による抽象的論理式の実現を初めて明示的に研究
これは，理論的に飛躍的な進歩だった。命題（ブール）計算の構文的，意味的性質と，リレーおよびスイッチ回路の演算動作の間に成り立つ形式的な類似性を解明した。

- 1943年の初頭, Turingは，２ヶ月間にわたってベル研究所にShannonを訪ね，スピーチスクランブルについて討論した。また確かに，両者はディジタル計算についても討論したようである。このときのアメリカ訪問で, Turingはvon Neumannとも会い，軍事的な計算の問題について討論した可能性がある。

ています。

Hodgesの伝記には, von Neumannは Turingがプリンストン大学の大学院生だったときに, Turingに対して支持する手紙を書いたことが記されています。von Neumannは Turingをよく知っており, Turingと彼の研究を賞賛し，しかもTuringに特別研究員の地位が与えられるように熱心に主張したようです。Turingがプリンストン大学で２年間におよぶ学生生活を終える頃, von NeumannはTuringにInstitute of Advanced Studyでの研究助手としての地位を申し出しまた。しかし，そのような名誉には目もくれず, Turingはケンブリッジ大学に戻るという理由でその申し出を断りました。その後すぐに明らかになったことですが, Bletchley Parkで軍事研究に従事することになりました。von Neumannが，この事実を記録した人々に対していかにTuringの研究を賞賛したかについては，すでに述べました。たとえば, Institute of Advanced Studyのプロジョクトでvon Neumannの主任エンジニア兼主任設計者を務めたJulian Bigelowは，そのプロジェクトに参加したら最初の課題として, Turing論文を研究したと言われています。おそらく, von Neumannの親友であった有名な数学者のStanislaus Ulamは, Adventures of a Mathematicianという著書の中で, von Neumannは, Turingとそのすばらしいアイデアをしばしば賞賛したと書いています。

Stanley FrankelとNicholas Metropolisは, ENIACが稼働したとき，その機械で動く最初の主要なプログラムを作成しました。それは，ある種の衝撃波の計算プログラムで，極秘事項でした。その後, Frankelは，計算機史の研究家であるBrian Randell教授に手紙を書いていますが，その内容はスライド８の最後の引用の所に出ています。Frankelは，計算機技師でもなければプログラマでもありませんでした。Frankelは，計算機のユーザであり，実際的な計算問題を解決しなければならない物理学者でした。

スライドの９と10は, Turingともう１人の人物，もちろん非常に有名なClaude Shannonとの接触を簡単に表してします。

今日計算機科学に従事する研究者で Shannonの研究に詳しい人はあまり多くないと思いますが, von Neumannが計算機分野に大きく踏み込んだとき，おそらくShannonの研究もvon Neumannの頭の片隅に入っていたと思います。Shannonの1938年発表の論文では，ブール論理（すなわち，本質的な命題論理の抽象的概念）がリレー・スイッチ回路に関連付けられており，この論文は当時の多くの研究者に大きな影響を与えています。また，この論文は，論理の設計を理解するための理論的な方法を導入し，計算機設計の問題に関わっている

１０　CLAUDE SHANNON

A MATHEMATICAL THEORY OF COMMUNICATION,1948

- 抽象的な情報の概念を初めて深く分析した論文。ハードウェアでの実現から情報（および計算）を分離した。表現とコーディングを強調した

A UNIVERSAL TURING MACHINE WITH 2 INTERNAL STATES,1956

- n状態とm記号を持つ任意のチューリング機械を，２状態と4m（n+1）以下の記号を持つ等価のチューリング機械に変換できる独創的な構造

- ShannonとTuringは，1943年初頭のベル研究所で２ヶ月間にわたって行われた「スピーチスクランブル」の会議の間に，ディジタル通信とチューリング機械の理論に関するShannonの考えについても話し合ったことは確実である

者すべてにとって突然悩みの種になった問題，たとえばスイッチの構成要素の数を最小化するためのアルゴリズムの開発について紹介しました。TuringとShannonの関係がうまくいっていたことや，1942年末から1943年初頭にかけてお互いにかなりの時間を話し合いに費やしたことに感銘を受けました。見かけ上も，また公式の発表でも，両者はスピーチスクランブル装置について討論しただけとなっています。しかし，両者がディジタル計算の分野で，やや異なるテーマについては討論しなかったということは考えにくいことです。Turingとvon Neumannが何らかの極秘の会議を持ったのは，おそらくこのときの訪問ではないかと思います。そして実際に，その会議は今でも極秘になっています。このときの会議の内容は，それに関連する記録が公開されたときに最終的に明らかにされるでしょう。

スライド10からわかるように，Shannonは通信の数学的理論または情報理論の先駆けとなった偉大な研究者でした。Turingはあらゆることに興味を持っていたので，このShannonの理論がTuringを引きつけたことは間違いありません。状況の本質的な特徴を，その特徴を実現する無関係で些細な物理的事柄から抽象化するという，もう１つのすばらしい具体的な例です。ShannonがJohn McCarthyと共同編集した1956年版のAutomata Studiesに，Shannonが，万能チューリング機械の構造の「可塑性」に関して，独創的な論文を寄稿していることは非常に興味深いことです。

スライド11に出てくる２つの重要な名前は，現在では半分忘れ去られているかもしれませんが，人工神経回路網（「神経回路網」）の再興が叫ばれているので，すぐにまた広まることになるでしょう。

McCullochとPittsは，1943年に人工神経回路網の概念を生み出しました。この同じ年に，TuringはShannon（そしておそらくvon Neumann）を訪ねています。McCullochとPittsは，特定の２次元のデータフローのような表記法を導入しました。この表記法では，離散状態オートマタを基本単位の回路網として表すことができます。基本単位は人工神経と考えられ，それぞれ他の類似の単位に結合するための入力回線と出力回線があり，有限状態スイッチ要素として動作します。２人は，抽象的ニューロサイエンスの研究を試みました。それは絶大な影響力を持っていましたが，おそらく２人の予想とは異なる種類の影響でした。特に，von Neumannは，計算機を考える上での正しいやり方として，その表記法とそれに付属する抽象

11　W.S.McCULLOGH AND W.H.PITTS

A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY,1943

- 人工神経回路網は，一般的な抽象的ディジタルスイッチ回路である
- McCullughとPittsは，Turingの万能機械を引用し，任意のチューリング機械を人工神経回路網として実現可能であると指摘した
- 人工神経回路網は，von NeumannとTuringが1945年に2人で設計した機械だけでなく，EDVAC,IAS,およびACEの各計算機の解説にも使用された
- Kleeneは，1951年に正規表現の概念を導入し，正規表現が，McCullughとPittsが提唱する神経回路網によって認識される対象そのものを正確に表現していることを証明した

モデルにすぐ飛びつきました。von Neumannは，計算機は有限状態の装置を結合した回路網として抽象的に表現できると考えており，それは電気的または電子的な構成要素で物理的に実現できると考えていました。しかし，McCullughとPittsの表記法で計算機を抽象的に設計することによって，実際に計算機を構築しなくても，計算機の動作を論理的に研究できます。抽象的にも論理的にも正しく理解できたと感じたとき，しかもその時にだけ先へ進むことで，実際のハードウェアの抽象化を理解することができます。McCullughとPittsの神経回路網は，von Neumannを始めとする研究者にとって計算機設計の方法論になりました。実際，Turingも，ACEに関する報告書の中でこの表記法を採用しています。

専門家である聴衆の皆さんの中には，McCullughとPittsの表記法と，いわゆる抽象計算機科学の正規表現との関係に気づいている人もいるでしょう。

スライド12は，計算機科学と人工知能の両分野に論理的なアプローチを採った，影響力を持つ先駆的な研究者の1人，John McCarthyに関するものです。彼は，現在でも精力的に研究を行っています。McCarthyは，不滅のプログラミング言語であるLISP（文字どおり不滅の

12　JOHN McCARTHY

A BASIS FOR A MATHEMATICAL THEORY OF COMPUTATION 1961

- 計算機科学は，知的なプロセスを機械にどのように実行させるかを研究する学問である
- 計算と数学的論理との関係が，19世紀の解析学と物理学との関係のように，21世紀において実を結ぶように願うのは当然のことである

PROGRAMS WITH COMMON SENSE,1958

- ADVICE TAKERは，形式的な言語で書かれた文を操作することによって問題を解決するように提案されたプログラムである
- プログラムが，指示内容と既知の内容の直接の結果を十分に広範囲なクラスにわたって自分で自動的に導き出す場合，そのプログラムは常識を備えていることになる

言語である：今から何世紀にもわたって，人々がLISPプログラムを書き続けることは疑いない）を考案したことで一番有名であり，また名誉を受けているでしょう。歴史的には，LISPは最初の偉大な論理プログラミング言語でした。基本的に，形式的論理体系，いわゆるChurchのラムダ計算法にほんの少し追加修正したものです。McCarthyは，この図で示されている最初の論文から始まって，我々の考えに大きな影響を与えてきました。この論文では，LISPの基盤に横たわる論理的概念が詳しく論じられているが，帰納的定義と再帰が果たす役割の研究に関してはそれをはるかに超えています。現在よく知られている条件式を考案したのはMcCarthyであり，その論文の中で紹介されています。それは非常に重要な論文です。MaCarthyが研究に着手したのはこのように計算とプログラミング言語の理論の分野においてだけではなく，人工知能に対しても論理的なアプローチを強く押し進めました。McCarthyの1958年の論文が最初に発表されたのが，TuringがACE報告書を作成するためにしばらくの間働いていた，イギリスのテディングトンにある国立物理学研究所であったという事実は興味深いことです。McCarthyは，単純に知識を伝えられるプログラムというすばらしい概念を提唱しました。このプログラムは，実際に，公理演繹体系，すなわち計画立案とロボットの動作の制御などの土台となる演繹的知識のベースを実現するでしょう。現在ではありふれた考えではあるが，ここが出発点なのです。それは，あたかもTuringが1954年に論理学の火を大学院を卒業したばかりの若手のMcCarthyに手渡したかのような感じです。あとで詳しく述べるが，Marvin MinskyもTuringから論理学の火を受け継いでいるとも言えよう。Minskyは当時単に論理的なアプローチではなくTuringの精神に則って「脳の構築」に着手しようとしていたのです。

もちろん，ある点で論理プログラミングについて述べなければなりません。

スライド13の有名な等式アフォリズムは，Kowalski教授が創り出し，またしばしば引き合いに出されるものですが，プログラミングにおいて2つの構成要素，すなわち使用している知識とその知識の使用方法を分離しなければならないことを簡潔に指摘しています。そうすることによって，各構成要素を個別に扱うことができます。両方の構成要素が必要であることを強調しておかなければなりません。どちらか1つを無視してはいけません。実際の経験では，両極端な例を発見することがあります。研究者の中には，純粋な命令プログラムを書くことによってすべての論理を捨ててしまい，制御だけを残す者もいます。不思議なことに，チューリング機械のプログラミングは，そういうことを

13　論理プログラミング

- アルゴリズム　＝　論理　＋　制御（Robert Kowalski）
- 単にそれを記述（Alan Colmerauer）
- Prologの論理プログラミングは，Herbrandのユニフィケーションプロセスを使用して私が1965年に導出した述語論理のホーン節の特殊な例に基づく。ColmerauerのProlog IIIは，ユニフィケーションを制約解決に置き換える点で基本的なPrologより優れている
- LISPの論理プログラミングは，Churchのラムダ計算法のMcCarthy版に基づく異なる論理を使用する

意味しています。何をするかを単にステップごとに指示するだけであり，つまり制御を与えるだけになります。

論理だけを使用することに関して言えば，これを純粋な形で行ってきているのは論理学者と哲学者だけのようです。知識をそのように系統立てると，何も起こりません。それは単に時代を超越したプラトン哲学の１組の命題であり，その一部は公理であり，それ以外は定理となります。それは，導出したかどうかにかかわらず，公理の静的帰結です。何も起こりません。それは，計算の基本ではありません。計算では，実際に何かが起こります。たとえば，Prologは，Kowalskiの等式とColmerauerの動作原理を表しています。「単にそれを記述せよ」という動作原理は，単にそれが何であるかを記述しています。そして，計算システムはその記述を取り入れ，オブジェクトであればそれを構築し，アクションであればそれを実行します。これは，論理プログラミングがどんな感じのものか表現するには，非常に適切な方法です。

Prologの論理的な起源は，私個人にとっては非常に満足のいくものです。その基本となる概念は，私が1960年代初期に従事していた開発でたまたま使用していた概念と同じ１階述語論理から導出したものだからです。しかし，論理プログラミングの歴史はそれよりも深く，1930年代初期にまで遡ります。当時，Turingはケンブリッジ大学の大学院生でした。その頃，Jacques HerbrandとKurt Godelは独創的な研究を発表したばかりでした。Herbrandが特に論理に貢献したのは，現在我々が計算論理学と呼ぶ概念の前身を具体的に構築したことです。Herbrandの1931年の博士号論文には，たとえば我々が今日ユニフィケーションと呼ぶ概念に関する記述があります。それはやや曖昧に書かれているものの，それを最初に考えたのがHerbrandであることは間違いありません。現在では周知の事実ですが，ColmerauerのProlog−3の例の場合，このような概念はかなり一般化されているだけでなく大幅に拡大されています。したがって，単なるユニフィケーションの代わりに，制約解決を，論理プログラミング体系を支える指導原理として考える方が一般的です。とはいえ，基本概念は同じです。好き嫌いに関わらず（私はどちらでも構わない），LISPは別の論理プログラミング言語として分類されなければなりません。その論理は，Alonzo Churchのラムダ計算法という別の論理です。したがって，McCarthyがその抽象的論理体系を計算で処理できると考えていたことを考えると，彼は言ってみればLISPのKowalskiとColmerauerでした。

スライド14の名前の一覧は大まかに年代順に並んでおり，計算論理学の歴史に従って３つに

１４　計算論理

1879 − 1935	GOTTLOB	FREGE
	DAVID	HILBERT
	JOHN	VON NEUMANN
	JACQUES	HERBRAND
	EMIL	POST
	KURT	GODEL
	ALONZO	CHURCH
	ALAN	TURING
	HASKELL	CURRY
	STEPHEN	KLEENE
	J.BARKLEY	ROSSER
1955 − 1960	HAO	WANG
	PAUL	GILMORE
	MARTIN	DAVIS
	HILARY	PUTNAM
	DAG	PRAWITZ
1960 − 現在	WOODY	BLEDSOE
	ALAN	ROBINSON
	LARRY	WOS
	GEORGE	ROBINSON
	DON	LOVELAND
	GERARD	HUET
	ROBERT	KOWALSKI
	ALAIN	COLMERAUER
	KEITH	CLARK

分類されています。Fregeはほとんど何もない所から述語論理を考案し，今世紀初頭には，Hilbertが述語論理の発展の最初の数十年間を先頭に立って推進しました。von Neumannは，集合理論の論理的な体系化にきわめて重大な貢献をしたので，このグループに入れてあります。Post, Godel, Church, Curry, KleeneおよびRosserの各研究者は，1930年代に論理体系の発展で先駆的な役割を果たし，計算とその可能性の論理的理論を奥深く研究しました。この分野の主な成果の大部分は，1940年までに得られています。Turingの研究がこの成果の大半を占めていたのはもちろんです。1955年頃には，計算機が計算論理学などのあらゆる種類の研究に幅広く使用され始めました。その頃から15年間にわたって，Wang,Gilmore,Davis,PutnamおよびPrawitzが非常に重要な研究を行いました。これらの研究者はすべて，私を始めとする同世代の研究者に直接影響を与えました。私たちを学問的に指導し，莫大な可能性を示して私たちの興味を駆り立てたのは，この研究者たちでした。彼らは，自分たちではそのような可能性を実現するところまでは到達しませんでした。その後すぐに実現されたものの，お膳立てをしたのは確かです。３番目のグループには，年齢順に言うと，まずBledsoeと私が入ります。このグループの他の研究者は皆年下です。WosとGeorge Robinsonは1960年代半ばにArgonneグループを結成し，それ以降名声と成功を欲しいままにしています。Lovelandのモデル削除を取り入れた述語論理は，Prologの基盤に横たわるKowalski−Kuehner SLD導出と基本的に同じです。Huetは，ユニフィケーションアルゴリズムを高階述語論理に一般化しましたが，これは高階論理プログラミングと定理証明に向けての非常に重要なステップです。Woody Bledsoeは，その生涯をかけて計算論理学について研究し，「現実の」定理を計算機上で証明するためのシステムの開発に専念した数学者です。

KowalskiとColmerauerは，論理プログラミングを述語論理という点で提示しました。ここでKeith Clarkについて述べなければならないと思いますが，彼は「失敗による否定」という概念を発掘並びに展開しただけでなく，並列論理プログラミングの分野で先駆的な研究を行うことによって，抽象論理の枠組みに欠かせない研究者の１人となりました。現在まで時代を

１５　ハードウェアとソフトウェアとの交換可能性

- Turingの1936年発表の論文で，最初に理論的に提唱された
- 1945年に開発されたACEは「最小限のハードウェア」機械
- 1945年に開発されたEDVACは，ハードウェアを効率的に使用するために並列性を最小化
- この交換可能性の概念は，現在ではありふれた考え方である
- RISC計算機
  - 特殊なアーキテクチュアに代わるインタプリタ
  - シミュレーション
- 基本的な疑問：　ハードウェア形式とソフトウェア形式にはそれぞれ何が存在すべきか
- もう１つの疑問：　多種多様な万能機械の中でどの機械を使用すべきか

下ってくると，当然のことながら，このリストに入れるべき名前はまだたくさんあります。しかし，計算論理学の完璧な歴史を２分間で書こうとしても不可能です。計算論理学の研究に関わった研究者と，そのさまざまな貢献度を思い出してほしかっただけです。

スライド15は，ハードウェアとソフトウェアとの論理的な交換可能性の概念を示しています。我々はこの概念をあたりまえのこととして捉えていますが，Turingの万能機械という概念を実際に別の角度から捉えただけです。一定の最小限の「何か」をハードウェア形式に入力するだけで，その他のすべてのものが，単にハードウェアをプログラミングすることによって作成されたソフトウェアとして認識できるようになります。現在では，この概念は，RISCでお目にかかることができます。この計算機は，万能機械をどれだけ最小化できるかを，本格的に工学的に実現したものです。プログラミングで何もかも実行できるようにするには，少なくともどのくらいのハードウェアを所有しなければならないのでしょうか。RISCの概念でさえも，Turingのことを思い出させてくれます。

スライド16に示す現実的な方法を使用すると，現実の計算機と抽象計算機との違いが非常に重要なことが明かになります。特にvon Neumannにとって，それは非常に大きな問題でした。訴訟に巻き込まれ，ENIAC/UNIVACを設計したEckertおよびMauchlyとの間で非常に不愉快な摩擦が生じました。

何に対して特許権を取得することができるのか。概して特許権法は，これまでハードウェアの特許権を認めているだけです。概念に対しては，特許権を取得することはできません。しかし，進歩の推進力となるのは，しばしば概念なのです。ハードウェアとソフトウェアの間，または抽象と現実の間のバランスを考えると，抽象的な側面の重要性を認めたくなります。抽象的概念の創出に傾倒するように仕向けるために，抽象的概念の発案者がその功績に対して特許権を取得できるようにしたらどうでしょうか。たとえば，Turingが，20世紀だけでなくすべての時代で最大かつ最も有意義な発見の1つである万能機械の概念に対して，特許権を取得できなっかたかもしれないという点は注目に値します。

１６　抽象機械ではなく，現実の機械の特許資格

- 特許資格に関する基本的な法律的事実：　特許権は物理的な装置（ハードウェア）にのみ与えることができ，概念ないし科学的な原理には与えることはできない
- Turingは，自分が考案した万能機械の特許権を取得することができなかっただろう
- von NeumannがEDVAC報告書で示した表記技術は，ハードウェアと工学的な内容を抽象化し，McCullughとPittsの図を使用して抽象的にその基本的な概念だけを示すことだった
- これによって，摩擦はおろか訴訟にまで巻き込まれる
- Eckert & Mauchly対von Neumann
- Eckert & Mauchly対Atanasoff
- 論理を工学的な内容から抽象化することによって，von Neumannは，工学的な研究を行わずにEDVACの論理を抽象化することができ，その結果，設計を最初のドラフトで本質的に完成させることができた

Arthur W.Burks,1980

17　論理と技術

N.MetropolisとGian-Carlo Rota,1980：

- Turingとvon Neumannを除けば，1930年代に「数学的論理こそプログラミング言語と計算機設計の魔法の鍵である」と認識していた研究者はほとんどいない
- Peano,RussellおよびWhiteheadの記号体系, Gentzenによる証明の分析，およびChurchとTuringによる計算可能性の定義は，計算機革命の幕開けを記した

E.W.Dijkstra,1981：

- 論理は，事実を記述するための学問から，処方や指令を記述するための学問へと変化した。新しい論理学者は技術者である
- 機械に命令を与えることは，もはやプログラムの目的ではない。プログラムを実行するのが機械の目的である

すでに述べたように, EDVACの設計においてvon Neumannが取った表記技術は，すべてのハードウェア概念を捨ててしまい，プログラム記憶方式という概念を含めて，ディジタル計算機の抽象的レイアウトと機能に関するMcCullochとPittsの神経回路網の記述だけを残すことでした。von Neumannは, EckertとMauchlyの工学的な概念を抽象化して取り除いたので，この２人の研究者との間に摩擦が生じることになりました。２人の研究者は, von Neumannに自分たちが無視され，自分たちの革新的な発明の功績が否定されたことに憤慨しました。ある意味では, von Neumannは，実際にこの２人の研究者の言い分にあるようなことをやりましたが，その動機はEckertとMauchlyの言い分とは全く異なっていました。von Neumannは，自分が不当に褒められていると感じたときには，他の研究者，たとえばTuringを賞賛するように言っています。

長い年月を経た今となって, EckertとMauchlyがいかに重要な研究者であったかがわかります。なぜなら，高速で動く現実の機械の構築を始めとして，さまざまな発展に貢献したのは他ならぬ彼らだったからです。その後発生したEckert,Mauchly,Atanasoff, およびvon Neumannの業績に関する訴訟においては，誰がいつ何を発案したのかという点が問題でした。議論の対象がハードウェアであったことは，間違いありません。von Neumannと直接一緒に研究活動を行ったArthur Burksの指摘によれば, von Neumannが工学的な研究を行わずにEDVACの論理を抽象化することができ，また最終的に設計を１つの案として完成させることができたのは，まさしく工学的な内容から論理を抽象化したことによります。計算機設計の方法論の一環として，現実の計算機と抽象計算機との違いは，控えめに言っても非常に重要です。

スライド17では, von Neumannをよく知っていたMetropolisとRotaという２人の研究者が,「Turingとvon Neumannは1930年代に数学的論理がプログラミング言語と計算機の両者にとっての魔法の鍵になることを認識していた数少ない研究者の仲間だった」，と語っています。もっと最近では, Dijkstraは,「プログラム作成と計算機科学を正しく把握するならば，論理が工学の一分野になりつつある」と飽きることなく強調しています。しかしながら，プログ

１８　抽象的UNIVACか，または現実のUNIVACか

- 1956年に私がデュポン社に入社したとき，最初に与えられた仕事はEckertとMauchly Remington Randが開発したUNIVACのプログラムを作成する方法を学ぶことだった
- すべての関連する定義と規則（30ぐらいの命令，レジスタとメモリの論理的構造など）が，UNIVAC PROGRAMMING MANUALに記載されていた。この形式体系を研究し，抽象的なUNIVACのためのプログラムの作成について学んだ
- デュポン社の現実のUNIVACは，鍵の掛けられた室に入っていた。UNIVACプログラムを実行するには，それを受付に渡した
- 我々の知る限りでは，その鍵の掛けられた室の中には，現実のUNIVACは入っていなかったと思う。現実のUNIVACをシミュレートする処理能力の非常に高い人間を１チーム雇ったとしたらどうだろうか。実行時間を除けば，その違いを感知し得たであろうか

ラムを作るという行為は，抽象工学というべきものです。プログラミングにおいては，技術者が構築した機械を見せて，「それで何をしたいのか」と聞いてくれるまで待たなくてもよくなったのです。Dijkstraが指摘するように，現在はそれが全く逆になっています。プログラマの方が，自分たちの欲しい種類の機械を説明して，技術者に構築してもらっています。

プログラマは，常に抽象的な機械のためのプログラム作成を学んできました。実際の物理的な機械は，これまで決して「目に見える」機械ではありませんでした（スライド18)。

FORTRAN（スライド19）やLISP（スライド20）などの初期の「一段と高級な」プログラミング言語の場合にも，同じような状況でした。その場合，その言語の実行規則が組み込まれていて，処理すべきプログラムが与えられると，評価アルゴリズムに従って実行を行う機械

１９　抽象的なFORTRANマシン

- 1959年においては，FORTRANは機械語プログラミングから我々を解放する素晴らしい言語であった。現在では，FORTRANよりも一段と高級かつ自然な言語でプログラムを作成できる
- FORTRANプログラムを実行する場合にも，またもや，そのプログラムを受付に渡すだけだった。しかし，その閉ざされた扉の向こうには，真のFORTRANマシンが存在しなかったことは現在では明かである

２０　抽象的なLISPマシン

- John McCarthy率いるMIT研究チームは，再帰関数とデータ構造を使用して，直接かつ自然に計算できるようにLISP言語と抽象LISPマシンを設計した。当時，現実のLISPマシンは存在しなかった
- LISPでは，機械語プログラミングからの解放度が，FORTRANよりもさらに大きかった。LISPの方が，概念的にはるかに高水準にあり，効果的な形式論理であるAlonzo Churchのラムダ計算法に直接基づいており，しかもそのきわめて抽象的な論理計算法の使用可能なバージョンにかなり近いものだった
- 後に現実のハードウェアでLISPマシンが実際に構築されたが，皮肉なことにSunワークステーションや他の汎用計算機で効果的にシミュレートされた抽象的なLISPマシンとの競争に生き残ることができなかった

2 1　理想的な標準形の機械

- たとえばラムダ計算法や，Curryの組合せ論理などの他の書換え体系は，暗黙の抽象的「リダクション」機械である
- 標準形の機械の状態は形式言語の式である
  状態　＝　式
  遷移　＝　書換え
  終端　＝　書換え不能　＝　標準形
- そのような標準形の機械を使用するには，式を入力としてキータイプし，標準形の等価式を出力として受け取る
- これは，記述型または宣言型のすべての計算のパラダイムである

が「どこかそのあたりに」ある，という幻想をユーザは抱いていました。

これが幻想であることはもちろん周知の事実でしたが，おそらく大きなメモリをダンプすることによってコンパイルされたプログラムをデバッグするという，実際的ではあるが（現在ではおそらく）評判のよくない慣習を除いて，このことを意識することは，ほとんどまたは全くありませんでした。

プログラムは，一定の抽象的な形式体系の規則に従って作成された後，プログラミングマニュアルで紹介されました。そして，そのプログラムが，それらの抽象的な形式体系の現実(またはシミュレートした）ハードウェアで，実際に高速で実行できたという事実はあまり重要ではありませんでした。

論理プログラミングのあらゆる側面を，この同じ視点から考えることができます（スライド21)。実際の計算の実用性の基礎になるのは，抽象的な書換え機械です。この機械は，直接取り扱っているような感覚で使用できるだけでなく，また基本的には形式論理体系に等しいものです。

当然のことながら現在でもこの傾向は続いており，その傾向は一段と強くなっています（スライド22)。プログラマは，実際にはどのハードウェアで実行しているのか気にしないばかりか，それさえも知らないことがしばしばあります。プログラマは，現実の機械ではなく抽象的な機械を扱っているのです。

かつてLandinは，現代のすべてのプログラミング言語は見かけ上は基本的にラムダ計算法でしかなく，これは遠い将来においても変わらないと明快に指摘しました（スライド23)。今

2 2　インタプリタとコンパイラ

- ある機械を別の機械上でシミュレートするというソフトウェア技術により，抽象機械を見かけ上現実の機械として直接に扱うことが可能になった
- これは「仮想の現実」の形式である
- 今日のFORTRAN,LISP,Prologなどのユーザは，自分のプログラムをほとんどどすべての現実の機械上で実行できると考えてよい。ユーザは，「抽象機械が存在するという幻想」を支える現実のハードウェアが何であるか，というような疑問を無視してもよい。

２３　Peter Landinの論文

THE NEXT 700 PROGRAMMING LANGUAGES,1964

すべてのプログラミング言語は，表面的にはラムダ計算法に他ならない。単に，構文によってひと味違うラムダ計算法でしかない。

- Landin固有のSECDマシンは，ラムダ計算法に関する高度な標準形の機械である。この機械は，標準形に還元される式の構造を分解および再組立てするのに，４つのスタックを使用する

２４　計算機科学と計算機工学

- 計算機科学は，抽象計算機と抽象的な計算の論理をテーマとして扱う論理学である。抽象機械はいつでもシミュレートできるので，その動作を見るためだけなら，そのハードウェアを構築する必要は全くない
- 一方，計算機工学は，現実の機械の構築と必要なハードウェア技術の向上に関する学問である
- この学問は，これまでに数々の奇跡を生み出してきているが，今後も奇跡の誕生が期待できるようなあらゆる兆候を示している

日では，Landinの論文を一般化すれば，プログラミング言語は見かけ上は論理体系であり，これからもそうであると言えるでしょう。

抽象機械と現実の機械とを区別した結果は，教育機関や研究機関の管理様式にまで及んでいます。計算機科学と計算機工学の区別が，一般的になっています（スライド24)。

人工知能それ自体においては，現実の人間と抽象的な人間との間に見られる同様な相違点が，人工知能の分野の解明に役く立ちます。スライド25は，Marvin Minskyの著作からいくつか引用したものです。Minskyは，人工知能にとって論理はあまり重要でないという見解を表明しています。

Minskyは，論理とは人工知能にとって重要な数多くの原則や概念の１つであるという見解を示していますが，この点については誰でも賛成するに違いありません。人工知能の使命が，実際の人間に見られる知能的な思考や動作を再生することであるならば，論理は，実際に人工

２５　論理と心理学

- Marvin Minsky,1985：
- 論理がどこか間違っているというつもりはない。「一般的な推論はおおむね論理に基づいている」という仮説に反対しているだけだ
- 論理は，物事を１つにまとめるために我々が長い間蓄積してきた各種の有益な方法のほんの一部である
- Minskyは，「我々の本質的なプログラミングの知識は抽象化と論理である」という点には賛成するだろう
- Turingの人工知能に関する基本的公理とは，他の人間，および自分たち自身に関する本質的な知識は，それと同じ特徴を持っているということである。これが，かの有名なチューリングテストの背景となる考え方である

知能の一部でしかないと思います。論理は，心が実行できることが何であるかを，論理学者が遡及的に再構築した結果に基づいています。しかし，心は，実際にはそのようには動かないと思います。心がどのように動くのか，本当はまだ知られていません。Minskyが論じるように，論理は，我々がどのように思考するのかについては解明してくれません。事実にあてはまらない，非常に人工的な思考モデルでしかないのです。ご存知のように，この考えに対しては盛んに賛否両論が交わされています。McCarthyは，自然ではそうなっていないとしても，論理をもっと概念の中心に据える知的な思考エージェントを持つことが可能であると現在でも主張しています。この主張の長所は，それをテストできることと，将来実際にテストされるという点です。最終的には，McCarthyの主張が正しかったかどうかがわかるのです。それが実現されるまでに，あと数十年待てばよいでしょう。

スライド26は，実際の人間と抽象的な人間との区別が実際に非常に便利な方法論的な区別であり，それが人工知能研究と認知科学研究においてなされ，書かれてきたものの多くを説明するということを示しています。一方で，自然が実際に知能のある生物をどのように創造してきたのかを理解し，説明しようとすることと，他方で，適切で使用可能な概念や技術を使用して知能的な生物を人工的にどのように作り上げるのかについて理解し説明しようとすることとの違いを明確にしておく限り，議論が紛糾することはありません。たとえそうであっても，人工的な産物に知能を与えようとする試みにおいてさえ，最良の方法はできるだけ自然を模倣することであると信じたいという研究者はいると思います。

スライド27は，いわゆるエキスパートシステムの場合，おそらく人工的な技術と非自然な方法を使用する「人工的な」エキスパートシステムの方が，実際の人間の専門知識に基づくエキスパートシステムよりもすばらしい働きをすることを意味しています。

最後に，論理と計算，および論理と人工知能のそれぞれの将来の関係を非常によく特徴づけていると思われる引用（スライド28）を紹介します。

Martin Davisは，計算論理学では著名な先駆的研究者であり，1950年代初期から論理プログラミングと自動証明の分野にきわめて大きな影響力を及ぼしています。

認知科学者であるHelmut Schnelleは，知能のある生物体の形式的ではあるが「自然な」論理的再生を実現するために，(セルオートマタ

２６　抽象的な人間と実際の人間

- McCullogh とPittsの表記法は，生物学的なニューロンの抽象的モデルを意図しており，そこには完全な脳および神経組織が，そのような要素の適切な回路網として人工的に再生可能である，という暗黙の主張が隠されている
- 人間の知能と人工知能の差はまだ果てしなく大きい。GodelとPostは，満足できる数学的知能に関する理論は，有限ではなくかつ創造的な推論を考慮に入れなければならないと考えていた。そのような理論を打ち立てる，新たなTuringの誕生を期待すべきなのか
- Turingの明らかな人間の限界性の主張，神経回路網が有限オートマタとして動作可能であるというMcCullogh とPittsの発見，および計算機能力の急激な向上によって，計算機を中央神経組織の働きのモデルとして，また知能の動作のモデルとして考えることが広く一般的に受け入れられるようになった

Robin Gandy,1988

27　論理とエキスパートシステム

- エキスパートシステムは，人間の知識の非常に狭く限られた範囲の部分集合モデルである。
- 高水準の専門知識（等式解決，医療診断，または質量分析に必要な専門知識など）の方が，低水準の技術（自転車やタクシーの運転など）に比べて論理的規則や公理の集合として形式化しやすいのは皮肉なことである
- 人間のほとんどの認知技術は，直観的かつ潜在意識的である
- 専門家にインタビューしても，専門家が使用している直観的な方法を明示的に表現できない場合には，その専門家の専門知識を知ることはできない
- したがって，将来の自然エキスパートシステムは，かつて期待されていたほど一般的ないし効果的なシステムにはならないと思われる
- 人工エキスパートシステムを効果的なものにするためには，(Stillerの完全に超人的なチェス終盤プログラムに見られるように)，人間そのものに基づく必要はない

28　論理とその将来

- Martin Davis,1984：
- 論理と計算の結合は，今後もきわめて重大である。普遍性，すなわち種々の異なるハードウェアを作る代わりに，単一の汎用目的の装置のプログラミングによってその機能を実現するという可能性は，今後も適切であり続ける
- Helmut Schnelle,1988：
- Turingの業績は，von Neumannに貴重な出発点を与えた。Turingは，彼のモデルは人間の動作を論理的かつ実際に理解するのに十分であると考えていたようである。それとは対照的に，von Neumannは，生物体のアーキテクチュアと構成を洞察することが最も重要であり，少なくとも実行可能性の証明が原理的に不十分であっても，実際的な理解が必要な場合にはその洞察が重要であると考えていた

の自己再生の理論の中で）von Neumannの試みを詳細に研究しました。Schnelleは，TuringやvonNeumannとは異なるアプローチでこの問題に取り組み，本日の講演で我々が関心を持ってきている抽象と現実との違いと基本的に同じものを見つけました。

それでは，私の講演のポイントを要約してみましょう（スライド29)。

論理が計算機科学の中心であると信ずる理由を挙げてきました。また，論理は人工知能の重要な要素ではあるが，それが人工知能の中心をなすわけではないと考える理由についても述べました。時がたてば，きっと論理は今以上に計算機科学において重要な役割を占めているでしょう。人工知能では，論理は，心理学，神経科学，言語学など関連する他の多くの学問の中の1つにすぎません。計算機科学並びに計算機の歴史に登場する傑出した人物は，Turingとvon Neumannという2人だけであると述べました。この2人の研究者は，どちらも最初から計算機分野で主導的な立場にあったが，この2人の中で科学的に重要な役割を果たしたのはTuringの方です。Turingという名前は，計算機科学全体を言い表す唯一の名前です。人工知

29　結論

- 論理は，計算機科学と人工知能の両分野における主要なテーマの１つであり，これまでもずっとそうであった
- 計算機と計算機科学の発展の歴史を見てみると，論理と技術は互いに有効に協力している
- しかし，一般的な見方とは裏腹に，論理と技術の提携においては，常に論理が主導権を握ってきた
- 計算機科学の将来では，論理と技術の提携が今後も続くだけでなく，今まで以上に緊密な関係になるが，論理の方がこれまで以上に一段と重要性をおびてくる
- 一方，人工知能の分野には論理が関係しているだけでなく，心理学，神経科学，言語学，ロボット工学，認知科学などが関係している。数え上げればまだあるが，このような他の多くの分野も，論理に比べたらその重要性は小さい
- 一般的に計算機科学では，TuringとvonNeumannは歴史上他に類を見ない研究者であるが，この２人の中ではTuringの方が第１人者である
- 人工知能では，Turingはユニークな役割を果たしている

能の研究に関しては，Turingが，他の研究者をはるかに引き離して独走していたことは間違いありません。

したがって，1992年６月23日は，Turing生誕80周年記念を思い起こすのにふさわしい日です。第五世代コンピュータシステムプロジェクトの大フィナーレである，このすばらしい会議とそれが我々に示すものは，Turingの生誕記念に最もふさわしいだけでなく，Turingの注目すべき科学的貢献と彼が創り出しかつ第１人者として発表した概念を記念するものであると思います。Turingが生きていれば，きっとこの機会を温かく見守ってくれただろうと思います。

ありがとうございました。

〔質疑応答〕

質問：あなたの講演はすばらしかったと思います。計算機科学における論理の役割を理解することに関して，１つ意見があります。我々は，おそらく論理の役割と，対照的な哲学の合理論と経験論の役割を比較しているのかもしれないということです。論理は，計算機科学という学問の合理的な側面です。論理プログラミングは，論理推論という点ですべての物事を合理的に表現しようとします。しかし，実際の世界は経験に基づくので，実際の世界がどのように構成されているか理解するには，論理以外の経験に基づく手法を使用しなければならないかもしれません。たとえば，計算機科学では，オブジェクト指向プログラミングは世界のモデル化に対して経験的なアプローチを採用していますが，論理プログラミングは合理的アプローチを採用しています。アプリケーションを行うときには，計算機科学と人工知能の分野で論理を超越する必要があると思います。

Robinson：はい。その意見に賛成です。それは，基本的にMinskyが人間の動作を理解するというケースで我々に教えてくれていることであると思いますが，もっと一般的にすべての自然現象や自然の過程にも全面的に適用することができます。自然界で見られる事象は，おそらくあまり論理は関係していないでしょう。しかし，ここで強調しておきたいこと，そしてあ

なたの類推には出てこないことは，論理は人工的な産物（自然界では発生しないものであると定義する）において，一段と重要な役割を果たすかもしれないということです。我々自身は，人工的に物を設計構築しますが，そうすることによって自然が自ら生み出してきた概念や技術に限定されるわけではありません。Herb Simonの言葉を借りれば，人工科学において，論理はもっと中心的な役割を果たすことが可能であるし，実際に現在その通りであるし，これまでもその通りであったし，また将来もその通りであるということです。これは，私が論文の中で強調したい点です。計算機は自然界には存在せず，我々が創り出した物です。それによって，論理を主な指針となる概念として使用し，いわばそれを合理的にすることが可能なのです。