コンピュータと共に歩んで

アントニー・ホーア

本日、この素晴らしい会場におきまして、私が受けた教育、経験、ならびにそうしたものが私の人格形成にどのように影響を与え、さらには先端技術分野におけるキャリアにどのように結びついていったかを振り返る機会を与えていただきましたことは、私の大きな名誉であります。私はこの記念講演において、私の人生哲学に大きな影響を与えた、現在、ならびに過去の知的指導者に敬意を表したいと思います。彼らの多くも、京都賞の受賞者としてこの晴れのステージで栄誉を受けておられます。今回、名だたる受賞者の皆様の末席に私をお加えいただけることを、大変光栄に思います。

1947年、私はカンタベリーのキングズ・スクールという中等学校に入学しました。それからの5年間、私は風光明媚なカンタベリー大聖堂の構内に住み、その気高い塔を眺め、鐘の音を聞きながら学生生活を送りました。

最初の年には、英語、英文学、歴史、フランス語、ラテン語やギリシャ語の古典語、それに初級、上級の数学の2コースなど、合わせて10教科を選択しました。そのために週に1レッスン、趣味程度にやる以外には、理系の教科をやっている時間はなくなってしまいました。私は数学が大好きだったのですが、次年度に継続履修できるのは2教科のみという決まりがあり、当時は、ラテン語やギリシャ語が得意な男子生徒は、聖書またはフランス語を副教科にするというのが一般的でした。

毎週、古典作品の一節の英訳や、さらにこちらのほうが難しいのですが、英語の散文をラテン語、ギリシャ語へ翻訳する宿題が出されました。もちろん、古典語の文法を勉強しなければなりませんでしたが、英語のそれよりもはるかに複雑でした。というのも、すべての名詞と形容詞に性があり、それが単数、複数でそれぞれ5つから6つの格に変化するのです。格変化にはパターンが4つから5つありましたが、例外もたくさんあって、それをまた一つ一つ覚えなければなりませんでした。ラテン語やギリシャ語の文章では、形容詞を修飾する名詞の性、数、格に一致させなければなりません。文法に従って、こまかくチェックしていたつもりなのですが、それでも間違いはなくなりませんでした。余談ですが、こうした間違いは、先生たちにはあまりにもばかばかしくてすぐに分かるため、「粗忽者」と呼ばれていました。こうしたこともあってか、後にコンピュータとの通信、制御に用いる人工言語の設計、実行の研究をするようになった時、私は人工言

語の文法をできるだけ簡単かつ規則的なものにすることに心を砕きました。また、インプットされたプログラムがルールに反している場合には、あの頃の私の先生よろしく、コンピュータがすぐさまそれを見つけ、確実に報告がなされるようにしました。

翻訳や作文の作業は楽しいものでした。大事なことは、ある言語で書かれた文章の意図、内容、思考の流れ、文体をよく理解したうえで、構文や言葉の持つニュアンスも異なる別の言語を用い、それと同じメッセージを再構築することです。そして私は、バランスとコントラスト、展開とバリエーション、テーゼとアンチテーゼに関する修辞法の簡単なルールを学びました。この記述は、それ自身が、その中で説明されている3種類のコントラストの展開というルールをバランスよく、かつコントラストを付けて例証したものとなっています。この文は何度も慎重に書き直したものですが、最善の結果というものは、それを選択する過程でいくつも他を試してからでないと得られないものです。後に論文を書いたり、大学で教えるようになってからも、自分なりにこうした作業を楽しみながら行ってきました。難しいのは、アイデアをできるだけ分かりやすく、かつ覚えやすく説明したうえで、自分の主張、それに対する反論を公平に、しかもできるだけはっきりと提示することです。私が指導してきた学生には、彼らが自分たちの研究成果をうまくまとめられるように、こうした技術を伝えることを心掛けてきました。

学校では古典語を取らざるを得なかった私ですが、幸運にも数学に対する関心は失っておらず、特に不確実な知識を説明する手段として、またトランプで勝つチャンスを増やす手段として、確率に関心を抱いていました。

私は学校の図書館でランセロット・ホグベンの『百万人の数学（*Mathematics for the Million*）』という本を見つけました。それは大変興味深い内容でした。私は、試験で再現が求められる知識の集合体としてではなく、面白い問題を解明するために用いる道具として数学を趣味として勉強できたことを幸運に思います。もしも古典ではなく数学を専攻していたら、数学に対する愛情を失っていたかもしれません。少なくとも、プログラミング理論やコンピュータ科学の研究に役立つ、興味深い数学の諸理論を学ぶことはなかったでしょう。

また、学校の図書館で、バートランド・ラッセルの『西洋哲学史』という分厚い本にも出会い、初期ギリシャ哲学から近代あたりにかけての部分を読破しました。こうして私は哲学にも目覚め、今もそうした関心を失ってはいません。ラッセルの哲学書の中で特に面白かったのは、ホワイトヘッドとの共著による『プリンキピア・マテマティカ（数学原理）』という3冊にも及ぶ作品の簡約版、『数理哲学序説』でした。この本が素晴らしいのは、数学的論法をシンプルな論理形式へと置き換え、数学のコンセプトを、単一か

つ均一、そして見た目にはシンプルな集合というコンセプトとして提示している点です。

それから何十年も経ってオックスフォード大学の教授になった時に、私は数学的論法を用い、すでに確立された数学理論に過度に頼ることなく、コンピュータプログラムの正当性を証明することを思い立ったのです。ジャン=レイモンド・アブリアルをリーダーとする共同研究で、私たちは、すでにラッセルが数学的には多くの点で適切であることを証明していた集合理論が、プログラムのスペックだけでなく、設計、実行の正当性の証明にも適していることを明らかにすることに成功したのです。この発見を商品化する作業の第一歩は、IBMの協力を受けて踏み出されました。

1952年、私はオックスフォード大学マートンカレッジに入りました。そこを選んだのは、いわばホーア家の伝統でした。私の父もマートンカレッジで学び、専攻も私と同じでした。最初の2年間は、ラテン語とギリシャ語を履修し、詩作を修めました。ホメロスやウェルギリウスのほとんどの作品だけでなく、キケロ、ユウェナリス、ホラチウス、エウリピデスの代表作も勉強しなければならなかったため、読むスピードも速くなりました。

時間が空いた時には、数学の論理や基礎などをやり、哲学面での好奇心を満たしていました。入学した年には、学校の宿題を済ませた後の寝る前の時間に、1996年に京都賞を受賞することになるウィラード・ヴァン・オーマン・クワイン［photo 1］の『数理論理学』という教科書を片手に、古典語、数学のクラスで知り合った2、3人の友人と夜遅くまでコーヒーを飲みながら話し込んでいました。その時の教科書は今でも持っています。この本は、数学の証明とは、チェスの棋譜にも似た、短い行が順番に並んだ文章のようなものであることを私たちに教えてくれました。ある証明が正しいかどうかは、チェスやラテン語の文法のように、一度に一行、一手、一語ずつ検証していけば確かめることができました。こうしたことは、むしろ証明の主題、ゲームの戦略、文章の意味などが分かっていないほうがうまくいくものです。証明の場合、一つ一つの行は、公理をコピーしたものか、その前の行の特殊なケース、もしくは「モダス・ポネンズ」とラテン語で呼ばれているルールに従い、その前の二行の内容を踏まえたものでなくてはなりません。こうした論理的証明構造の正式手法は、現在、コンピュータに数学の応用作業をアシストさせたり、数学的証明をさせたり、あるいはそれをチェックさせようというあらゆる試みの基礎となるものです。もちろん、公理を引用した行は証明する必要がなく、その正当性を確認することはできません。公理はあくまでも「自明」として、よりくだけた言い方をするなら、数学者が選んだゲームのルールとして受け入れなければなりません。公理というものは、それ自身から発展した数学の各分野の根本となる概念を定義する

ものです。

それから25年後、私は研究者として初めて世に問うことになった論文、「コンピュータプログラミングのための公理的基礎」の中でこうした証明手法を採用しました。当時私は、友人でもある偉大なコンピュータ科学者、エズガー・W・ダイクストラの理想に触発されていました。それは、「コンピュータプログラムの設計は、自身の正当性を示すものでなくてはならず、また設計担当者もそうした義務を果たしていかなければならない」というものでした。ロバート・フロイドの提言もあって、私はこの「義務」をコンピュータプログラミング言語の設計者にまで拡大適用しました。こうすれば、プログラマーが必要な証明を構築していく作業も楽になるはずです。したがって、優れたプログラミング言語の意味を記述する良い方法は、すべての証明のベースになっている公理を記述することになります。公理の単純さ、そしてその適用が何を基礎としているかは、プログラミング言語の設計の質を客観的に判断する際の基準となります。後にニクラウス・ヴィルトは、この考え方に沿って、あの有名な教育言語PASCALを設計しました。

オックスフォード大学で私が選んだコースは、リタライ・ヒューマニオーレイスと呼ばれる人文学で、昔ながらの伝統に則したものでした。3年目からは、古代史、哲学という2教科が新たに加わり、ツキディデスやタキツゥス、プラトンやアリストテレスなど、こうした学問の祖を直接学びました。また、デカルト、ヒューム、カントなど、近代ヨーロッパの哲学者についても学びました。私がコンピュータの機能について初めて知ったのは、哲学コースでのことでした。1937年にアラン・チューリングが発明した、かの有名なチューリングマシーンでした。現代の哲学者では、科学的仮定は、その誤りを立証するべく、念入りに設計された実験にその意味があるというドクトリンを提唱した、第8回京都賞受賞者カール・ポパー[photo 2]を学びました。こうした彼の教えから、私は後年開発することになる分散形計算ネットワークの理論構築のヒントを貰い、プログラムの意味を、きわめて直接的に、プログラムが誤りを犯す可能性がある道筋をすべて明らかにする観察記録をまとめたものと定義しました。

その当時、イギリスの若者には2年間の兵役が義務づけられていましたが、私の直後に廃止されました。私はラテン語とギリシャ語ができたので、英国海軍でロシア語を学ぶことは簡単に許可されました。(叔父が海軍大佐というコネも効いたようでした。)私のクラスは、標準的な訓練を受け、射撃場で銃の撃ち方を学び、駆逐艦や掃海艇にも2、3度日帰りで乗船しましたが、てきめん船酔いに襲われました。しかし、泊りがけで海に出るということはありませんでしたし、もちろん、軍事行動らしいことは一切ありませんでした。

ロシア語の指導は主に陸の上のロンドン大学スラヴ地域研究学部の分室で行われました。クラスでは、ロシア語の文法に関する基礎知識を徹底的に叩き込まれました。ロシア語の文法は、ラテン語やギリシャ語に負けないほど複雑で、名詞、形容詞が6つの格に語形変化します。ただ、幸運なことに、性は2つしかありませんでした。私たちは、文法ミスを犯すと、即座に学校から追放されるだけでなく、将校の身分を失い、他の新兵と一緒に厳しい訓練を受けさせられることになる、と脅されていました。私は以前に勉強したラテン語やギリシャ語の作文を思い出して、一つ一つの性と格を、習った法則に照らし合わせて確認するようにしましたが、それでもこれまた同様、見落としがしばしばありました。

古典語の学習と一番違ったのは、話し言葉としてロシア語を学んだことでした。口に出して話すとなると規則を頭で考えてから適用する時間などありません。ところが、驚いたことに、しばらくすると、子供がロシア語を習得するのと同様に、文法などあまり気にしなくても、たいていは正しいロシア語が喋れるようになっていました。また、数字に関する信じられないほど複雑なルールでさえ、すぐになんでもないように思えるようになりました。どうしてそのようになったのかはその当時も、今も謎のままです。

ロシア語のコースを終えた私は、学生生活はもう十分だと思い、社会に出て働いてみることにしました。しかし、人文系の学生にありがちな、会計職や行政職、管理職に進むのではなく、科学技術関係の職に就くことを希望していました。そうした仕事を始めるには技術的資格が必要だと考えていた矢先に、オックスフォード大学に戻って学部生を1年やれば統計学の修了書が貰えることに気付きました。ただ、自分がコースについていけるだけの数学的知識を十分持っていることをオックスフォード大学の統計学の教授陣に伝えるのには苦労しました。ですから、後に私がコンピュータ学のコースに学部生を受け入れるようになってからは、人文系、特に言語系のバックグラウンドを持つ志願者には特別の同情を示したものでした。

オックスフォード大学には、他の学部の講義にも自由に出てもよいという伝統があり、私はやがてホー・ワン教授の数理哲学コースに顔を出すようになりました。教授は初期のIBM704コンピュータを使って、ラッセルとホワイトヘッドの『プリンキピア・マテマティカ』の冒頭の9章に出てくる証明をひとつ残らずチェックするためのプログラムを書きあげたばかりでした。1957年当時としては、これは特筆すべき偉業であり、チューリングの理論、初期の論理学者の目的、17世紀のドイツ人哲学者ライブニッツの夢をも事実上実現するものでした。この時の講義で私は、その後日米両国で哲学者として素晴らしいキャリアを積み重ねることになる石黒ひでという日本人学生に出会いました。

同じ頃、私は、コンピュータプログラミングのコースにも出ていましたが、後にも先にもこれきりとなりました。指導してくださったのは数値解析が専門のレスリー・フォックス教授で、20年後に私が数理科学の教授としてオックスフォード大学に戻った時には、奇遇にも学部長になっていらっしゃいました。先生はマーキュリーオートコードという、原始的ではありますがレベルは高い、当時ひとつしかなかったプログラミング言語を教えてくださいました。私は最初のプログラムとして、子供のじゃんけん遊びのような2人の競技者による零和ゲームの確率値を求めるためのおおまかな手順のコード化を試みました。ちょうどその頃読んでいた、フォン・ノイマンとモルゲンシュテルンによるゲーム理論に関する本に感化されたのです。私は自分で作ったプログラムが読み込まれていくのを見て大変感動しました。プログラムは翻訳された後に数秒間実行され、私が考えていた6つの数字が出力側から出てきました。しかし、その数字が正しいものかどうかは分かりませんでした。プログラムにチェック機能を入れていなかったのです。もし入れていれば、簡単に分かったでしょう。現在、こうしたチェック機能は「表明（assertions）」と呼ばれていますが、私はプログラムの正当性を証明する際のベースとして、こうした機能を薦めるようになりました。

オックスフォード大学の統計学コースも最後の学期となった頃、私は学寮の掲示板でブリティッシュ・カウンシルの広告を見つけました。ブリティッシュ・カウンシルというのは、イギリスと各国の文化、教育面での交流を促進することを目的とした公的機関です。彼らは、ソ連の大学と年間20名の大学院レベルの学生の交換留学に関して合意に至り、その募集をしていたのです。ところがよく読んでみると、締切りは次の日に迫っていました。勢いもあって、私は24時間以内に書類が用意できれば応募しようと考えました。そして運よく締切りに間に合わせることができました。

最大の動機は、新たなチャレンジへの期待感でした。1960年当時、外国人のソ連への渡航はまだ難しく、それ以前に学生のグループがソ連に入国したことはありませんでした。ソ連に行けば、180度正反対の政治体制を体験、実感することができるのです。ヨーロッパ諸国はルネッサンス期にギリシャ、ラテンの影響を再度受けることになるのですが、ロシア文化は、それとは一線を画する形で進化してきました。また、私はロシア語を使い、磨きをかけたいと思いました。さらには確率論の基礎理論を勉強したいとも思っていたのですが、ロシアには、その道のパイオニアであるアンドレイ・ニコラビッチ・コルモゴロフという数学者がいました。そこで私はモスクワ国立大学の彼のいる学部に学生登録をしたのです。

最初は苦労しました。外国語で講義を受けるのは、1時間でも大変な集中力が要る

のですが、モスクワでの講義は2時間でした。また、教室の椅子も大変硬い、座り心地の悪いものでした。それでもロシア語での講義にも徐々に慣れてきた私は、椅子の上に敷くクッションを買おうとモスクワで1日がんばったのですが、結局うまくいかず、両親に頼んで送ってもらいました。こうして当座の問題が解決し、ようやく学生の本分に取り組む準備が整ったわけですが、実は数学の勉強が一番大変でした。特に、新参者が大学院生レベルで抽象的手段の理論を一から始めるのは並大抵ではありませんでした。しかし、幸運なことに、大学の図書館にハルモスの素晴らしい本を見つけることができました。そこには、後年、私がコンピュータプログラミング言語の研究に応用される変域理論の研究を行うことになった時に再び出会うことになる、集合論、格子理論、近似・極限の分析などが紹介されていました。

私はモスクワで、その後の私の方向性を決定づけた1通の手紙を受け取りました。差出人は、ロンドン近郊のテディントンという町にある国立物理学研究所という、英国政府の中心的研究所でした。私は、上級研究員としてロシア語から英語へ自動翻訳を行うコンピュータのプログラミングに関する新たなプロジェクトへの参画を求められたのです。

この手紙を口実に、私は、数学の研究から少し距離を置いて、当時のソ連における機械翻訳研究プロジェクトについて少し調べ始めました。もちろん英語からロシア語です。『機械翻訳』というロシア語の専門誌のバックナンバーを読み返し、幾人かの執筆者からも直接話を聞くことができました。この調査をまとめたレポートは、私にとって初めての学術論文となりました。ロシア語で書いたので、ロシア語のタイプライターを友達から借りました。しかし、ソ連のコンピュータを実際に目にすることはできませんでした。おそらく、当時まだ外国人に漏らしてはならない機密事項だったのでしょう。

機械翻訳について調べているうちに、モスクワのレーニン図書館に行く機会がありました。第4回京都賞基礎科学部門の受賞者、ノーム・チョムスキーによる統語論研究に関する本があったのはそこだけでした。チョムスキーの言語統語論は、コンピュータを使って文法をチェックするのに用いられている規則を正確に描写したものです。彼の重要なアイデアは、回帰的定義、つまり、実際に定義される言葉それ自身のコピーを含んだ定義です。例えば、英語の文法では、「従属節」は動詞といくつかの名詞、さらに他の従属節をも含むものと定義することができます。これは、アリストテレスによる定義に反するものですが、注意して使えば、高い確度で意味を伝えることができます。プログラミングの世界では、チョムスキーのアイデアを最初に取り上げたのはジョン・バッカスで、続いてピーター・ナウアが国際算法言語であるALGOL60の設計に用いてい

ます。それ以降、チョムスキーの考え方は、他のプログラミング言語の設計だけでなく、計算機言語学、コンピュータ科学全般に至るまで広く影響を与えました。

機械翻訳で最初にしなければならないことの一つは、原始言語で書かれた文章の各語の辞書的な意味を見つけ出すことです。辞書はアルファベット順に長い磁気テープの上に記憶されており、最後まで読むのはもちろんのこと、スキャンにさえ数分間かかることがありました。したがって、文章中の単語を一つ一つ個別にスキャンしていく作業は大変非効率的でした。そこで、最初に単語をアルファベット順に並べ替えることによって、1回のスキャンで全部を調べ、テープ読取り機のヘッドを通過する時に辞書にあった意味を一つ一つピックアップするという、より効率の高い手法が考え出されました。私が整列アルゴリズムであるクイック・ソートを開発したのは、非常に小さな主記憶装置しかついていないコンピュータでこうした問題を解決するためでした。

モスクワ留学もそろそろ終わろうかという頃、またしても思いがけない手紙が届きました。今回の差出人は、すでに海軍を退役し、イギリスの科学機器メーカーの業界団体で代表を務めていた叔父でした。叔父は製品の展示、即売を行う展示会をモスクワの中心部で開く準備をしており、私に、イギリスからの出展社や講演者と、地元の人たちや展示会に訪れる科学者や潜在顧客との通訳の仕事を依頼してきました。

その時の出展社の一つにエリオット・ブラザーズという会社がありました。同社のコンピューティング事業部は、ボアハムウッドという町で研究者向けの小型コンピュータを作っていました。彼らは展示会で、エリオット803という当時最先端だったコンピュータのデモンストレーションを行い、私もほとんどの時間を彼らのブースで過ごしました。展示会が終わる頃、彼らのコンピューティング事業部長が、モスクワまでコンピュータを運ぶのに使ったバンが空いているので、イギリスまでただで乗せて帰ってくれると言ってくれました。私にはロシア語やロシア人、官僚に関する知識もあるので、帰りの道中、ドライバーの役に立つのではないかということでした。そして、同時に、正社員として働いてみないか、という誘いも受けました。

イギリスに戻った私は、就職活動の手始めとして、国立物理学研究所の面接に臨みました。この時、私はアラン・チューニングの指導のもとで設計された、かの有名なACEコンピュータのプロトタイプを初めて目にしました。キャビネットは巨大な広間を埋め尽くすほど大きく、機能ユニットは真空管で作られており、2、3ダースある即時アクセスレジスタは水銀で満たされたタンクの中で音波遅延として記憶されていました。また、主記憶は磁気ドラム上にあり、電源は自らの電動発電機で起こさなければなりませんでした。値段も現在の貨幣価値で数百万ドルはするものでした。研究所側が提示して

きた雇用条件について詰めていく段になって、上級科学職員という肩書を用意するというのは、誤解だったことが判明しました。確かに、科学職員というのはそもそも畑違いで、私は技術者階級に分類されるので、本来なら実験職員となるべきでした。さらに、私に理系の学位がないことが分かると、彼らは契約ベースでしか雇用できないと言い、理系公務員として終身雇用の望みはないという説明をしました。結局、私はこの話を断ったのですが、後になって、この決して魅力的とは言い難いポストを私が断ったことに担当者が驚いたと聞きました。しかし、断った最大の理由は、私がコンピュータによる自然言語翻訳の可能性を見限っていたことにありました。

結局、私はプログラマーとしてエリオット・ブラザーズ社に入り、モスクワで初めて見た803コンピュータ向けのライブラリプログラムを10進マシンコードで書くことになりました。プログラムの中でも頻繁に実行される部分の速度を上げ、限られた記憶装置を効率良く使えるようにするために、そうした部分を書いたり、書き直したりするのは、ある種チャレンジでしたが、本当に楽しみながら行うことができました。当時、唯一気になっていたのは、上司が私よりも4歳年上であるだけで、その上の上司もまだまだ若かったことでした。私は、コンピューティング拡大の最初の波を逃してしまったと思っており、昇進するチャンスはしばらく来ないものと考えていました。当時の私は、めまぐるしく変化を遂げるエキサイティングな時代をその後迎えようとは知る由もありませんでした。私が現役の間にコンピュータの速度が百万倍、価格は千分の一程度で、信頼性が千倍、そして消費電力も千分の一になることなど、考えてもいませんでした。ましてや、サイズも百万分の一となる一方で内部記憶容量が数千倍になり、地球上で数億台が普及するとは夢想だにしませんでした。こうしたハードの費用効果の天文学的上昇に伴い、新たに生み出されたパワーを効率的に活用するプログラマーが求められるようになりました。しかし、40年ほど前に私が自らの経験から学び、その後の講義、研究活動のテーマとなったプログラミングの原則は、その重要性を失っていないどころか、むしろその価値は増しているほどです。

エリオット社でプログラミングをしている時以外には、モスクワで思いついた整列アルゴリズムについて引き続き考えを巡らし、それがどれほど速いものなのかという問いに対する答えを捜していました。私は、比較や交換の必要平均数を支配する連立差分方程式を紙に書き出していました。ある日の日曜日、私はソファに寝そべりながら、なんとはなしにそれで遊んでいました。そして、「0 = 0」という明白な事実をまたしても証明したことに気付き、突然手を止めました。こうしたことは、素人数学者にはよくあることです。こういう時には、前の作業に立ち返って、慎重にチェックし直すのが一番です。

その時もそうしてみたのですが、驚いたことにミスはありませんでした。そう、私はその等式を解く正しい公式を見つけることに成功し、整列プログラムの平均速度を求めたのです。計算してみると、私が望んでいたものと同じくらい速いものでした。この時の経験がきっかけとなって、1962年、「クイック・ソート」と題した論文をブリティッシュ・コンピュータ・ジャーナルに寄稿することになったのです。

10年後、私は第12回京都賞先端技術部門の受賞者、ドン・クヌースの招きを受け、スタンフォード大学を訪れました[photo 3]。教授は「クイック・ソートの平均ケース複雑性に関する私の分析に勇気づけられ、後に研究室で、より強力な分析技術を応用し、同算法の整列時間の統計的分布についてより多くのことを明らかにすることに成功した」と語ってくれました。

エリオット社に就職してから6カ月が経ち、同社の新しい高速コンピュータ向けに、新たにハイレベルなプログラミング言語の設計を行うという役目が私に与えられました。大変幸いなことに、ピーター・ナウアの国際算法言語であるALGOL60のレポートをちょうど入手でき、私たちはその部分集合の実行を目指すことにしました[photo 4]。そしてさらに幸運なことに、以前のモデル向けにオートコードの実行を終えたばかりのプログラマー、ジル・ピムをプロジェクトのメンバーに迎え入れることができました。その後まもなく、私たちは結婚し、現在も幸せな生活を送っています。

1961年という黎明期においては、プログラミング言語用に変換系を書く作業は、自然言語の翻訳用に変換系を書くことに比べて明らかに簡単というわけではありませんでしたが、幸運にも、当時、会社には専門誌が豊富に揃っており、私は毎週のように拾い読みをしていました。そこで私は、プログラミング言語の実行に関する特集記事を掲載した、計算機協会の『コミュニケーション』1960年4月号を見つけました。中でも最も関心を引かれた記事は、第4回京都賞先端技術部門受賞者、ジョン・マッカーシーによるS式の帰納的関数についてのものでした。彼は、純粋に機能的なプログラミング言語であるLISPの最初のバージョンを、驚くべきほどクリアに描写していたのです。それは再帰的プログラミングの技術を取り入れたもので、それ自身の定義と同時に、再帰呼び出しの助けを借りてプログラムのサブルーチンを定義することができました。私は、マッカーシーがそれ自身にデータとして提出された他のLISPプログラムを翻訳により定義するために書いたLISPプログラムのシンプルさに驚かされました。それは、アラン・チューニングが自身で発明した機械を使って同様のタスクを行うために考え出した労作ともいえる構造よりもはるかにシンプルでした。さらに、LISPバージョン1.5のプログラミングマニュアルの最初の例にも感銘を受けました。それは、ホー・ワンが以前にラ

ッセルとホワイトヘッドの『プリンキピア・マテマティカ』の冒頭9つの章に出てくる定理をチェックするために使ったアルゴリズムを驚くほどシンプルにした表現式だったのです。

それからさらに10年が経った、スタンフォードに客員として招かれていた年の夏、私はジョン・マッカーシーの人工知能研究所で研究を行う機会に恵まれました。マッカーシーの機能プログラミング、データ構造、公理、非決定性に関する理論は、こうした分野における研究のその後の進歩を促すきっかけとなった歴史的なものです。

マッカーシーをはじめとする研究者による関連論文を徹底的に分析した私は、エリオットのALGOL変換系の設計、書き込みを可能にする鍵を見つけました。それはチョムスキーの著作で初めて出会い、LISPやALGOL60自身でも再会した再帰法というコンセプトでした。私は、自分の整列アルゴリズムの公式バージョンをプログラミングする際、それを大いに利用しました。

ALGOL60の実行を目的とした私たちのプロジェクトは順調に進んでいました。1年ほどが経ち、私は納品できる日もそう遠くないのではないかと考え、そのことを上司に報告しました。ちょうどその直後、土壇場でIBMに乗り換えてオーダーをキャンセルしてきた客に翻意を促すために、コンピューティング事業部の上の者がニューヨークに行くことになりました。その客はエリオット803上で走るALGOLコンパイラの可能性に感銘を受け、再度私たちの製品へとくら替えしてくれました。この売上回復で、私たちの小さなプロジェクトは、社内でも大いに評判となりました。ところが、以前には全くなかった、納期という厄介なものまでいただいてしまいました。幸い、納期には間に合わせることができましたが、効率的な動作環境でコンパイラの使い勝手を上げるために、かなりの後作業が必要となりました。

ALGOLコンパイラの思いがけない成功に気をよくしたエリオット社は、さらに速いコンピュータ向けに、さらにパワフルかつ効率的な動作環境の設計、実行を行うという、野心的なプロジェクトに乗り出しました。私は設計と、約50名のプログラマーの監督を任され、その実行を目指しましたが、3年後には失敗を認めざるを得なくなり、顧客との約束を全く実現することはできませんでした。この時の失敗は、それ以前の小さな成功よりも、その後の私のキャリアに多くの影響を及ぼすことになりました。具体的に言うと、それから後に行った研究では、OSの書き込みをプログラム言語交換系の書き込みと同じくらい簡単にすることを目指すようになりました。

さて、とりあえずは体制の建て直しが求められ、私はその任務をも任されました。立ち直りには2年かかりましたが、それから後は会社も、ハードの技術やマシンアーキテク

チャの進歩に歩調を合わせていかなければならない、より大きく、より速いコンピュータの設計に関心を向けるようになりました。こうした目的を推進するため、私は開発部の技術マネジャーからコンピューティング研究部の主席エンジニアに任命されました。私たちは少数精鋭でキャッシュメモリや、現在バーチャルメモリとして知られているページングシステムなど、アーキテクチャのイノベーションに関する研究を行いました。しかし、私たちはこれからのコンピュータの売上を左右するのはソフト、特にOSであることを承知していました。会社はひとつもソフトの開発に成功しておらず、私自身OSがどういったものかさえも知りませんでした。そのため、私はケンブリッジ大学の数理科学研究所を訪れ、そこで稼動していたタイタンコンピュータとタイタンOSについて学びました。このソフトは同大学のスタッフや教授が、あるコンピュータメーカーと協力して書いたもので、第8回京都賞先端技術部門の受賞者であるモーリス・ウィルクス[photo 5]の指導の下、ロジャー・ニーダムがプロジェクトの指揮をとりました。私はキャッシュメモリに関する彼らの功績、ならびにOS全般の構造、機能に関する視点など、多くを学んでケンブリッジを後にしました。

1967年、エリオット・ブラザーズ社は自分のところよりも大きなライバル社に買収され、新しいコンピュータを設計するという私たちのプロジェクトは立ち消えになってしまいました。翌年、会社はさらに大きな別の会社に吸収されてしまいました。当時の記憶はあまり定かではないのですが、私はベルファストのクイーンズ大学でコンピュータ科学の教授を募集しているのを見つけ、自分でも驚いたことにそれに申し込み、さらに驚いたことに、採用されることになったのです。

こうして1968年に私の大学教授としてのキャリアが始まりました。私はライフワークとして、コンピュータプログラムの正当性という根源的なテーマを選びました。こうした研究は、往々にして基礎的かつ純粋で、長い時間を要するものとされていましたが、そんなことは気になりませんでした。この研究は、私が教授職から引退する頃にならないと産業界で応用されることはないだろうと考えていましたが、事実、その通りになりました。昨年、私は大学の定年を迎え、世界的ソフト会社、マイクロソフトのケンブリッジ研究部門で上級研究員となりました。モーリス・ウィルクスは、ケンブリッジにある私たちの新居の近くに住み、ロジャー・ニーダムは、私の直属の上司となります。

本日は皆様の前で、私の人生、ならびに私が生きてきためまぐるしい時代についてお話をさせていただくことができ、大変嬉しく思っています。これまで私は、幾度となく大胆な方向転換をしつつも、偉大な哲学者、理論家の教え、コンピュータ学の先駆者による発見、さらには企業での成功、失敗から得た経験を常に糧とし、新規かつ重要

な技術分野における基礎理論を発見することができたことがお分かりいただけたと思います。本日この会場にお越しのお若い方の教訓にすることができることがあるとすれば、それは私を含め他人の轍を踏むな、ということです。自分で選んだ道を開拓し、その時々で最も関心のある方向を選んでください。他人から与えられたカリキュラムや、平々凡々としたキャリアのほうが居心地がいいという方々には、ご幸運をお祈りします。しかし、自分の興味やチャンスを追い求めた末、他人とは違う方向に行ってしまったとしても、一般的な規範から逸れることを恐れないでください。若い頃に勉強したり、経験したことを常に活かすことを考えていれば、予想だにしていなかった場面でそうしたものが活きてくるものです。この広い世の中に、同じ背景、関心、人格を持った人は二人といません。人知のさらなる進歩、様々な文化の持続的更新、そして私たち人類の究極の繁栄と幸福を確かなものとしていくためには、こういった自由に広がる多様性を育み、開発していかなければならないのです。

photo 1

photo 2

photo 3

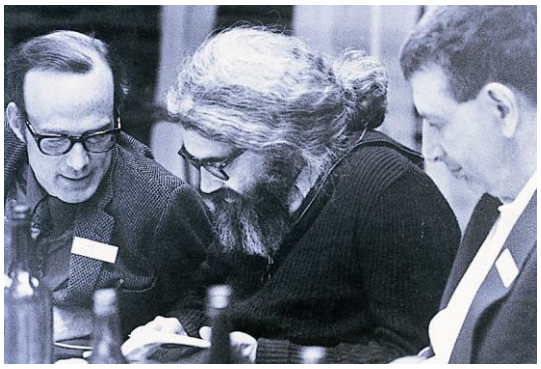

photo 4

photo 5