題名	アルゴリズムのパワーと限界
Title	The Power and Limits of Algorithms
著者名	リチャード・マニング・カープ
Author(s)	Richard Manning Karp
言語 Language	日本語・英語　Japanese, English
書名	稲盛財団：京都賞と助成金
Book title	Inamori Foundation: Kyoto Prize & Inamori Grants
受賞回	24
受賞年度	2008
出版者	財団法人 稲盛財団
Publisher	Inamori Foundation
発行日 Issue Date	7/1/2009
開始ページ Start page	110
終了ページ End page	143
ISBN	978-4-900663-24-7

アルゴリズムのパワーと限界

リチャード・マニング・カープ

本日は講演の機会を与えていただき、大変光栄に存じます。私の生まれ育った環境、受けてきた教育、コンピュータ科学者の道に私を導いた出来事、そしてこれまでに私が下してきた様々な決断などについてお話ししたいと思います。私は、コンピュータの発明後に大人になった最初の世代の一人です。学校教育を受けていた時期は、科学分野への大幅な支援強化策をアメリカが打ち出していた時に符合します。私は二つのすばらしい大学と優れた民間の研究所で仕事をする機会に恵まれました。私より前の世代の研究者は、これほど恵まれた環境を得ることはありませんでした。あと数年早く生まれていれば、おそらく今とは全く違った、そして間違いなく今より満足感に欠けるキャリアを歩んできたであろうことは想像に難くありません。

家族

私の家族は、マサチューセッツ州ボストンにあるドーチェスターという町に住んでいました。写真はドーチェスターの典型的な通りの冬景色です（Fig. 1）。私たちはこれとそっくりな通りに住んでいました。ドーチェスターは人種によって居住地区が分かれていて、私たちの街区は、私の家族を含めてほとんど全員がユダヤ系でした。住まいは小さなアパートで、私は三人の弟や妹と一緒に一部屋を使っていました。

私が成長する過程で一番大きな影響を与えてくれたのは、母でした。母は高校卒業後に就職しましたが、その後、夜間コースに長年通い、57歳でハーバード大学から学位を授かりました。子どもの教育を常に一番に考え、私の学校の成績が良かったことをとても誇りにしてくれました。また、家計は楽ではありませんでしたが、富そのものに価値があるのではない、ということを母は教えてくれました。

父は、ハーバード大学の出身でした。上品で慎ましく、近所の人からも広く慕われていました。元々は医学部を志望していたようでしたが、金銭的な理由でそれを断念して学校で数学を教えるようになり、最終的には学区の教育長まで務めました。教員の給料だけでは十分ではなかったので、父は家庭教師をしたり、食料品店でアルバイトをしたり、百科事典のセールスをしたりと、様々な副業により一家を支えてくれました。

まだ小さかった頃、父の授業を見学したことがあり、その時のことは今でもよく覚えています。父は黒板にフリーハンドで、ほとんど完璧な円を描くことができました。後に、私だけではなく、弟のデイビッドや妹のキャロリンが教師になったのも決

The Power and Limits of Algorithms

Richard Manning Karp

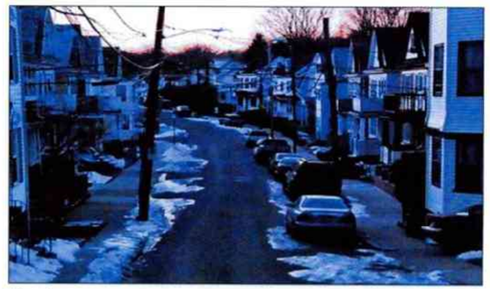

Fig. 1

It is a great honor to address you today. I would like to tell you about my background and education, and about the events and decisions that have shaped my career as a computer scientist. I belong to the first generation that came to maturity after the invention of the digital computer. My school years coincided with enormous growth in the support of science in the United States. I have been able to work in two great universities and a great industrial research laboratory. No previous generation had access to such opportunities. If I had been born a few years earlier I would have had a very different, and undoubtedly less satisfying, career.

My Family

My family lived in Dorchester, a section of Boston, Massachusetts. Figure 1 shows a typical street in Dorchester in wintertime that is a quite similar to the street where my family lived. The city was divided into ethnic neighborhoods, and nearly everyone in our neighborhood was Jewish, as I am. We rented a small apartment, and I slept in the same room with my three younger siblings.

My mother was the strongest influence on my development. She went to work after graduating from high school, but eventually earned a Harvard degree at age 57 after taking evening courses for many years. Her top priority was the education of her children and she took great pride in my achievements at school. We had little money, and she taught us not to value wealth as an end in itself.

My father was a graduate of Harvard University. He was a humble and

して偶然ではないと思います。デイビッドは社会学の教授で、キャロリンは英語の教諭です。私たちは父をお手本にして育ったのです。

両親はすでに亡くなってしまったのですが、母の一番下の妹である叔母のフィリスが、今日ここに来てくれたことを私はとても嬉しく思っています。叔母は私の子守をしてくれました。また、一緒に遊んだり勉強を教えてくれたり、本の読み方や自転車の乗り方も教えてくれました。

残念ながら、妻のダイアナと息子のジェレミーは、私と一緒にここに来ることができませんでしたが、彼らの存在なくして私の人生はこれほどまでに豊かで満ち足りたものとはならなかったでしょう。ちなみに、ダイアナはサンフランシスコ近代美術館でガイドを、ジェレミーはブランダイス大学で優等学生として経済学と心理学を学んでいます。

子どもの頃の思い出

私はおとなしい、静かな子どもで、よく本を読んでいました。古代ギリシャや『マルコ・ポーロの冒険』、カウボーイとアメリカン・インディアン、そしてアメリカの独立戦争についての本などが、今でも印象に残っています。詩も好きで朗読を楽しみました。また、地元の野球チーム、ボストン・レッドソックスの大ファンでした。海水浴に行ったり、自転車に乗ったりするのも好きでした。祖父が経営していた近所の食料品店で配達のアルバイトをし、冬はそりに商品を載せて運んだりもしました。

学校の勉強は良くできましたので、他の子どもたちより早く学校に進み、飛び級もしました。その結果、クラスメートよりも年齢が下で、習字や図画工作といった、手先の器用さを必要とする教科では他の子どもたちに遅れをとっていました。今でもこうした作業には自信がなく、研究所でもずっと苦手にしてきました。

得意な科目はラテン語と数学でした。ラテン語が好きだったのは、文法の構造が論理的だったからです。10歳の時、「ボストン・クイズ・キッズ（Boston Quiz Kids）」というラジオ番組に何度か出演したことがあるのですが、そこでは数学パズルを得意にしていました。その後、13歳で平面幾何学に出会い、そのパワフルかつエレガントな証明に魅せられた私は、仮病を使って学校を休み、家で幾何学の問題を解いたりしていました。

gracious man who was widely admired in our community. He had wanted to go to medical school but could not afford it financially, and became a school mathematics teacher instead, rising to the position of head of a school district. Teachers were poorly paid, so to support our family he took on a variety of extra jobs such as tutoring private students, working in a grocery store, and selling encyclopedias.

I have fond memories of visiting his classroom as a youngster. I remember that he was able to draw a nearly perfect circle on the blackboard freehand. It is probably no accident that two of my siblings, my brother David and my sister Carolyn as well as I, became teachers. David is a Sociology professor and a Carolyn is a teacher of English. We were following in our father's footsteps.

My parents have passed away, but I am delighted that my Aunt Phyllis, my mother's youngest sister, is here today. She was my babysitter, playmate and mentor, and taught me to read and ride a bicycle.

My wife Diana and son Jeremy cannot be here today, but I would like to acknowledge how they have enriched my life and contributed to my well-being. Diana is a docent at the San Francisco Museum of Modern Art, and Jeremy is an honor student at Brandeis University, majoring in Economics and Psychology.

Childhood Memories

I was a rather meek and quiet child and spent a great deal of time reading. I particularly remember books about ancient Greece, the travels of Marco Polo, cowboys and Indians, and the American Revolutionary War. I liked poetry and enjoyed reading it aloud. I was a great fan of the local professional baseball team, the Boston Red Sox. I enjoyed swimming in the ocean and riding my bicycle. I also worked in my grandfather's neighborhood grocery store, making deliveries to customers, carrying the groceries on a sled in the wintertime.

I excelled in academic work, but because I started school early and then skipped a grade I was much younger than my classmates and lagged behind them in manual skills such as penmanship, art and carpentry. Lack of confidence in my manual skills has continued throughout my life, and I have always had a resistance to doing laboratory work.

At school my favorite subjects were Latin and mathematics. I enjoyed learning Latin grammar because of its logical structure. At age ten I appeared several times on a radio show called the Boston Quiz Kids, where my strong point was solving

科学者を志望

私がハーバードに入学した頃は、科学分野の中では物理が最も人気があったのですが、自分は特に物理に向いているとは思えず、数学の方がぴったりくる感じがしていました。ところが、本格的に数学をやり始めたのは良いものの、当時のハーバードには、後にフィールズ賞やノーベル賞を受賞するような特段の才能を持った学生が在籍していたため、私の影は薄くなってしまいました。これにはいささか気落ちしましたが、確率論の先生であるハートリー・ロジャーズ教授が、そうした私を熱心に励ましてくれました。当時、私が特に関心を抱いていたのは、離散数学や離散確率といった学問を、野球の戦略やギャンブルのシステム、ゲーム理論などでの意思決定に応用することでした。当時、こうしたテーマは数学界の主流ではなく、数学担当の教授の多くもほとんど関心を示しませんでした。しかし、こうしたテーマは最近ますます大きな注目を集めるようになっています。

この頃にはコンピュータに秘められた大きな潜在力が明白になりつつあり、私はハーバードの計算研究所で博士号を取ろうと決心しました。1957年のスプートニク・ショックで、技術分野にもブームが訪れていました。夏休みに興味深いアルバイトに就くことができた私は、そこでちょっとした成功を収め、ひょっとしたら自分もひとかどの人物になれるのではないかという思いが芽生えてきました。

博士号を取ってすぐに、幸運にも数理科学の研究者としてIBMのトーマス・J・ワトソン研究センターに採用された私は、優秀な仲間と一緒に研究活動に没頭する自由を与えられました。当時、まだ数学者として一人前とは言えなかった私は、IBMで私の指導担当であった数学者のアラン・ホフマン氏との対話から多くのことを学びました。ホフマン氏は、組合せ数学について多くを語ってくれただけではなく、私の能力を信じてくれていました。また、ホフマン氏自身が数学の美しさを深く楽しんでいる様子を見聞きし、また、そうした喜びを分かち合うことへの氏の熱意に接する中で、教育・研究に対する自分なりのアプローチが固まっていきました。さらに、ランド研究所、プリンストン大学、米国国立標準局といったところで、意思決定問題への離散数学の応用に先駆的な仕事をしたすばらしい数学者と知り合うこともできました。

IBM在籍中は、ニューヨーク市内の複数の大学の夜間クラスで教えていました。そこで私は、講義の準備として新しい教材の予習をすることや学生との交流など、教えるという行為に関わるすべての事柄が好きでたまらない自分に気づきました。大学教

mathematics puzzles. At the age of 13 I was introduced to plane geometry and was wonderstruck by the power and elegance of mathematical proofs. I recall feigning sickness in order to stay home from school and solve geometry problems.

Becoming a Scientist

I entered Harvard College at a time when physics was the most glamorous scientific field, but I found that I had no special aptitude for it, and mathematics came to me much more easily. But as I advanced in mathematics I was overshadowed by a few exceptional mathematics students, including a future Fields Medalist and a future Nobel Prize winner. This was somewhat discouraging but, at the same time, I received strong encouragement from Prof. Hartley Rogers, whose course in probability I had taken. I was particularly attracted to the applications of discrete mathematics (finite mathematics) and discrete probability to the application of problems of decision-making, ranging from baseball strategy to gambling systems to game theory. Such topics were not in the mainstream of mathematics at that time, and were of little interest to most of my mathematics professors. They have become much more prominent in recent years.

The great potential of computers was becoming clear by this time, and so I decided to pursue a Ph.D. at the Harvard Computation Lab. Sputnik in 1957 led to boom times in technical fields. I was able to obtain interesting summer jobs, had a few successes, and began to feel that I might amount to something.

Upon receiving my Ph.D., I was fortunate to obtain a position in Mathematical Sciences at IBM Research, where I was given great freedom to pursue my research in collaboration with excellent colleagues. My mathematical education was inadequate at that point, and I benefited greatly from conversations with my mentor at IBM, the mathematician Alan Hoffman. He taught me a great deal about combinatorial mathematics, showed confidence in my ability, and shaped my approach to teaching and research through the example of his deep enjoyment of beautiful mathematics and his zest for communicating it. I also became acquainted with a whole school of wonderful mathematicians, at places like the Rand Corporation, Princeton University and the National Bureau of Standards, who pioneered the application of discrete mathematics to problems of decision-making.

During my years at IBM I taught a number of evening courses at universities in New York City. I discovered that I loved everything about teaching—the

授は自分の天職であるという考えが頭をもたげてきたのは、ちょうどこの頃でした。

アルゴリズムに対する思い

また、この頃には、アルゴリズムの研究が自分のライフワークになるであろうと考えるようになっていました。アルゴリズムとは、特定の問題を解くための体系的な計算手順のことを言います。最も身近なアルゴリズムというのは、学校の算数でも取り上げられるようなものですが、実はアルゴリズムは至るところに存在します。例えば、情報処理に関するありとあらゆるアプリケーションの中心となるのがアルゴリズムです。検索クエリの処理であれ、ネットワークを介したメッセージのルーティングであれ、ネットオークションであれ、アルゴリズムはその中核をなしています。電子商取引の基礎となるすべての暗号化プロトコルの中心にもデータの暗号化を行うアルゴリズムが存在するのですが、そのアルゴリズムの中にも、ある数が素数であるか否かを判定するための別のアルゴリズムが存在しています。そうしたアルゴリズムは数行で書き表すことができますが、非常に緻密であり、それがなければ現在のようにインターネットを使ってビジネスを行うことはできなかったでしょう。

第12回(1996)京都賞を受賞した偉大なコンピュータ科学者、ドナルド・クヌース氏は、計算プロセスの構造を注意深く考えることに大きな美的歓びを見出すようなタイプの人々を、世間に認めさせることに成功しました。自分もそうした種類の人間であると自覚した瞬間のことは、今でもよく覚えています。それは、私の学友だったビル・イーストマン氏が、割り当て問題を解くための、いわゆるハンガリアン・アルゴリズムを示した時のことでした。ハンガリアン・アルゴリズムは、(得られる利益を最大にするように) 労働者に仕事を割り当てるという、古典的な割り当て問題を解くためのアルゴリズムのことです。このアルゴリズムでは足し算と引き算以外の計算は一切行わず、そのエレガントなまでの簡素さに、私はすっかり心を奪われてしまったのです。しかも、このアルゴリズムを使えば、少ない計算ステップで解を見つけることが可能なのです。

アルゴリズムに最初に要求されることは、その正確さです。つまり、なすべきことがきちんとなされるということです。それに加えて、効率的でなくてはなりません。アルゴリズムの効率やコストを測定する際に我々が主に用いるのは、実行にかかる時間、荒っぽく言えば、基本ステップの数です。こうした物差しに照らし合わせてみて

mastery of new material in preparing a lecture, the contact with students—and the idea began to grow on me that I was destined to be a professor.

A Feeling for Algorithms

I also came to realize that my life's work would be the study of algorithms. An algorithm is a systematic computational procedure for solving a specified problem. The most familiar algorithms are perhaps the ones for arithmetic that we learned in school. Algorithms are everywhere. At the core of every information processing application there is an algorithm. Whether it's processing search queries or routing messages through a network or conducting auctions over the Internet, algorithms are at the heart of it. At the core of all the cryptographic protocols that underlie e-commerce, there's an algorithm for encrypting data, which depends in turn on an algorithm for testing whether a number is prime. That algorithm can be written down in a few lines, but it's very subtle and without it we wouldn't be able to conduct business over the Internet the way we do.

Don Knuth, a great computer scientist who received the Kyoto Prize in 1996, has called attention to a breed of people who derive great aesthetic pleasure from contemplating the structure of computational processes. I still recall the exact moment when I realized that I was such a person. It was when a fellow student, Bill Eastman, showed me the so-called Hungarian Algorithm for solving the Assignment Problem, a classic problem of matching workers with jobs. I was fascinated by the elegant simplicity with which the algorithm converged inexorably upon the solution, using no arithmetic operations except addition and subtraction.

The first requirement of an algorithm is that it should be correct. It should do what it's supposed to do. Beyond that, it should be efficient. The primary way we measure the efficiency or cost of an algorithm is by the time it takes to execute—roughly speaking, the number of basic steps. I've spent a lot of my time devising algorithms that are efficient by that measure.

My first attempts to develop an algorithm came as a schoolboy. My father was a school principal and every fall before school started he had to create a schedule of classes that satisfied a variety of constraints. Certain classes could only be taught in certain rooms and at certain times. Certain pairs of classes could not conflict with one another. Certain teachers were unavailable on certain days, and so on. He got out a stack of index cards representing the different classes, laid

効率的と呼べるアルゴリズムを考案するために、私はかなりの時間を費やしてきました。

私が初めてアルゴリズム作りに挑戦したのは、まだ子どもの頃でした。当時、父が校長をしていた関係で、毎年秋に新しい学年が始まる前に、様々な制約を同時に満たした時間割を作らなければなりませんでした。この教室、この時間でないとだめ、という授業や、一緒のコマに設定してはいけない組合せがあったりするだけでなく、この曜日はだめ、という先生もいらっしゃいました。父は授業名を書いたインデックスカードの束を取り出して台所のテーブルに広げ、シャッフルをし始めました。残念ながら、私は大した力にはなれませんでした。莫大な数の理論上可能な時間割から、こうした条件すべてを満たすものを見つけ出す作業は、困難を極めました。このような作業は、ちょうど干草の山から一本の針を探し出すようなものです。この問題の攻略法として私が唯一思いついたのは、試行錯誤による探索作業でした。

この種のパズルのような問題、すなわち、莫大な数のパターンや配列の中から条件に合うものを探し出すという問題は、すべての科学分野、そして人間の活動のあらゆる領域で見ることができます。これらの問題を解く方法は、社会資源の効率的な活用に欠かせないツールとなります。例えば、工場における作業計画、コンピュータチップ上の部品の配置や配線、インターネット上でのメッセージや電力網を介した電気のルーティング、ヒトゲノムの配列決定などです。私の研究は、こうした組合せ探索問題に対する有用なアルゴリズムを構築して、その基本的な限界を明らかにすることにより、それらすべての分野に貢献しています。

複雑性

組合せ探索問題の解の中には、驚くほど簡単なものもあります。道路網のある地点から別の地点まで移動する際の最短ルートを計算するための効率的なアルゴリズムは、衛星技術を利用したカーナビゲーションシステムの基盤技術となっています。しかし、ほとんどの組合せ探索問題では、このように簡単な解は見つかりません。問題の記述の複雑度が増すにつれて、指数関数的に増大する計算リソースを求めているようでもあります。

一般に、商業、物理科学、工学、そして人文科学や社会科学の分野で生じる、時間割問題のような組合せ問題を解くのは非常に困難です。それぞれの授業を時間割に当

them out on the kitchen table, and started shuffling them around to contract a schedule. I was not able to offer much help. Finding a schedule that met all these conditions, among the vast number of possible schedules, was like searching for a needle in a haystack. The only method I could find to attack my father's scheduling problem was trial-and-error search.

Puzzle-like problems of this kind, which involve searching through a vast set of patterns or arrangements to find one that satisfies a set of conditions, arise in every scientific field and in every domain of human activity. They are essential tools for the efficient exploitation of society's resources. Examples include scheduling jobs in a factory, arranging components and wires on a computer chip, routing messages in the Internet or electricity in a power grid, and sequencing the human genome. My work contributes to all these fields both by creating useful algorithms for such combinatorial search problems and by clarifying their fundamental limitations.

Complexity

Some combinatorial search problems are surprisingly easy to solve. For example an efficient algorithm to compute the shortest route from a starting point to a destination in a road network underlies the satellite navigation systems in our cars. But most combinatorial search problems seem to elude such easy solutions, and to require exponentially growing computational resources as the description of the problem becomes more complex.

In general, combinatorial problems like the class-scheduling problem, which arise in commerce, in physical sciences, in engineering, and even in the humanities and social sciences, are very difficult to solve. There are a vast number of ways the objects can be arranged—assigning classes to time slots, for example—but only a few of these arrangements meet all the requirements. So the question is, are there shortcuts? Some problems have clever, efficient algorithms, but for most of the problems that come up, there seems to be no radical shortcut. You can solve small examples, but as the size grows—for example, the number of classes to schedule gets bigger—the running time of even the best algorithm tends to explode. That is, it roughly doubles every time you increase the size of the problems just a little bit.

At IBM during the 1960s I worked on a number of applied combinatorial problems in computer design and operations research and became painfully aware

てはめていく作業と同じように、配列の方法は膨大になります。しかし、すべての条件を満たしてくれるものは一握りしかありません。そこで、近道はないのか、ということが問題となります。一部の問題に対しては、巧妙かつ効果的なアルゴリズムがありますが、現実的な問題のほとんどでは、そのようなアルゴリズムは存在しないように思えます。小さなサイズの問題であれば解けるかもしれません。しかし、サイズが大きくなり、例えば、割り振らなければならない授業の数が増えたりすると、たとえ最高のアルゴリズムを用いたとしても計算の実行時間が爆発的に増える傾向にあります。すなわち、問題のサイズをほんの少し大きくしただけで、大雑把に言えば、実行時間は倍々に増えていくのです。

1960年代のIBM在職中は、コンピュータデザインやオペレーションズリサーチにおける組合せ問題の応用を数多く扱っていましたので、計算におけるこのような組合せ爆発という問題には何度も苦い目に遭わされました。米国国立標準局にいた、慧眼な研究者、ジャック・エドモンズ氏との対話に刺激を受けた私は、こうした実行時間の指数関数的増大は、本質的に不可避な問題かどうかについて考え始めました。

さらに私は、計算の効率に対する基本的かつ本質的な限界を研究するコンピュータ科学の一分野である、計算複雑さの理論の進展についても注意を払っていました。物理学にエネルギー保存則や熱力学の法則、ハイゼンベルクの不確定性原理などの基本的法則があるように、計算複雑さの理論においても、計算効率に関する基本的かつ本質的な限界の確立に向け、現在、研究が続けられています。計算複雑さの理論によって、アルゴリズムの計算時間における組合せ爆発が説明できるか、という問いに到達したのは半ば自然の成り行きでしたが、その頃の私にはこうした問題を解明できるだけの用意はまだできていませんでした。

教授職に

1968年、私は、カリフォルニア大学バークレー校から教授職のオファーを受けました。IBMで研究を行うのは大変な名誉でしたし、そこでの研究は私自身にとって、コンピュータ科学者、そして数学者としての成長には欠かすことのできないものでした。この身を天職に捧げる時が来たのです。

1960年代後半のバークレーは、当時の社会変革の最前線にありました。ベトナム戦争の激しさは頂点に達しており、私が着任して間もなく、カンボジア侵攻に対する

of combinatorial explosions in computation. Stimulated by conversations with Jack Edmonds, a visionary researcher at the National Bureau of Standards, I began to wonder whether such exponential growth in running time might be inherently unavoidable.

I was also following developments in computational complexity theory, the branch of computer science that studies the fundamental, inherent limitations on the efficiency of computation. Just as physics has fundamental laws such as conservation of energy, the laws of thermodynamics and the Heisenberg uncertainty principle, computational complexity theory strives to establish fundamental, inherent limitations on the efficiency of computation. It was natural to ask whether computational complexity theory could explain combinatorial explosions in the running time of algorithms, but the time was not yet ripe for me to attack this question.

Becoming a Professor

In 1968 I was offered a professorship at the University of California at Berkeley. It had been a great privilege to be a researcher at IBM, and my work there had been crucial for my development as a computer scientist and mathematician, but now it was time to fulfill my destiny as a teacher.

Berkeley in the late 1960s was at the forefront of social change. The Vietnam War was at its height. Soon after I arrived, there were major protests against the Cambodian invasion. I remember teaching some classes in my home because protesting students had closed the campus, and paying bail to release a faculty colleague who was arrested for participating in an illegal protest march against the war.

Computer Science at Berkeley was also in turmoil at that time. The Electrical Engineering Department was home to a number of computer scientists, but several had split off to form a new department to serve the College of Letters and Science. As the rivalry between the two departments was unhealthy both academically and socially, the administration decided to fold the two groups into a single unit within Electrical Engineering. I was not very experienced in academic affairs having just moved to academia a few years before, but since I hadn't taken sides in the controversy I was chosen to lead this new unit. I wasn't a born administrator and I didn't stay in the job very long, but I played some role in quieting things down

大規模な抗議活動がいくつも起こりました。学生がキャンパスを封鎖してしまったので、自宅で何度か授業を行ったのを覚えています。また、反戦を掲げた不法抗議行進に参加したことで逮捕された同僚の釈放を求めて、保釈金を支払ったこともあります。

当時、バークレーのコンピュータ科学関連組織も大荒れの状態でした。コンピュータ科学者の多くは電気工学科を活動の拠点としていたのですが、そのうち何人かは文理学部の下で新しい学科を立ち上げるために、電気工学科を去りました。この二つの学科が張り合うことは学問的にも社会的にも健全とは言えませんでしたので、大学側はこれらを一つにまとめて電気工学科内に置きました。私はこうした騒動のほんの数年前に大学に移ってきたばかりで、いずれのグループにも与していなかったので、経験不足は承知の上で新しい組織の長に選ばれることになります。私は根っからの管理者タイプという訳でもありませんでしたし、また、在任期間も短かったのですが、事態を沈静化させるために一肌脱ぎ、新たな枠組みに対する承認を関係者から取り付け、再び一緒になって研究活動に邁進することを承知してもらったのです。

こうして学内の政治問題が一件落着したバークレーは、コンピュータ科学に関する一大学科を立ち上げる準備を開始しました。バークレーでは常に、理論コンピュータ科学を専攻する学生のコミュニティーがあり、皆、協調心と熱意に満ちあふれています。バークレーが理論面で成功を収めることができたのは、マニュエル・ブルム氏の存在に依るところが大きいと言えます。彼は、バークレーで私が最も親しくしていた25年来の同僚ですが、彼からは出来の良い学生だけではなく、すべての学生に敬意を持って接することを学びました。また最良の教授法とは、学生が自ら発見を行えるように導くこと、そして自身の研究に関しては、単に技術的な進展だけでなく、幅の広い、統合的なコンセプトを中心として研究を組み立てることの大切さを学びました。

教えることの喜び

教えることは、私のキャリアを通じて最大の喜びの一つです。他者の人生に影響を及ぼす行為としては、最も意義のあるものだと思います。まず、教材を一通り自分でマスターし、そこから首尾一貫した講義を組み立てる、という作業は本当に楽しいものです。こうした作業は、私にとっても大切な学びの場となりました。講義では自分

and getting people to accept the arrangement and start building something together.

Once the politics settled down Berkeley was poised to build a great computer science department. We have consistently had a thriving community of students of theoretical computer science, and there has always been a spirit of cooperation and enthusiasm among them. A major reason for the success of theory at Berkeley has been Manuel Blum. Manuel was my closest colleague at Berkeley for twenty-five years. Through his example I learned to have respect for all my students, not merely the strongest ones. I learned from him that the best way to teach is to lead students to make their own discoveries, and I learned to build my research around broad unifying concepts, rather than mere technical advances.

The Joy of Teaching

Teaching has been one of the greatest pleasures in my career, and possibly the most important way in which I have influenced the lives of others. I greatly enjoy the process of mastering a body of material and shaping it into a coherent lecture; it is the primary means by which I learn. Lecturing permits me to express my personality, share my experiences and opinions, and connect with each new generation of students.

I greatly value the opportunity to serve as a mentor and role model for talented research students. I have had 41 Ph.D. students. They have gone on to become leading professors and industrial researchers, found companies, and lead professional societies. They include the inventor of a famous algorithm for linear programming, the creator of new links between the fields of economics and computer science, the developer of a radical method of distributing data over the Internet, the designer of mechanisms that allow the resources of a computer network to be shared efficiently, and several who have made key contributions to computational molecular biology and served as my co-workers and mentors in that subject. I stay in touch with most of my former students, and enjoy witnessing the growth of their families.

My students asked me to formulate some principles of advice for pursuing their careers as theorists. This is what I told them. Understand what you are good at and what you like to do, and choose accordingly. In the words of Socrates, "Know thyself." Disregard the fashions of the day and search for new areas

の個性を表現することもできますし、私自身が経験したことや普段考えていることを学生の前で披露することもできます。そして、常に新しい世代の学生たちとつながりを持つことができます。

また、研究室の才能ある学生を指導し、手本となる機会もかけがえのないものです。私は、これまで41名の博士の誕生を見守ってきましたが、大学や民間の研究所において第一線で活躍する者、起業した者、学会を牽引する者など、そのキャリアは多岐にわたります。彼らの中には、線形プログラミングに関する有名なアルゴリズムを発明した者、経済学とコンピュータ科学という二つの異なる分野の架け橋となった者、インターネット上でのデータ配信に関する画期的な手法を開発した者、コンピュータネットワークのリソースを効率的に共有するメカニズムを編み出した者、そして計算分子生物学に重大な貢献を行い、この分野で私の研究仲間兼アドバイザーとなった者もいます。こうした元教え子たちとは今でも連絡を取り合い、彼らの家族が成長していく様を見るのを楽しみにしています。

教え子から、キャリア形成のための基本的なアドバイスを求められました。以下はその時に語ったことです。まず、自分が得意なものは何か、また何をするのが好きかをよく知った上で職業を選びなさい。ソクラテスの言葉を借りるなら、「汝自身を知れ」ということです。その時々の流行を追うのではなく、いままさに重要になろうとしている新分野の研究をしなさい。エキサイティングな問題を見つけるために、境界領域の研究動向に注意を払いなさい。自分を信じ、自分のセンスと判断に自信を持ちなさい。たとえゆっくりでも、進歩は必ずあるということを理解すべきです。遅々として進まないような状況でも、創造の過程を楽しみなさい。仲間との交際を楽しみ、アイデアの交換を惜しんではいけません。もし、本当に研究者としての資質に恵まれて生まれてきたならば、進歩がないような状況下でも、研究テーマの持つ美しさゆえに楽しめるようになるでしょう。

NP完全性

1971年、バークレーで一緒だったスティーブン・クック氏が発表した論文を読んだ私は、計算における組合せ爆発は避けられないとする問題に、計算複雑さの理論を応用することはできないものだろうか、と考え始めるようになりました。クック氏は、非常に多くの組合せ問題の様々なバージョンを含んだ、現在はNPと呼ばれる問

of research that are about to become important. To find exciting problems, look to the interfaces between disciplines. Believe in yourself and trust your own taste and judgment. Understand that progress comes slowly but that it will come with time. Enjoy the process of creation, even when progress is slow. Enjoy the company of your colleagues and be generous in sharing ideas. Perhaps the most important of these considerations is that if you are truly born to be a researcher you will enjoy what you are doing even on the days when you are not making progress, simply because of the beauty of the subject.

NP– Completeness

In 1971 a paper by Stephen Cook, a former Berkeley colleague, led me to think that computational complexity theory could be brought to bear on the question of whether combinatorial explosions in computation are inevitable. Cook defined a class of computational problems, now called NP that includes versions of a vast number of combinatorial computing problems. This class includes many famous problems, some of which have resisted efficient solution for 200 years going back to the time of the great mathematician Gauss. Cook showed that a certain problem in mathematical logic known as the Satisfiability Problem was as hard as any problem in this class. My experience with combinatorial search problems suggested to me that many well-known combinatorial search problems were just as hard as Satisfiability. Starting from the Satisfiability Problem I used a technique called "reduction" to build up a set of problems that are all of equivalent difficulty, and as hard as any problem in the vast class NP which includes virtually all the combinatorial search problems that we encounter. The problems that have this property of being the hardest in the class NP are called NP-complete. To show that a problem is NP-complete, all one needs to do is to show that, if you had an efficient algorithm for solving the new problem, you could also use that algorithm to solve some other problem already known to be NP-complete. Using this method of reducing one problem to another, I showed that 21 classic problems, arising in fields such as engineering, mathematics, natural science, biology and commerce, having no overt similarity in their descriptions, are NP-complete. The NP-complete problems are equivalent in the sense that if you could solve any one of them efficiently, you could solve all of them efficiently. By now thousands of problems have been proven to be NP-complete by means of the reduction technique. Many

題のクラスを定義しました。このクラスには有名な問題が数多く含まれています。中には、その起源を有名な数学者ガウスの時代に求めることができる問題もありますが、200年経った今でも効率的な解が得られていません。クック氏は特に、このクラスにおいて充足可能性問題として知られる数理論理の問題が困難であることを示しました。組合せ探索問題に取り組んだ経験を持つ私は、良く知られている組合せ探索問題の多くも、この充足可能性問題と同じくらい難しいのではないかと思うようになりました。そこで私は、充足可能性問題から始め、「還元」という手法を用いて、NPクラスの問題の中で解くことが同程度に難しい一連の問題を見出しました。NPクラスというのは、非常に大きなクラスで、ほとんどすべての組合せ問題を含んでいます。すなわち、NPクラスの中で解くのが最も難しい問題は、NP完全問題と呼ばれています。新しい問題がNP完全問題であることを証明する方法は簡単です。その問題を解くための効率的なアルゴリズムが既に存在するのであれば、そのアルゴリズムを用いることによって、NP完全であると分かっている他の問題も解くことができるはずなので、それを証明すればよいのです。このように、一つの問題を別の問題に「還元」するという手法を用い、私は、一見すると関係が全く無いように思える、工学・数学・自然科学・生物学・商業などの幅広い分野にわたる21の古典的な問題がNP完全であることを証明しました。これらのうち一つでも効率的に解くことができれば、他のものもすべて効率的に解くことができるという意味において、NP完全問題はすべて同等だと言えます。これまでに還元という手法によって、何千もの問題がNP完全であることが証明されました。今日、理論計算機科学の分野の専門家によって発表されている多くの還元法は、私が提案したものよりもずっと複雑になっています。現在でも毎日のように、新しい還元法やNP完全の証明が発表されています。ここにＮＰ完全問題の例を一つ示します（Fig. 2）。これはパッキング問題といわれる問題で、大きな長方形の中に大きさの異なる小さな長方形を互いに重なり合わないように配置する方法があるか否かを問うものです。このような例では、ある配置が与えられた時に、それが条件を満たしているかどうかを検証するのは簡単です。しかし、ＮＰ完全問題とはこのような検証をすることではなく、条件を満たす配置が存在するか否かを判定することなのです。ＮＰ問題の特徴は、解が与えられた場合、その正しさを検証するのは簡単だけれども、解そのものを見つけるのが非常に難しい点にあると思います。

of the reductions that specialists in the theory now produce are far more intricate than the ones I originally came up with. New reductions and NP-completeness proofs are published every day. Figure 2 illustrates one example: We see a solution to a particular instance of a NP-complete problem. The figure shows a large square into which non-overlapping rectangles of various sizes are packed. It is easy to verify that the solution is correct, but the NP-complete problem is not to verify a solution but to construct one: given the small rectangles, to find a way if possible to fit them into the big square without overlapping. So, it is a fundamental characteristic of the complexity class NP that solutions once given are easy to check, but may be very hard to find.

The next three cartoons illustrate three ways that a worker might explain to his boss that he has failed to solve an NP-complete problem. In the first one (Fig. 3), being ignorant of NP-completeness, he takes personal responsibility. He says, "I can't find an efficient algorithm. I guess I'm just too dumb."

For the second one (Fig. 4), he would have to know that P is unequal to NP, in other words, that NP-complete problems are difficult. This is currently an open mathematical problem. He would say in this case, "I can't find an efficient algorithm because no such algorithm is possible."

The third one represents our current state of knowledge (Fig. 5). The empirical evidence of multitudes of failed attempts strongly indicates that we will never find an efficient algorithm to solve every instance of an NP-complete problem, but theoretical mathematical questions are resolved by proofs, not by empirical evidence, and we seem to be very far from proving that this is the case.

There have been many advances related to NP-completeness over the years. For example, when I first worked on certain problems, I was unable to classify them. One of these problems is to test whether a given number is prime. But, just few years ago, it was proven this problem is actually in the class P. So, there is an efficient way of solving it.

Another development is based on a beautiful theorem developed around 1990 called the PCP theorem (PCP stands for Probabilistically Checkable Proof). The theorem leads to reductions showing that certain problems are not only hard to solve exactly if P is unequal to NP, but also are hard to solve even approximately.

Unfortunately, at this time, because we can not resolve the problem of P vs. NP, the field of complexity theory is filled with conditional theorems of the form, if

これからお見せする3つのイラストは、それぞれ、ある会社員が上司にNP完全問題の解法に失敗したことを報告する方法を示しています。Fig. 3の「彼」は、NP完全に関する知識を持ち合わせておらず、自分が失敗の責任を取ろうとして、「どうしても効率的なアルゴリズムが見つかりません。自分の能力が至りませんで…」と言います。Fig. 4の「彼」は、「PとNPは等しくない」と言います。すなわち、NP完全問題は難しいという、現在も未解決の数学的問題を知る必要があるようです。この場合、彼は、「効率的なアルゴリズムが見つからないのですが、そもそもそうしたアルゴリズムは存在しないはずです」と言うでしょう。Fig. 5は、現在、我々が持っている知識のレベルを示しています。おびただしい数の失敗により得られた証拠は、あらゆるNP完全問題を解く効率的なアルゴリズムは今後も決して見つからないであろう、ということを強く示唆しています。しかし、理論的な数学問題は、経験ではなく、証明によって解かれるべきであり、この推測が正しいと証明されるまでには、まだまだ時間がかかるようです。

NP完全問題に関連した、多くの発展がこれまでにありました。例えば、私が初めてある種の問題を取り扱った時には、これらがNP完全問題であるかどうか判別できませんでした。そのうちの一つは、与えられた数が素数かどうかを判定する問題でした。ほんの数年前のことですが、この問題はPというクラスに属することが証明されました。つまり、この問題を効率的に解くアルゴリズムは存在するのです。

別の方向の発展は、1990年頃に開発された美しい理論、PCP理論と呼ばれる確率的検査可能証明理論に基づいています。クラスPとクラスNPが等しくないという仮定のもとでは、PCP理論によって、ある種の問題は厳密に解くのが難しいばかりか、近似的にさえ解くことが難しいことを示す還元法が与えられます。

現在、計算複雑さの理論の分野や、理論計算機科学に関連した分野においては、「P対NP問題」、すなわち、二つの複雑性クラスが同じか同じでないかを解くことは残念ながらできていませんので、「PとNPが等しくないならば、次のようなことが成り立つ」という条件付きの定理が数多く存在します。ここで、2つの例を挙げてみます。最初の例は、PとNPが等しくないならば、情報伝達に有用な暗号化プロトコルは破られない、というものです。次の例は、PとNPが等しくないならば、ある問題は近似的に解くことさえ難しい、というものです。このように、NとNP双方の複雑性クラスが等しいか否かを解決するのは大変重要なことです。NP完全問題を効率的に解けるかどうかという問題は、「P対NP問題」と呼ばれています。これは、現在未解決の数

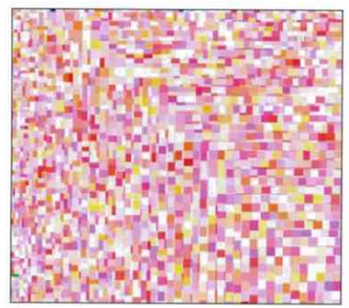

Fig. 2

Fig. 3

Fig. 4

Fig. 5

学問題の中では大変有名なもので、クレイ数学研究所がその解に百万ドルの懸賞金を提示している7つの問題の一つです。また、この問題は、果たして証明法の発見は証明法の検証よりも困難であるか、と問うのに等しく、我々が考えるように、証明法の発見には本質的に創造的なインスピレーションが必要なのか、という問いにもつながるため、多くの難問の中でも、哲学的に特に大きな意義を持っています。NP完全は、計算の複雑性の抽象的理論と組合せ問題を解くという現実的な作業をつなぐ、最も重要な架け橋であると言えます。今もなお、「NP完全」という言葉を、タイトルやアブストラクト、またはキーワードに含んでいる論文が、毎年6,000本ほど発表されています。NP完全は、大学でコンピュータ科学を専攻する学生全員が学ばなければならない、理論計算機科学のトピックなのです。

NP完全によって、様々な分野の科学者が計算の有する基本的な限界に関心を持つようになりました。NP完全は今や専門用語としての意味に加えて、動的システムのカオス、経済学の限定合理性、理想的とされる投票システムの不可能性、人工知能は人間の脳を刺激することができないこと、そして遺伝的非決定論などを含む現象におけるメタファーにもなっています。

NP完全の研究は、純粋数学のコミュニティーにおける計算複雑さの理論に対する認識を改めさせ、その地位の向上に貢献しました。1975年、私はプリンストン大学の高等研究所で行ったヘルマン・ヴァイル・レクチャーのテーマとして複雑さの理論を取り上げましたが、その数年後には、この分野におけるプログラムが同研究所で立ち上がり、成功を収めました。1985年には、フィールズ賞を受賞したスティーヴン・スメイル氏とともに、バークレーの数理科学研究所で計算の複雑さに関する年間プログラムのまとめ役をしました。

ある組合せ探索問題がNP完全であると分かった場合には、いくつかの対処方法があります。例えば、車で様々な町を巡回する計画を立てるとします。それぞれの町間の移動時間は分かっていて、全部の町を回るトータルの移動時間をできるだけ短くする方法を考えます。これは「巡回セールスマン問題」と呼ばれる、一般的なNP完全問題です。この問題はNP完全なので、あらゆるケースに当てはまる解を持つ効率的なアルゴリズムを期待することはできません。しかし、Fig. 6に示すように、特定の例における解を見つけ出すことは可能かもしれません。

もし不可能であっても、シンプルな試行錯誤的アプローチを用いることによって、最適とは行かないまでも十分に満足できる解を得ることができるかもしれません。実

P is unequal to NP, then some conclusion follows. Here are two examples: if P is unequal to NP, then cryptographic protocols that are useful for information transmission cannot be broken; if P is unequal to NP, then certain problems are hard to solve approximately. So it is crucial that we settle this question of deciding whether these two complexity classes are equal. The problem of deciding whether the NP-complete problems (and hence all problems in NP) are efficiently solvable is known as the P vs. NP problem. The P vs. NP problem has taken its place among the most prominent open questions in mathematics. It is one of the seven problems for whose solution the Clay Mathematics Institute has offered a million-dollar reward. Among those select problems it may well have the greatest philosophical significance, since it is equivalent to asking whether finding a proof is harder than checking a proof, and hence whether finding proofs inherently requires creative inspiration, as we expect. NP-completeness is the most important bridge between the abstract theory of computational complexity and the practical business of solving combinatorial problems. About 6,000 papers published each year have the term "NP-complete" in their title, abstract or list of keywords. NP-completeness is the one topic from theoretical computer science that every computer science undergraduate must learn.

NP-completeness has drawn the attention of many scientific fields to the fundamental limits of computation. Beyond its precise technical meaning, NP-completeness has become a metaphor for phenomena such as chaos in dynamical systems, bounded rationality in economics, the impossibility of certain ideal voting systems, the inability of artificial intelligence to simulate the human brain, and genetic indeterminism.

The work on NP-completeness contributed to raising the visibility and respectability of computational complexity theory within the pure mathematics community. In 1975 complexity theory was the topic of my series of Hermann Weyl lectures at the Institute for Advanced Study in Princeton, and a few years later the Institute established a thriving program in this subject. In 1985 I was co-organizer with the Fields Medalist Stephen Smale of a yearlong program in Computational Complexity at the Mathematical Sciences Research Institute in Berkeley.

Even when a combinatorial search problem is found to be NP-complete, there may be ways to deal with it in practice. Suppose, for example, that you are planning a driving trip that will take you to many cities. You know the driving

際に我々は、計算が速く、典型的なケースにはまずまずの解を与えてくれる発見的（ヒューリスティック）アルゴリズムをしばしば利用しています。ただし、そのパフォーマンスを普遍的に保証することはできません。Fig. 6は、101の町を回る時のヒューリスティックな方法で得られた解を示しています。こういった発見的アルゴリズムは、現実に発生する問題で大きな成果を上げることがしばしばありますが、なぜそうなるかについては理論的に説明されていません。時にはこうしたヒューリスティックなアルゴリズムが、大半もしくはほとんどの例に適用可能であることを証明することもできますが、果たしてそれがすべての例に適用され得るのかというのは別の問題です。

計算生物学

1990年、ヒトの遺伝学的構造を解析するという大きな目標を掲げてヒトゲノム解析プロジェクトが始まりました。我々の遺伝的資質というものは、主に染色体の中にあるDNAで構成されています。DNA分子とは、A、C、T、Gという４種類の塩基のいずれかを持ったヌクレオチドが２本の鎖状に長く連なったものです。従って、このプロジェクトが基本的に目指したことは、ヒトゲノム、すなわち染色体中に含まれる30億個のヌクレオチドの配列解析を行うことでした。最先端の技術を用いることにより、ごく低い誤り率で、それぞれ数百のヌクレオチドを含む、数百万の無作為な断片の配列を決定することができました。Fig. 7は、短いDNA配列を自動的に読み取る機械から得られたデータを示しています。

私は、この数百万のノイズを含んだデータの断片から長いDNAの配列を再構築する、という組合せ問題にすぐさま惹きつけられました。こうした作業の第一歩は、ゲノムに沿って散らばった数万というランドマーク配列の位置を決定するという、いわゆる物理的マッピング問題でした。1990年代には、私は少数の学生から成るチームをいくつか組織し、この物理的マッピング問題を解決するためのアルゴリズムとソフトウエアの開発を行いました。私たちのアプローチは科学的な見地からは大きな成功を収めることができたのですが、影響度という点ではほとんどありませんでした。なぜなら、当時、ゲノム配列を構築するためのデータのとりまとめを行っていた中核的な研究機関と十分なつながりが無かったからです。

ヒトゲノムの暫定的な配列が初めて発表された2000年以来、多くの生命体のゲノ

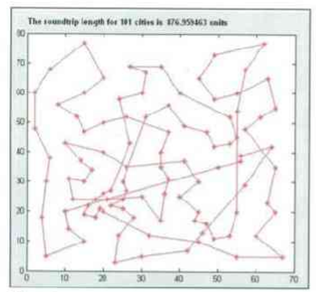

Fig. 6

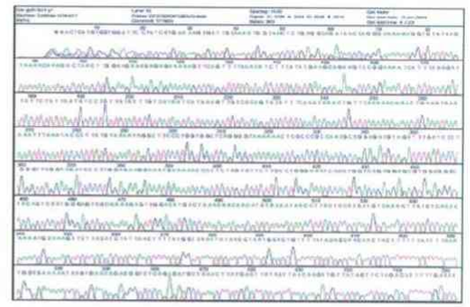
Fig. 7

time between each pair of cities, and you want to visit the cities in an order that will minimize your total driving time. This is an instance of a general NP-complete problem called the traveling-salesman problem. Because the problem is NP-complete we cannot expect an efficient algorithm that will solve all cases, but it still may be possible to find the minimum-time solution for a particular instance, such as the one shown in Figure 6.

And, if not, a simple trial-and-error approach may give a solution that is satisfactory, although not optimal. In practice, we often use so-called heuristic algorithms, which run fast and give acceptable solutions to typical examples, even though there are no general guarantees on their performance. Figure 6 shows a heuristic solution to an example with 101 cities. Such heuristic algorithms often perform quite well on the instances that arise in practice, but we have no theoretical explanation of why this is the case. Sometimes we can prove that a heuristic algorithm succeeds on most instances, or even on almost all instances, but this does not imply that it will perform well on the examples that occur in practice. The mysterious success of some heuristic algorithms remains a challenge to be explained in the future.

Computational Biology

In 1990 the Human Genome Project was started, with the ambitious goal of determining the genetic makeup of the human species. Our genetic endowment consists mainly of the DNA in our chromosomes. A DNA molecule is a long,

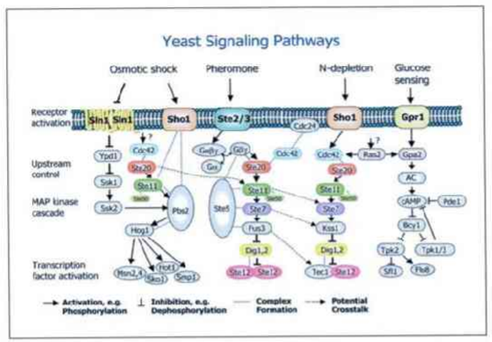

Fig. 8

ム配列が解析されてきましたが、この配列を決定する作業というのは、生細胞がどのように機能しているかを解明するための第一歩に過ぎません。こうした配列内の遺伝子を決定することは、今後の課題として残されています。配列が決まることにより、遺伝子がRNAやたんぱく質などの生体分子の生成をどのようにコントロールしているかを解明することや、Fig. 8に示す通り、細胞の複製、自己修復、構造の維持、他の細胞との相互作用、環境への反応、そして体内で特定の機能を発現するために一緒に働く生体分子の複雑なネットワークに関する推論が可能になると思います。多くの疑問が頭に浮かんできます。「肝細胞、血球、皮膚細胞の機能は、同じ遺伝子を含んでいるにもかかわらず、全く機能が異なっているのはなぜか？」「なぜ、がん細胞の振舞いは正常細胞とは異なっているのか？」「私たちのゲノムにおける差異のどの部分が疾病と関連しているのか？」「細胞の機能をコントロールしている複雑な制御ネットワークや免疫システムを本当に理解できるのか？」。このような分野においては、このたび京都賞を一緒に受賞したポーソン博士が根幹的な研究をされています。私は、ポーソン博士のような生物学者になろうとしたわけではありませんが、上記のような問いに答えるためのデータ解析を数学的に支援しようと考えていたので、そのために必要な生物学の知識を習得しようと努めました。今携わっている研究は特に、ゲノムレベルでの個体間の違いが病気への罹りやすさにどう関係しているのか、というものです。

この分野に関しては、私の教え子であり、現在は逆に私がアドバイスを貰う立場になっている優秀な年下の科学者と仕事をするという幸運に恵まれています。彼らは生

double-stranded chain of nucleotides, chemical units of four types: A, C, T and G. Thus, a fundamental goal was to determine the sequence of the human genome: the three billion nucleotides within the chromosomes. Using advanced technology it was possible to determine, with a small error rate, the sequences of millions of random fragments, each containing several hundred nucleotides. Figure 7 shows signals that come from a sequencing machine that automatically reads short DNA sequences.

I was immediately attracted to the combinatorial problem of reconstructing a long DNA sequence from noisy measurements of these millions of fragments. A first step in this direction was the so-called physical mapping problem of determining the positions of tens of thousands of landmark sequences scattered along the genome. During the 1990s I worked with small teams of students to develop algorithms and software for solving the physical mapping problem. Our approach was very successful from a scientific point of view, but had little influence, as we were not sufficiently connected with the leading centers where the data for assembling the genome sequence was being gathered.

The first draft sequences of the human genome were announced in 2000, and since then the genomes of many organisms have been sequenced, but sequencing is only the first step toward understanding how living cells perform their functions. It still remains to determine the genes within these sequences, to understand how the genes direct the production of biomolecules such as RNA and protein, and to infer the complex networks of biomolecules, such as the networks indicated in Figure 8, that act in concert to control how cells replicate, repair themselves, maintain their structure, communicate with other cells, respond to their environments and perform their specialized functions within the body. Many questions suggest themselves. How is it that liver cells, blood cells and skin cells function very differently even though they contain the same genes? Why do cancer cells behave differently from normal cells? Which variations in our genome are correlated with disease? How can we understand the complex regulatory networks that control the functioning of cells and systems such as the immune systems? This is, of course, the area in which my fellow laureate, Professor Pawson has done fundamental work. I did not aspire to be a biologist like Professor Pawson but tried to learn enough biology to provide mathematical support for the investigation of data that allows such questions to be addressed. My current work is concerned in particular with inferring how differences between individuals at the

物学に関する知識を私と共有することによって、こうした細胞の秘密を明らかにするにはどのようなアルゴリズムの問題が解明されなければならないかを明確にする作業を手助けしてくれています。大変優秀な生物学者であるワシントン大学のガレット・オーデル教授は、今私たちの多くが取り組もうとしている挑戦を次のように表現しています。「相互に強く作用しているような遺伝子のグループにゲノムを分けることによって、ゲノム全体がどのように機能しているかを理解できるようになるだろう。一旦これらのネットワーク・モジュールを特定し、その機能を理解したならば、今度はそのようなネットワーク間の相互作用を明らかにすることになり、それを続けていけば、長い時間がかかるかもしれないが、やがては、ゲノム全体の機能を理解することができるようになるだろう」。

「計算」レンズ

最後に、コンピュータ科学がこれから辿るであろう方向性についての私の考えをお話ししたいと思います。コンピュータ科学は、「人工物の科学」と呼ばれてきました。これは、コンピュータ、プログラム、アルゴリズムなど、自然界に元々存在するのではなく人為的に考え出され、特定の目的に使用されるようなものを基本的に取り扱っているためです。しかし、自然界で発生するシステマティックなプロセスの中にも、計算を必要とするという意味において、コンピュータ科学の範疇に分類することができるものがあります。こうしたプロセスの多くは生体システム、すなわち、ニューロンのネットワークによる学習、免疫系が侵入してきた微生物に対して防御態勢を整えるプロセス、胚発達の過程で細胞が特定の機能を得るプロセス、鳥の群れや蟻のコロニーの自己組織性といった動物群集の集団挙動、そして種の進化などで見られます。このような生物の様々なレベルにおいて、「自然はどの程度計算しているのか？」「いろいろな過程はどの程度アルゴリズム的なのか？」といった問いが浮かんできます。例えば、「蟻のコロニーの自己組織性はアルゴリズム的なのか？そして、そのアルゴリズムを理解できるのか？」(Fig. 9)。

また、細胞は計算を行うのか (Fig. 10)？免疫系は計算を行うのか (Fig. 11)？脳は計算を行うのか (Fig. 12)？最後の問いに関連したおもしろい質問があります。「脳は機械なのか？」というものです。もし、脳内で生起する全ての生化学反応を記述できたとすれば、例えば、「意識」のようなものまでも説明できるようになるので

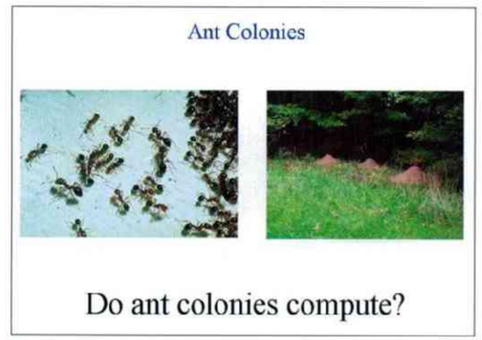

Fig. 9

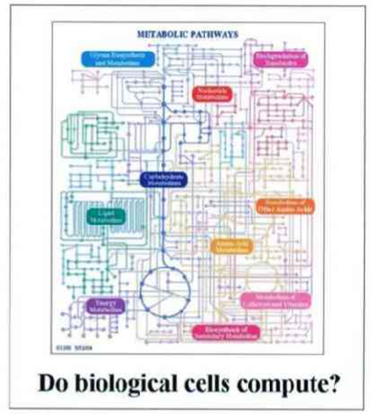

Fig. 10

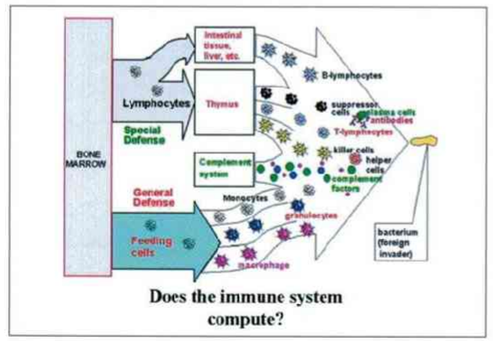

Fig. 11

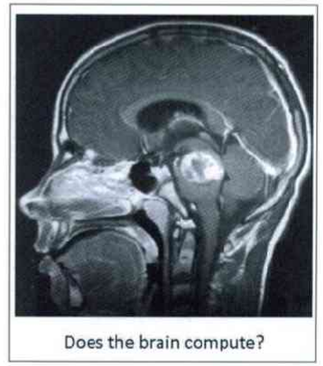

Fig. 12

しょうか？個人的には、このようなことができるようになるとは思いません。

ここまで、アルゴリズム的な性質を持った生物学的過程が計算レンズを通して見るとどのように見えるかについて話してきました。計算レンズという考え方は、人工物へも適用できます。インターネットを考えてみましょう。それを完全にデザインしたり、特徴づけたりすることはできません。インターネットは、むしろ複雑システムとして出現しました。そして、それは同時に計算科学的なものであり、社会的なものであり、経済的なものでもあります。その中では、多くの独立した主体が調和することなく行動しています。そのために、インターネットは、一人のクリエーターによって作られた人工的なものとしてよりも、自発的に進化する組織として研究されています。インターネットやウェブについて見識を持っている同僚のクリストス・パパミディミトリューは、「最初は、他の分野のパイオニアが宇宙や細胞や脳や市場に立ち向かった時に感じたのと同じ戸惑いを感じながら、人工物に立ち向かわざるを得なかった」と言っています。

インターネットの研究は、新興のコンピュータシステムとしてなされていますが、それと同時に、社会科学の研究対象にもなりつつあります。インターネットに基づいたネットワークや、個人、組織、コミュニティーを互いに関連づけるウェブに基づいた相互作用のネットワークでは、１分おきにデータが蓄積され、利用可能となっています。このような状況においては、以下のような問いが次々と浮かんできます。アイデアや意見や制度や技術が、ネットワークを通してどのように広がっていくのか？まとまりのあるコミュニティーやサブコミュニティーをどのようにして特定できるのか？「６次の隔たり」現象をどのようにして調査し、理解できるのか？

他のいくつかの分野も、計算レンズを通して見ることが可能です。電子レベルでの計算デバイスの挙動は、量子力学で記述される必要がありますし、量子計算の抽象的な数学モデルの構築へと導くものと思われます。量子コンピュータの実現が待ち望まれています。それは、量子コンピュータの情報基本単位が、キュービットになるためです。現在使われているコンピュータの情報基本単位はビットといわれるもので、０と１の値しかとることができません。キュービットはこれに比べると、ずっと多くの情報を担うことができます。量子コンピュータの潜在能力を十分に生かすことができるかどうかは大変興味ある問題です。量子コンピュータを作るための技術はまだ確立されていませんが、その能力については既に調べられています。また、量子力学の標準理論の正しさを検証する実験にも、量子コンピュータを使うことができます。今

genomic level affect their susceptibility to disease.

I have been fortunate to work with gifted younger scientists who began as my students but became my mentors. By sharing their biological knowledge with me they help me formulate the key algorithmic problems that must be resolved to reveal the secrets of the cell. Here is a quote from a distinguished biologist at University of Washington, Garrett Odell laying down a challenge, which many of us are now trying to address. He says, "We can approach understanding how the whole genome works by breaking it down into groups of genes that interact strongly with each other. Once researchers identify and understand these network modules, the next step will be to figure out the interactions within networks of networks, and so on until we eventually understand how the whole genome works, many years from now."

The Computational Lens

At this point, I would like to offer a speculation about the future direction of computer science. Computer science has been called a "science of the artificial" because it deals in large part with entities such as computers, programs and algorithms that are devised and specified by humans rather than occurring in the natural world. However, many processes that occur in nature also lie within the province of computer science, since they can be regarded as computational. Many such examples occur in biological systems. Each of the following processes can be regarded as to some extent a systematic process of computation: learning by networks of neurons; the process by which the immune system marshals its defenses against an invading microbe, or by which cells acquire specialized functions during embryonic development; the collective behavior of animal communities, such as the flocking of birds or the self-organization of ant colonies; and the evolution of species. So, at each of these various levels of biology, we can ask, "To what extent is nature computing?" "To what extent are the processes algorithmic?" "Is the self-organization of ant colonies algorithmic in nature? Can we understand it as an algorithm? (Fig. 9)"

Do biological cells compute (Fig. 10)? Does the immune system compute (Fig. 11)? Does the brain compute (Fig. 12)? In this latter regard, the most interesting question is the following. Is the brain a machine? If we could elucidate every biochemical reaction occurring in the brain, would we be able to explain

後、このような検証実験について、より多くのアイデアが提案されるようになると思われます。私の同僚、ウメシュ・バジラーニは、「量子コンピュータは強力な計算機というよりも、むしろ量子力学を検証するためのものだ」と言っています。

計算理論は、統計物理学とも深い関係があります。統計物理学は、互いに相互作用を及ぼし合う、原子・分子・分子スピンなどの集合を確率論的に取り扱う学問です。水が凍る現象や、金属が強磁性化する現象などを取り扱います。このような現象を取り扱う統計物理学と、コンピュータ科学とは、共通する考えを持っているように思われます。なぜなら、コンピュータ科学では、多数の組合せ論的変数や条件を含む確率論的モデルや、インターネットを通して互いにコミュニケートしたり、交渉したりする主体を含む確率論的モデルを取り扱うからです。

計算理論は、経済とも共通する考え方を含んでいます。例えば、経済メカニズムデザインという分野では、計算理論と同じような考えが使われています。この分野は、インターネット上での商取引に必要な経済メカニズムの発展に影響されています。経済デザインはある種のアルゴリズムで、その入力は個人的なデータや個人的な興味をもった主体によってなされます。一例が、オンラインオークションです。経済メカニズムデザインのゴールは、参加者に何らかのインセンティブを与えることによって、その目的を正直に表明させ、社会福祉や社会利益を最大にするといったデザイナーのゴールを達成するように行動させることだと思われます。計算理論と経済との別の接点は、計算ゲーム理論です。古典的なゲーム理論では、参加者が完全に合理的に振舞うと仮定して、利害が対立する参加者がどのように振舞うかを研究していました。複雑系としての計算ゲーム理論では、新たな要素、すなわち、計算量について上限が存在するために、参加者の戦略的振舞いが制限され得るという点を導入しています。そのために参加者は、完全に合理的に判断したり、振舞ったりすることができなくなる可能性があります。

これまでの大まかな説明により、計算における限界や制限というレンズを通して、どのように物理的・社会的・工学的システムが見えてくるかを説明してきました。そして、計算における限界や制限がもたらす新しいものの捉え方や考え方をも提示しています。コンピュータ科学におけるこのような見方や考え方は、科学研究の中心を占めるようになると思われます。私の将来の研究も、このような方向に沿ったものであると考えています。

最後に、個人的な見解を述べさせていただいて講演を終わります。研究を主体とす

consciousness? I personally don't believe that this is the case.

The computational lens that I applied to biological processes can also be focused on man-made artifacts. Consider the Internet. It was never completely designed or specified. Rather, the Internet emerged as a complex system, simultaneously computational, social and economic, through the uncoordinated actions of many independent agents. For this reason, it is perhaps best studied empirically as an evolving organism, rather than an artificially constructed process with a single creator. It is more like a market or society than a well-specified mathematical object. My colleague, Christos Papadimitriou, speaking of the Internet and the Worldwide Web, said, "For the first time, we had to approach an artifact with the same puzzlement with which the pioneers of other sciences had to approach the universe, the cell, the brain and the market."

The study of the Internet as an emergent computational system also raises questions for the social sciences. Minutely detailed data is becoming available about Internet-based social networks: networks of Web-based interactions linking individuals, organizations and communities. Many questions can be addressed. How do ideas, opinions, innovations and technologies spread? How can we identify coherent communities and sub-communities? How can we understand and exploit the "six degrees of separation" phenomenon?

Several other fields can be viewed through the computational lens. The behavior of computational devices at a sub-atomic level needs to be framed in terms of quantum mechanics, and leads to the abstract mathematical model of a quantum computer. The possibility of someday building quantum computer is tantalizing because, in principle, the qubit, the basic unit of information in a quantum computer, is much richer in content than the bit, the basic unit of information in a classical computer, which takes only the values of 0 and 1. So, there is a great interest in determining whether we can harness the potential power of a quantum computer. Although the technology for building a quantum computer has not yet been realized, the computational power of this model is being investigated, and can be used to define experiments that test the validity of the standard theory of quantum mechanics. And so, in connection with trying to build quantum computers much more probing experiments can be defined, which could possibly reveal flaws in the standard theory of quantum mechanics. As my colleague Umesh Vazirani puts it, "Quantum computing is as much about testing quantum physics as it is about building powerful computers."

る大学の教授は、少なくとも私にとっては、世の中で最高の職業です。個人の自主性の占める割合が大きく、優秀な学生の指導や彼らの手本となることができるだけでなく、科学技術の新しい分野における最先端の研究を支え、後押ししてくれる環境が整っています。研究者がこうしたキャリアを歩むことが可能となった時代に生まれたことは、私にとって幸運でした。

Another area where there are intense links between the theory of computation and physical science is statistical physics. Statistical physics is based on probabilistic models of the interactions of large assemblages of atoms, molecules or molecular spins. The field studies such phenomena as the freezing of water and the magnetization of metals. Statistical physics finds common ground with probabilistic models in computer science involving the interactions of large numbers of combinatorial variables and constraints, or the interactions of many agents communicating and negotiating over the Internet.

The theory of computation is also finding common ground with economics. One area of overlap is economic mechanism design. This area has been stimulated by the development of economic mechanisms for conducting transactions over the Internet. An economic mechanism is an algorithm whose inputs come from economic agents with private data and selfish interests. One example is the design of on-line auctions. The goal in economic mechanism design is to induce the participants, via incentives, to state their objectives truthfully and respond in ways that achieve the designer's goals, such as social welfare or profit maximization. Another emerging area where the theory of computation meets economics is computational game theory. Classical game theory is the study of rational behavior in situations of conflict, and assumes perfect rationality. Computational game theory based on complexity theory injects a new ingredient, by postulating that the strategic behavior of the participants is limited by computational complexity, so that they may not be able to achieve perfect rationality because of computational limitations.

The brief descriptions I have given illustrate how viewing physical, social or engineered systems through the lens of their computational capabilities or requirements provides new insights and ways of thinking, and promises to move computer science closer to the center of scientific discourse. I expect much of my future work will be in this direction.

Let me close with a personal statement. Being a professor at a research university is, for me at least, the best job in the world. It provides great personal autonomy, the opportunity to serve as a mentor and role model for talented students, and an environment that encourages and supports work at the frontiers of emerging areas of science and technology. I am fortunate to have come along at a time when such a career path was available to me.

稲盛財団2008——第24回京都賞と助成金

発　　行　2009年7月1日
制　　作　財団法人 稲盛財団
〒600-8411　京都市下京区烏丸通四条下ル水銀屋町620番地
Phone: 075-353-7272　Fax: 075-353-7270
E-mail admin@inamori-f.or.jp　URL http://www.inamori-f.or.jp/
ISBN4-900663-24-7 C0000