Kiến thức bổ sung

1. Bài toán và thuật toán

Theo một nghĩa nào đó, có thể xem máy tính như một công cụ để giải các bài toán. Khái niệm bài toán ở đây được quan niệm một cách rất mềm dẻo bao gồm các bài toán hiểu theo nghĩa thuần tuý toán học như những trường hợp riêng. Nói một cách đơn giản, ta có thể hiểu việc giải một bài toán như việc làm một việc gì đó bằng máy tính. Như vậy, học sử dụng máy tính ở đây thực chất là học cách giao cho máy tính làm được việc mà ta muốn nó làm. Cần chú ý rằng máy tính là một vật vô tri vô giác, sự "thông minh" của máy chỉ là thể hiện sự thông minh của người sử dụng nó. Cùng một loại máy tính, chất lượng khai thác công cụ đó rất khác nhau ở mỗi người. Vậy thì thế nào là giao cho máy điều cần làm? Nói ngắn gọn, đó là việc lập chương trình cho máy tính. Mỗi chương trình nhằm mục đích giải quyết một bài toán nào đó, mục đích đó phải được thể hiện thành những dãy các thao tác để cho máy thực hiện. Việc viết các "mệnh lệnh" để giao cho máy thực hiện được gọi là việc lập chương trình cho máy tính.

Giải bài toán trên máy tính gồm các bước sau: *xác định rõ bài toán cần giải quyết, lập phương án giải quyết (lựa chọn hoặc thiết kế thuật toán), viết chương trình, thử và hiệu chỉnh chương trình, viết tài liệu.*

Trong một phạm vi nào đó, ta có thể xem sự phát triển của toán học gắn liền với quá trình giải các bài toán. Ở đây ta không đề cập đến quá trình toán học tự hoàn thiện nhằm xây dựng một hệ thống ngôn ngữ và các công cụ để có thể mô hình hoá những quá trình diễn ra trong các lĩnh vực khoa học và thực tiễn khác nhau, phát biểu chúng thành các bài toán.

Ví dụ 1. Bài toán tìm nghiệm của đa thức một ẩn:

Input: Số nguyên dương n; các số thực $a_0, a_1, \ldots, a_n$;

Output: Các số thực X sao cho $a_0 + a_1X + ... + a_nX^n = 0$ hoặc không có số thực X nào như vậy.

Ví dụ 2. Tính giải được theo căn của phương trình đa thức:

Input: Số nguyên dương n; các số thực $a_0, a_1, ..., a_n$;

Output: Công thức tính nghiệm của phương trình $a_0 + a_1X + ... + a_nX^n = 0$ theo các hệ số a_i thông qua các phép cộng, trừ, nhân, chia và khai căn.

Ví dụ 3. Bài toán thứ 10 của Hilbert:

Input: Đa thức P bậc N với các hệ số là số nguyên;

Output: P có nghiệm nguyên hoặc P không có nghiệm nguyên.

Một phương pháp hiệu quả để giải một bài toán nào đó là việc xác định một quá trình bao gồm một dãy hữu hạn các thao tác đơn giản được sắp xếp theo một trình tự xác định sao cho theo đó, từ Input, ta sẽ nhận được Output cần tìm hoặc khẳng định không có Output như bài toán đó đòi hỏi.

Về mặt thuật ngữ, ta gọi một quá trình như vậy là một *thuật toán (algorithm)*. Đó chính là quan niệm trực giác về thuật toán. Như vậy, giải một bài toán có nghĩa là xây dựng một thuật toán đối với bài toán đó. Nói cụ thể hơn, việc giải một bài toán P nào đó là việc xây dựng một tập hợp hữu hạn các thao tác đơn giản được sắp xếp theo trình tự xác định sao cho sau khi hoàn thành dãy thao tác đó, ta thu được Output của P. Chỉ những dãy thao tác như vậy mới có khả năng chuyển giao cho máy tính thực hiện được.

Toán học vẫn rất cần những nghiên cứu định tính và những kết quả có được theo hướng nghiên cứu này đã, đang và sẽ vẫn có những vị trí xứng đáng trong toán học nói riêng và khoa học nói chung. Tuy nhiên, trong khuôn khổ của tin học, khi nói đến việc giải bài toán, ta sẽ chỉ quan tâm đến những phương pháp hiệu quả để xuất phát từ Input của bài toán, ta dẫn ra được Output cần thiết.

Trong quá trình nghiên cứu cách giải bài toán theo nghĩa nêu trên, trong toán học đã diễn ra một quá trình phát triển đầy kịch tính. Cho tới đầu thế kỉ XX, với lòng tin tiên nghiệm vào việc mọi bài toán (phát biểu đúng đắn) đều có thuật toán giải, nhiều nhà toán học đầy tài năng và tâm huyết đã tiến công vào các bài toán đang tồn tại như những lời thách đố trí tuệ con người.

Vấn đề then chốt nhất của lí thuyết tính toán phát sinh ở đây. Để chứng minh một bài toán nào đó có thuật toán giải, ta chỉ cần đề ra một thuật toán theo quan niệm trực giác của khái niệm này và kiểm định tính đúng đắn của nó đối với bài toán đã cho. Tuy nhiên, khi cần chứng minh việc không có thuật toán giải một bài toán nào đó, quan niệm trực quan về thuật toán không thể là cơ sở bảo đảm cho một chứng minh toán học chặt chẽ được. Do đó, muốn chứng minh những điều khẳng định "không có thuật toán giải một bài toán nào đó", ta cần có một định nghĩa toán học cho khái niệm thuật toán.

Vào khoảng những năm 1930 – 1936, lần lượt các nhà toán học K.Gödel, S. Kleene, A. Church, A. Turing đã đề ra một số định nghĩa khác nhau cho khái niệm thuật toán. Trong số các định nghĩa toán học khác nhau (nhưng tương đương) về thuật toán, các khái niệm máy Turing (1937) và hàm đệ quy (1931–1936) được sử

dụng rộng rãi hơn vì có nhiều thuận tiện cho các nghiên cứu cả về lí thuyết lẫn thực hành. Ta có thể xem máy Turing là một mô hình toán học của máy tính.

Chính nhờ có được định nghĩa toán học về thuật toán, người ta đã chứng minh được có những bài toán không có thuật toán giải. Chẳng hạn, bài toán thứ 10 của Hilbert (ví dụ 3) không có thuật toán giải. Kết quả này là tập hợp công sức của nhiều nhà toán học và khâu cuối cùng được giải quyết bởi một sinh viên khoa Toán năm thứ 5 trường Đại học Tổng hợp Lê-nin-grat (Liên Xô cũ) năm 1972.

Cần chú ý rằng mỗi thuật toán chỉ giải một bài toán nào đó nhưng có thể có nhiều thuật toán khác nhau giải cùng một bài toán. Một vấn đề đặt ra là ta nên chọn một thuật toán tốt để giải bài toán đã cho.

Nhưng thế nào là thuật toán tốt? Đây là nội dung nghiên cứu của lí thuyết độ phức tạp của thuật toán. Nói một cách đơn giản, khi dùng máy tính thực hiện một chương trình (thể hiện một thuật toán nào đó), hệ điều hành cần cung cấp cho chương trình đó các tài nguyên như giờ CPU, bộ nhớ,... Số lượng các tài nguyên cần dùng thể hiện độ phức tạp của thuật toán. Thuật toán được xem là tốt nếu chương trình tương ứng dùng ít tài nguyên hơn.

Trong các tài nguyên cần dùng để chạy chương trình, hiện nay người ta quan tâm nhiều nhất đến thời gian vì đó là dạng tài nguyên không tái tạo được. Một thuật toán được xem là tốt nếu chương trình tương ứng chạy nhanh hay chính xác hơn, chạy trong thời gian chấp nhận được.

Để đánh giá độ phức tạp của một thuật toán, ta có thể ước tính số thao tác cơ bản cần dùng để thực hiện thuật toán. Hai cấu trúc trong các thuật toán: cấu trúc lặp và cấu trúc đệ quy thường có ảnh hưởng lớn đến độ phức tạp của thuật toán. Do vậy khi đánh giá người ta quan tâm nhiều nhất đến hai cấu trúc đó. Số thao tác của một vòng lặp dễ dàng tính được nhưng nếu có một chương trình con đệ quy, việc ước lượng số thao tác khó khăn hơn vì ta phải tính một đại lượng theo một công thức truy hồi.

Trong các ngôn ngữ lập trình, hợp ngữ là một phương tiện khá thuận lợi cho các nhà lập trình chuyên nghiệp nhưng vẫn chưa thật thích hợp cho đa số những người sử dụng máy tính muốn thể hiện thuật toán bằng những câu lệnh gần với ý nghĩa thực tế của các thao tác tương ứng mà câu lệnh đó mô tả. Từ đầu những năm năm mươi của thế kỉ XX, người ta đã xây dựng những ngôn ngữ lập trình vạn năng theo đó các câu lệnh được viết gần với ngôn ngữ tự nhiên hơn. Các ngôn ngữ lập trình này được gọi là các ngôn ngữ lập trình bậc cao. Cũng như đối với hợp ngữ, mỗi ngôn ngữ lập trình bậc cao đều cần có trình thông dịch hay biên dịch để dịch nó sang ngôn ngữ máy để máy có thể thực hiện được. Chú ý rằng nói chung trình hợp dịch dịch mỗi lệnh viết bằng hợp ngữ thành một lệnh máy trong khi đó các trình thông dịch hay biên dịch thường dịch một lệnh viết bằng ngôn ngữ vạn năng thành một nhóm lệnh máy. Các trình thông dịch có đặc điểm là dịch đến đâu thực hiện luôn trong khi các trình biên dịch, sau khi toàn bộ chương trình được dịch xong mới đi vào khâu thực hiện.

Ngôn ngữ bậc cao đầu tiên được xây dựng vào năm 1954 và đưa ra thương mại hoá vào năm 1957 là ngôn ngữ FORTRAN (FORmula TRANslator – Bộ dịch các công thức) của hãng máy tính IBM. Cho tới nay với rất nhiều cải tiến, FORTRAN vẫn là một ngôn ngữ được sử dụng nhiều để giải các bài toán về khoa học kĩ thuật. Tiếp theo là sự ra đời của các ngôn ngữ bậc cao khác như COBOL (COmmon Business–Oriented

Language – ngôn ngữ hướng vào các bài toán kinh doanh hay quản lí) ra đời năm 1959, BASIC (Beginner's All-purpose Symbolic Instruction Code - Mã lệnh hình thức vạn năng cho người bắt đầu) ra đời năm 1965. Hiện nay nhiều ngôn ngữ lập trình bậc cao được dùng nhiều cho các máy vi tính như PASCAL (ra đời năm 1971) và C (ra đời năm 1972), C++, Java,...

Để thể hiện thuật toán giải quyết một bài toán bằng máy tính, ta cần viết một chương trình theo một loại ngôn ngữ mà máy có thể hiểu được. Đó có thể là một ngôn ngữ bậc cao, hợp ngữ, ngôn ngữ máy hoặc một phần mềm chuyên dụng thích hợp cho công việc cần làm. Muốn làm được như vậy, ta cần học ngôn ngữ đó. Chú ý rằng khả năng hiểu ngôn ngữ của máy tính rất hạn chế nên mặc dù các ngôn ngữ lập trình rất đơn giản, muốn sử dụng được chúng, trước hết ta cần tuân thủ nghiêm ngặt những quy ước của từng ngôn ngữ.

2. Bàn phím (Keyboard)

Các phím được chia thành hai loại: nhóm phím kí tự và nhóm phím chức năng. Khi gõ phím kí tự, kí hiệu trên mặt phím sẽ xuất hiện trên màn hình. Các phím chức năng đều có tên và việc sử dụng chúng thường được quy định bởi các phần mềm cụ thể. Các phím chức năng gồm: Enter, Shift, Ctrl, Alt, Caps Lock, Backspace, Esc, Tab, Insert, Delete, Home, End, Page Up, Page Down, Pause, Print Screen, Scroll Lock, Num Lock, Windows, F1, ..., F12, bốn phím mũi tên theo bốn hướng dùng để dịch chuyển con trỏ màn hình.

Trong các phím kí tự, có 26 phím chữ cái tiếng Anh từ A đến Z, muốn gõ chữ thường, ta gõ trực tiếp phím tương ứng; muốn gõ chữ hoa, một ngón tay giữ phím Shift và ngón khác gõ phím tương ứng. Các phím kí tự khác có hai kí tự, nếu gõ phím đó, ta đưa vào máy kí tự dưới, nếu nhấn giữ phím Shift và gõ phím đó, ta đưa vào máy kí tự trên. Có một kí tự đặc biệt gọi là dấu cách (hoặc kí tự trống) được đưa vào máy bằng cách gõ phím dài (Space Bar), việc gõ phím này sẽ tạo dấu ngăn cách giữa các từ.

Trên mặt bàn phím có ba đèn hiệu gắn với các phím chức năng Num Lock, Caps Lock và Scroll Lock. Nếu đèn hiệu Num Lock sáng, gõ các phím ở khu vực phím số bên phải bàn phím để nhập dữ liệu số. Trong khu vực này, bốn phím /, *, – và + dùng để gõ các kí hiệu các phép toán chia, nhân, trừ và cộng. Vì nhiều công việc (kế toán, khí tượng,...) cần nhập chủ yếu các số vào máy tính nên phạm vi các phím số thu hẹp sẽ giúp người dùng thao tác nhanh hơn.

Phím Esc thường dùng để thoát ra khỏi tình trạng hiện tại trở về tình trạng trước đó. Phím Print Screen dùng để in nội dung màn hình ra máy in hoặc lưu nội dung màn hình hiện thời vào vùng nhớ tạm thời (Clipboard).

Nếu nhấn phím Caps Lock và đèn hiệu Caps Lock sáng, khi đó gõ các phím chữ, trên màn hình sẽ hiện ra chữ in hoa tương ứng còn nhấn giữ phím Shift và gõ phím chữ thì trên màn hình sẽ hiện ra chữ in thường tương ứng. Trong các hệ soạn thảo, phím Tab dùng để chèn một khoảng trống có độ dài cố định, độ dài này do từng phần mềm cụ thể mặc định.

Phím Del (Delete) dùng để xoá kí tự ở bên phải con trỏ, phím Backspace ở góc trên bên phải khu vực chính của bàn phím có kí hiệu ← dùng để xoá kí tự bên trái con trỏ.

Ngoài ra, các phím chức năng khác như Ctrl, Alt thường dùng kết hợp với phím khác để thực hiện một số việc do phần mềm cụ thể quy định. Các phím Home, End, Page Up, Page Down thường dùng để di chuyển con trỏ tương ứng về đầu dòng, cuối dòng, trở về trang màn hình trước, sang trang màn hình sau trong các hệ soạn thảo văn bản.

Phím Enter thường dùng để khẳng định với máy việc ta muốn làm. Trong các hệ soạn thảo văn bản, khi gõ phím Enter ta chuyển con trỏ xuống dòng tiếp theo. Chú ý rằng nếu đèn hiệu Num Lock tắt, khi gõ các phím trong khu vực phím số bên phải bàn phím, máy sẽ thực hiện theo các kí hiệu ghi trên phím.

3. Biểu diễn thông tin trong máy tính

Dữ liệu là thông tin được đưa vào máy tính sau khi mã hoá. Có nhiều loại dữ liệu khác nhau. Ở đây ta chỉ đề cập đến hai loại dữ liệu: kiểu xâu kí tự và kiểu số.

a) Kiểu xâu kí tự

Máy tính có thể dùng một dãy bit để biểu diễn một kí tự, thường là dạng nhị phân của mã ASCII của kí tự đó.

Để biểu diễn một xâu kí tự (dãy gồm các kí tự), máy thường dùng 1 byte để ghi nhận độ dài xâu (số lượng kí tự của xâu) và trong các byte tiếp theo, mỗi byte ghi một kí tự của xâu theo thứ tự từ trái sang phải. Với cách biểu diễn này, máy chỉ có thể biểu diễn xâu với độ dài tối đa bằng 255. Muốn biểu diễn các xâu dài hơn hoặc một văn bản, máy phải dùng các kiểu dữ liệu khác.

b) Các hệ đếm dùng trong máy tính

Trong Toán học, ta đều biết mỗi số nguyên dương P lớn hơn 1 đều có thể dùng làm cơ số để biểu diễn mọi số, ta dùng thuật ngữ hệ đếm P (hệ cơ số P). Dạng biểu diễn của số trong hệ đếm P được gọi là dạng P–phân. Hệ đếm phổ biến được sử dụng là hệ cơ số 10 hay còn gọi là hệ thập phân. Tuy nhiên, như đã nói trong SGK, việc dùng *hệ đếm cơ số* 2 (hay còn gọi là hệ nhị phân) tiện lợi và phù hợp hơn đối với cấu trúc của máy tính. Ngoài hệ đếm cơ số 2, trong tin học còn dùng hệ cơ số 8 và 16. Việc chuyển đổi giữa các hệ đếm này rất đơn giản. Nếu có biểu diễn số dạng nhị phân, việc chuyển sang hệ đếm 8 (tương ứng 16) có thể thực hiện bằng cách gộp các nhóm ba (bốn) chữ số liên tiếp với nhau từ phải sang (với phần nguyên) và từ trái sang (với phần phân).

Các địa chỉ của ô nhớ trong máy tính được đánh số bằng hệ cơ số 16 trong khi các số cần tính toán đều dùng biểu diễn nhị phân.

c) Cách biểu diễn số nguyên trong máy tính

Trong cách biểu diễn số nguyên có dấu, ta dùng bit cao nhất làm bit thể hiện dấu (bit này được gọi là *bit dấu*) với quy ước 1 là số âm, 0 là số không âm. Khi đó chỉ còn 7 bit để thể hiện giá trị số. Nếu số không âm, 7 bit còn lại là dạng cơ số 2 của số đó, do đó các số không âm sẽ có phạm vi giá trị từ 0 đến 127. Việc biểu diễn số âm trong thực tế phức tạp hơn so với cách được trình bày trong SGK. Số âm nhỏ nhất có thể biểu diễn được là –128 sẽ được biểu diễn bởi dãy bit

10000000

Nếu Y là một số âm trong phạm vi từ –127 đến –1, kí hiệu X = 128 – |Y|, khi đó Y sẽ được biểu diễn bởi dãy bit $1b_6 \ldots b_0$ mà $b_6 \ldots b_0$ là dạng nhị phân của X.

Ví dụ –1 và –127 sẽ được biểu diễn tương ứng bởi các dãy bit

11111111 10000001

Biểu diễn này được gọi là *mã bù* của số âm.

Để thực hiện tính toán, người ta còn dùng một biểu diễn khác gọi là *mã ngược* của số âm. Nếu số âm Y có trị tuyệt đối bằng X, mã ngược của Y sẽ có dạng

$$1c_6 \ldots c_0$$

trong đó với $0 \le i \le 6$, $c_i = 1 - b_i$ và $b_7b_6 \ldots b_0$ là dạng nhị phân của X. Ví dụ *mã ngược* của các số –128, –127, –1 tương ứng là

11111111 10000001 10000000

Việc chuyển đổi giữa hai dạng mã có thể tiến hành như sau: Nếu có mã ngược, phần từ bit thứ 6 đến bit 0 của mã bù sẽ nhận được bằng cách cộng 1 vào phần tương ứng của mã bù.

Tóm lại, nếu dùng 1 byte để biểu diễn số nguyên có dấu, ta có thể biểu diễn được các số trong phạm vi từ –128 đến 127.

Bằng cách lí giải tương tự, nếu dùng 2 byte hoặc 4 byte để biểu diễn các số nguyên không dấu, các số đó thuộc phạm vi tương ứng từ 0 đến 65355 hoặc từ 0 đến 4294967295. Nếu dùng 2 byte hoặc 4 byte để biểu diễn các số nguyên có dấu, các số đó thuộc phạm vi tương ứng từ –32768 đến 32767 hoặc từ –2147483648 đến 2147483647.

Có cách rất đơn giản để suy ra những phạm vi giá trị trên dựa trên các luỹ thừa của 2.

d) Cách biểu diễn các số thực

Nói chung có hai cách viết số thực: *cách viết thông thường* và *cách viết với dấu phẩy động*.

Chú ý rằng, nói chung ta chỉ có thể biểu diễn các số thực một cách gần đúng với một sai số nào đó.

Để biểu diễn một số thực, máy thường dùng 4, 6, 8, 10 byte. Từ bit cao trở xuống, dùng 1 bit cho phần dấu của số, 1 bit cho phần dấu của bậc, tiếp theo là một đoạn bit cho phần giá trị của bậc và cuối cùng là các bit dùng cho giá trị của phần định trị.

Đối với các máy vi tính dòng IBM, bảng sau đây cho ta các thông tin khái quát về các khả năng biểu diễn số thực, trong đó cột thứ nhất ghi số byte dùng để biểu diễn, cột thứ hai cho phạm vi của trị tuyệt đối của số thực, cột thứ ba cho số chữ số chính xác.

4 byte	$10^{-45}..10^{38}$	7 hoặc 8
6 byte	$10^{-39}..10^{38}$	11 hoặc 12
8 byte	$10^{-124}..10^{308}$	15 hoặc 16
10 byte	$10^{-4932}..10^{4932}$	19 hoặc 20

TIN HỌC CÓ PHẢI LÀ KHOA HỌC

GS. TSKH. Phan Đình Diệu

Trường Công nghệ Đại học Quốc gia Hà Nội

Tôi nhớ cách đây chưa lâu, không ít nhà khoa học tuy xem trọng tác dụng của máy tính, nhưng vẫn không xem tin học là một ngành khoa học. Điều đó không phải là không có lí do. Máy tính thì có công dụng rõ ràng, nhưng tin học phải chăng chỉ giúp con người biết sử dụng máy tính và vì vậy chỉ là một công cụ, một phương tiện hỗ trợ cho toán học và các ngành khoa học khác khi khai thác máy tính? Đúng là khoa học máy tính, giai đoạn khởi đầu cho ngành tin học, đã nảy sinh từ sự ra đời của máy tính, nhưng trải qua mấy thập niên phát triển, cùng với sự phát triển cực kì nhanh chóng của công nghệ máy tính và truyền thông, ngành tin học đã qua bao lần tiến hoá, thay đổi, bổ sung nhiều nội dung mới để trở thành một ngành khoa học thực sự phong phú và đang tiếp tục phát triển mạnh như ngày nay.

Có lẽ chưa có một ngành khoa học nào mới mấy chục năm mà đã có lắm tên gọi như tin học. Ngay cả cái tên *Tin học* (chuyển từ tiếng Pháp Informatique, tương ứng tiếng Anh Informatics) mà ta dùng ở đây cũng chưa được thống nhất chấp nhận là tên gọi chính thức của ngành. Bắt đầu với tên gọi Khoa học máy tính (Computer Science) hoặc khoa học tính toán (thuật ngữ dùng ở Liên Xô cũ), ngành khoa học này cũng đã có những tên gọi khác nhau: Tin học (dùng ở Pháp và nhiều nước Tây Âu phổ biến từ đầu thập niên bảy mươi của thế kỉ XX), khoa học thông tin, khoa học và công nghệ thông tin, gần đây là công nghệ thông tin (với thuật ngữ "công nghệ" hiểu theo nghĩa rộng là tổng thể các quan niệm và phương pháp khoa học, các công cụ và giải pháp kĩ thuật được sử dụng trong một lĩnh vực nào đó). Trong phạm vi giáo dục thì tên gọi Computing được dùng một cách nhất quán từ lần đầu tiên năm 1968 cho đến lần gần đây nhất năm 2001 bởi Hiệp hội máy tính ACM và hội máy tính IEEE khi xác định chương trình học của ngành, dù nội dung đã liên tục thay đổi. Tôi nghĩ tên gọi *Tin học* (Informatics) theo nghĩa là "Khoa học về thông tin và các hệ thống xử lí thông tin bằng công nghệ máy tính và truyền thông" được nhiều chuyên gia dùng hiện nay là có tính khoa học, vừa gọn và vừa đủ khái quát để bao hàm các nội dung mà tên gọi kể trên đề cập đến.

Máy tính điện tử ra đời trên cơ sở kết hợp nhiều thành tựu của nhiều ngành khoa học và kĩ thuật khác nhau, đặc biệt là toán học và kĩ thuật điện. Vì thế trong những

năm năm mươi và sáu mươi của thế kỉ XX Khoa học máy tính thường có hai nội dung tách biệt: toán học tính toán và kĩ thuật máy tính. Sự phát triển trong những năm sáu mươi và bảy mươi để dẫn đến quan điểm kết hợp một cách liên tục và nhất quán hai phần kiến thức về phần cứng và phần mềm, đồng thời khái niệm "tính toán" được mở rộng theo nghĩa "xử lí thông tin" để hình thành một ngành khoa học thống nhất, dù vẫn mang tên gọi cũ Computer Science hay lấy tên gọi mới Informatique. Những năm bảy mươi, rồi đến những năm tám mươi, khi bắt đầu phát triển máy vi tính, công cuộc tin học hoá trong các lĩnh vực sản xuất, kinh tế và xã hội được phổ biến nhanh chóng và nhu cầu ứng dụng mở rộng trong mọi lĩnh vực đòi hỏi phát triển nhiều nội dung mới về cơ sở dữ liệu và hệ thống quản trị dữ liệu, về công nghệ phần mềm,... Trong bối cảnh đó, nhóm tác nghiệp của ACM/IEEE khi nghiên cứu để đổi mới chương trình Computing đã thấy cần thiết phải có một triết lí cho cách nhìn ngành này như một ngành khoa học độc lập, có đối tượng và phương pháp đặc thù, có những khối kiến thức với nội dung được xác định rõ ràng: Đối tượng là hệ thống xử lí thông tin; Phương pháp là sự kết hợp ba yếu tố: lí thuyết, thực nghiệm và công nghệ. Lí thuyết được phát triển trên cơ sở các phương pháp Toán học; thực nghiệm (như đối với các khoa học thực nghiệm khác) là tiến hành nghiên cứu trên các quy mô thu được thông qua trừu tượng hoá và kết quả phải được thử nghiệm trong thực tế; công nghệ với nội dung chính là thiết kế các hệ thống thực hiện. Các khối kiến thức của ngành bao gồm: kiến trúc máy tính, các hệ điều hành, các thuật toán và cấu trúc dữ liệu, các ngôn ngữ và phương pháp lập trình, cơ sở dữ liệu và tìm kiếm thông tin, tính toán số và kí hiệu, trí tuệ nhân tạo và người máy, giao diện Người – Máy, công nghệ phần mềm và các vấn đề đạo đức và xã hội của tin học. Thập niên chín mươi đánh dấu một bước chuyển to lớn: công nghệ máy vi tính và công nghệ viễn thông phát triển một cách nhanh chóng, tạo cơ sở hình thành và phát triển không ngừng các mạng máy tính – viễn thông, các siêu xa lộ thông tin, xây dựng trên thực tế các kết cấu hạ tầng thông tin quốc gia và toàn cầu, làm thay đổi bộ mặt kinh tế, xã hội trên thế giới và trở thành một yếu tố văn hoá mới của xã hội hiện đại. Những chuyển biến to lớn đó đã làm cho ngành khoa học của chúng ta được liên tục bổ sung những chủ đề mới của thế giới hiện đại như: các hệ thống thông tin, quản trị thông tin, an toàn và bảo mật thông tin, kĩ thuật mạng, công nghệ web, các hệ tri thức, công nghệ tri thức và trí tuệ nhân tạo,... Những biến chuyển này đã phần nào được phản ánh kịp thời trong chương trình Computing 2001 do nhóm tác nghiệp ACM/IEEE đề xuất và bổ sung gần đây. Chắc chắn sự phát triển chưa thể dừng lại ở đây. Nhiều công nghệ mới đang được nghiên cứu và đầy hứa hẹn như máy tính lượng tử, máy tính sinh học, truyền thông với tốc độ ánh sáng. Các kiểu cấu trúc tính toán mới như: tính toán mạng, tính toán song song, các ứng dụng mới đòi hỏi phát triển các nghiên cứu mới về tri học (Cognitive Science), các phương pháp phát hiện tri thức, mô phỏng hành vi của các hệ phức tạp bằng máy tính,... tất cả những điều đó đòi hỏi nhiều phát triển mới của tin học trong tương lai.

Có những người hỏi: Tính chất của công nghệ thông tin thì ai cũng rõ. Đó là một công nghệ đang có tác động lớn đến mọi đời sống, kinh tế, xã hội và văn hoá loài người. Nhưng ngoài ra, tin học có đóng góp gì mới vào nhận thức của con người về thế giới? J. Gruska, một nhà khoa học (Tiệp Khắc cũ) có một nhận xét rất đáng chú ý:

Một công nghệ dù mới và đặc sắc đến đâu, nó chỉ có thể tác động rộng lớn đến sự phát triển của khoa học và xã hội khi nó mang đến một cách nhìn mới, một cách nghĩ mới cho nhận thức của con người về thế giới. Và ở đây, cái mới của tin học mang đến là một phương pháp mới, tăng thêm sức mạnh cho các phương pháp vốn có của khoa học là nghiên cứu lí thuyết và nghiên cứu thực nghiệm. Với lí thuyết ta tìm ra tri thức mới bằng chứng minh lí luận, với thực nghiệm ta có tri thức bằng kiểm chứng trong thực tế. Nhưng có những chứng minh lí luận mà tư duy lôgic của con người không đi đến tận cùng vì nó quá đỗi phức tạp, có những "thực tế" mà ta không dễ gì gặp được hoặc tạo ra được để tiến hành kiểm chứng. Trong những trường hợp đó, với các mô phỏng các quá trình lập luận lôgic trên máy tính hoặc mô phỏng các quá trình "thực tế" phức tạp bằng ngôn ngữ của xử lí thông tin rồi thực hiện cũng trên máy tính, tin học cho ta những giải pháp hữu hiệu. Như vậy, cái mới mà tin học đóng góp vào nhận thức khoa học là khả năng mở rộng tầm nhìn và sức mạnh các phương pháp lí thuyết và thực nghiệm vào những địa hạt phức tạp, không chỉ bằng năng lực xử lí thông tin to lớn của máy tính, mà còn bằng những quan niệm và phương pháp mới như mô phỏng, xây dựng các mô hình xử lí thông tin (khác với các mô hình toán học và mô hình vật lí), hiển thị trợ giúp tư duy hình ảnh,... Nhiều kết quả mới và đặc sắc trong các lĩnh vực khoa học được phát hiện bằng phương pháp mới nói trên của tin học. Ví dụ điển hình như việc chứng minh bài toán bốn màu (định lí nói rằng mọi đồ thị phẳng đều tô được bằng không quá bốn màu) trong toán học và đặc biệt là phát hiện hành vi "hỗn độn" tất định của các hệ động lực phi định tính để nghiên cứu những đối tượng vốn có bản chất rất phức tạp của tự nhiên, kinh tế và xã hội – một hướng mới của khoa học về các hệ thống phức tạp đang được phát triển mạnh như hiện nay.

Tác động to lớn của tin học đối với việc chuyển biến kinh tế, xã hội hiện nay là điều mà ai cũng rõ. Qua vài dòng giới thiệu sơ lược trên đây, tôi hi vọng chúng ta có một cái nhìn đầy đủ hơn về tin học như là một ngành khoa học, tuy còn trẻ và chưa định hình rõ, nhưng tràn đầy sức sống và hứa hẹn tương lai phát triển mà ta chưa lường hết được, chắc chắn sẽ đóng góp cho con người nhiều ý tưởng mới, quan niệm mới và phương pháp mới để nhận thức càng sâu sắc hơn và hành động càng thông minh hơn trong thế giới và cuộc sống vốn đầy những phức tạp và bí ẩn.