

# Une révolution permanente

Née au milieu du xx<sup>e</sup> siècle avec les premiers calculateurs électroniques, l'informatique n'a cessé de se développer, touchant tous les secteurs d'activité.

> PAR GÉRARD BERRY, INFORMATICIEN, MEMBRE DE L'ACADEMIE DES SCIENCES, TITULAIRE DE LA CHAIRE « INFORMATIQUE ET SCIENCES NUMÉRIQUES » AU COLLÈGE DE FRANCE

© C. LEBEDINSKY/INRIA

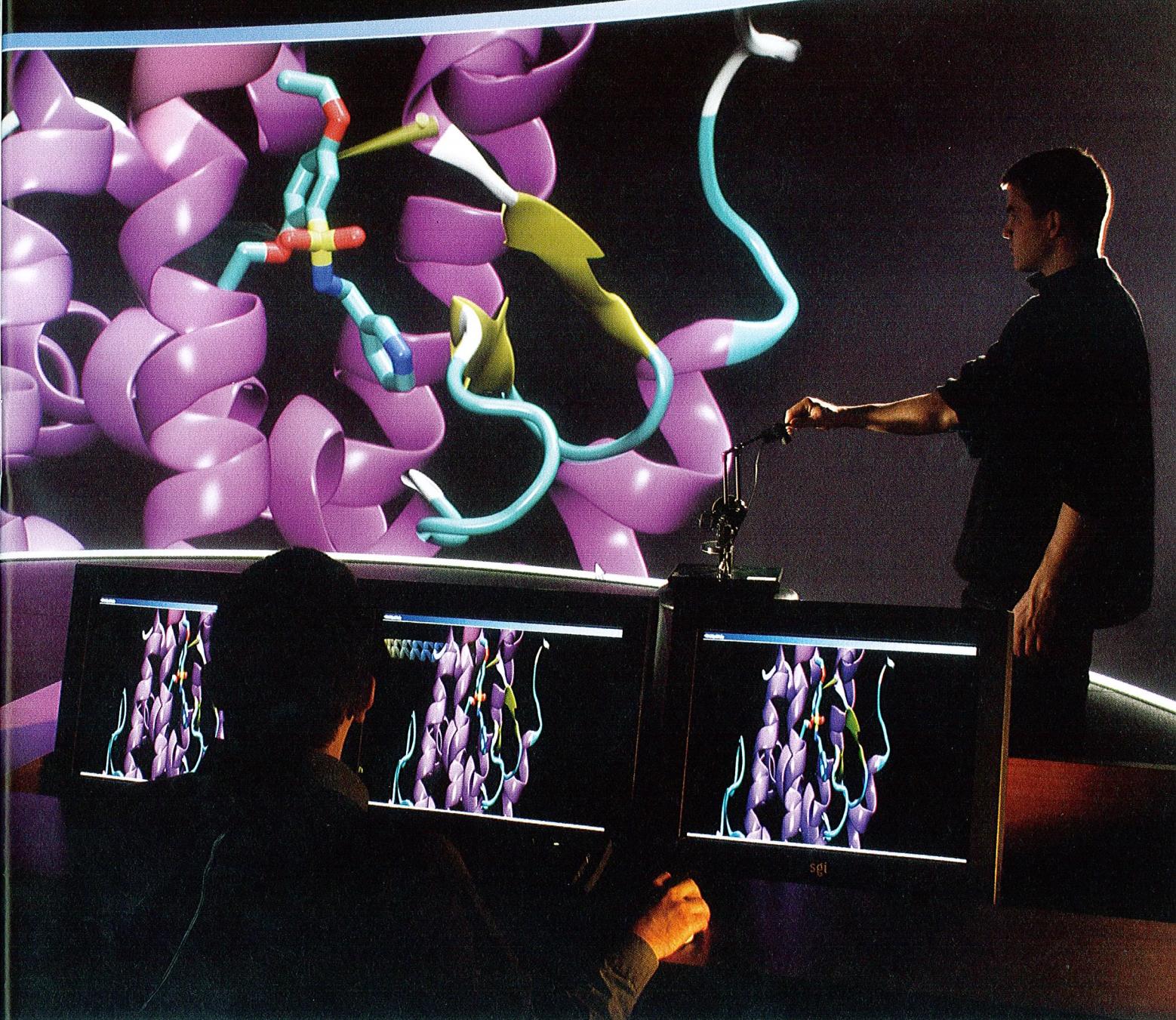
L'informatique recouvre la science, l'industrie et l'usage du calcul automatique en machine sous toutes ses formes, en incluant sa conception, sa réalisation et sa maîtrise. Dans ses principes, elle est apparue dès que les hommes ont ressenti le besoin de calculer de façon systématique pour prédire les saisons et les éclipses, pour savoir quand faire les semaines, quand vénérer les dieux, etc. Ce besoin précède certainement l'écriture : de très anciennes tablettes portent déjà la trace de calculs sophistiqués, comme celui de la racine de 2 en base 60 par les Babyloniens ; la numération binaire/octale était connue des Chinois 3 000 ans av. J.-C. ; les Indiens savent depuis longtemps manipuler les nombres positifs et négatifs, comme le montrent des textes de Brahmagupta vers 620, etc. Il s'agissait toujours d'établir des méthodes de calcul mécanique ne demandant plus d'intelligence une fois comprises.

Comme le suggère Gilles Dowek dans son livre *Les Métamorphoses du calcul* (Le Pommier, 2007), les mathématiques sont probablement nées du besoin d'établir la correction de calculs connus, les théorèmes de Thalès ou de Pythagore justifiant, par exemple, des calculs de longueurs géométriques. En étudiant la validité des raisonnements intuitifs, les Grecs ont aussi commencé à systématiser la logique, notamment à travers le syllogisme, et compris l'importance des paradoxes logiques. Tout cela reste fondamental dans l'informatique moderne.

**La systématisation des algorithmes.** Techniquement, le cœur conceptuel de l'informatique est la notion d'algorithme, ou procédé de calcul purement mécanique pour obtenir le résultat cherché. Le mot ne vient pas du grec, mais du nom du savant persan Al Khuwarizmi (783-850), qui a importé et systématisé les calculs indiens, puis apporté lui-même des nouveautés fondamentales ; le mot « algèbre »

vient d'ailleurs d'*al jabr*, du nom de son traité sur la solution des équations par changement de côté des termes d'une égalité. Ses travaux n'ont été connus que beaucoup plus tard en Occident. Ils ont été importés notamment par Léonard de Pise, dit Fibonacci (1175-1250), qui a introduit des algorithmes fondamentaux, dont le calcul itératif de sa fameuse suite. Mais l'algorithmique est restée longtemps inconnue chez nous ; au xv<sup>e</sup> siècle, les peu commodes chiffres romains étaient encore la règle.

Des abaque, tables, bouliers et règles de tous types furent développés. Il y eut quelques tentatives de développement de machines à calculer mécaniques, comme la machine de Pascal, mais sans lendemain. Il faut également citer le métier à tisser de Jacquard comme étant la première machine associant information et travail matériel à grande échelle, avec un système de cartes perforées qui allait perdurer jusqu'aux premiers ordinateurs électroniques.



**Le xix<sup>e</sup> siècle et les fondements des mathématiques.** À cette époque, le Britannique George Boole a compris que les raisonnements logiques de base peuvent se ramener à des calculs algorithmiques sur vrai/faux ou 0/1, en évacuant toutes les considérations pseudo-philosophiques qui encombraient le sujet depuis longtemps. Cette découverte fut essentielle, car le cœur véritable de l'informatique est la conjonction calcul numérique/décision logique. À la même période, le mathématicien allemand Georg Cantor a inventé et décrit la théorie des ensembles, proposant une organisation informationnelle des concepts mathématiques et systématisant des calculs et notations jusque-là laissés intuitifs. Il a montré que les nombres rationnels sont exactement aussi nombreux que les nombres entiers, alors que les nombres réels sont strictement plus nombreux. L'argument diagonal qu'il a employé pour ce faire joue

**ADA LOVELACE IDENTIFIA LA NOTION DE BUG**

toujours un rôle fondamental en informatique. Mais sa théorie n'était pas cohérente, car elle comportait un paradoxe majeur dû au même argument diagonal : l'ensemble de tous les ensembles devait avoir strictement plus d'éléments que lui-même. Entre 1820 et 1850, le Britannique Charles Babbage essaya de construire des machines mécaniques pour calculer notamment des polynômes, mais sans y

parvenir. Toutefois, Ada Lovelace, son assistante, pressentit que programmer cette machine allait être difficile, et identifia déjà la notion de *bug*. L'histoire retient Babbage comme le concepteur du premier ordinateur mécanique, inachevé, et Ada Lovelace comme celle qui écrivit les premiers programmes, jamais exécutés. Enfin, en 1900, l'Allemand David Hilbert changea de niveau de réflexion en posant son dixième problème, de nature purement algorithmique : étant donné un polynôme à coefficients entiers, existe-t-il un

#### ▲ Plongée au cœur du vivant.

Cette manipulation virtuelle d'une molécule à l'intérieur d'une protéine, représentée elle aussi en 3D, est effectuée à l'aide d'un ordinateur et d'un bras « à retour d'effort ». La visualisation interactive offre de multiples applications, notamment dans le domaine médical.

algorithme pour décider s'il a une racine parmi les nombres entiers ? Avant même de vouloir programmer un algorithme, il faut savoir s'il existe ; la réponse, négative, n'a été connue que dans les années 1970.

**La théorie de la calculabilité.** Un problème comme celui de Hilbert n'était pas bien posé, car la notion d'algorithme restait intuitive. Elle n'a été formalisée que dans les années 1930 avec le travail des logiciens américains Alan Turing et Alonzo Church. Turing a introduit une machine abstraite rudimentaire pour étudier la notion de calculabilité. Une machine ●●●

## ••• La dualité machine-langage est toujours d'actualité

de Turing M se compose d'une tête de lecture-écriture, d'un ruban et d'un jeu fini d'instructions très élémentaires. Elle prend en entrée des données D écrites sur la bande et exécute le calcul souhaité, en séquençant ses instructions selon les données. Turing a montré que ses machines pouvaient faire tous les calculs classiques des mathématiques. Mais une machine M n'a qu'un seul programme câblé. Turing a ensuite construit une machine U spéciale, dite universelle, qui introduit la notion fondamentale de programme enregistré et peut simuler tous les calculs de toutes les machines M directement câblées sur toutes les données D. Un ordinateur moderne n'est rien d'autre qu'une machine électronique universelle à programme enregistré, mais bien plus efficace.

Turing a ensuite montré qu'il n'existe aucune machine A prenant comme donnée le programme P d'une machine M et la donnée D, et décidant en temps fini si M va s'arrêter sur D ou non. Quoique techniquement très simple, ce résultat phare d'indécidabilité a établi d'un coup les limites de l'informatique : sa faculté de raisonnement sur elle-même est bornée. Il s'étend à l'indécidabilité de beaucoup d'autres problèmes, par exemple celui de l'égalité de deux machines. Le contourner en déterminant ce qu'on peut quand même décider d'utille a été l'un des grands objectifs ultérieurs de la logique formelle.

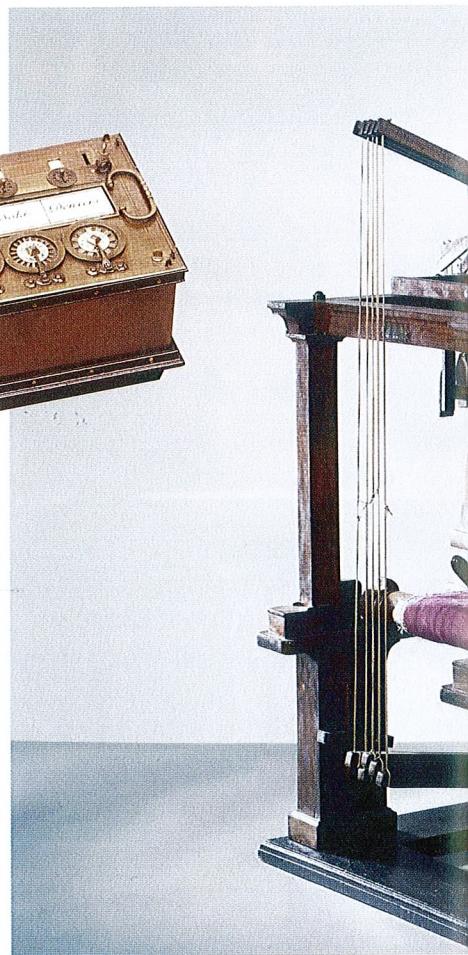
Church s'est davantage intéressé aux langages ; il a introduit en 1936 le  $\lambda$ -calcul ou calcul des fonctions. Il a montré que son langage ultra-dépouillé avait la même puissance que la machine de Turing. Cela demande de bien formaliser la notion de fonction récursive, qui est définie à partir d'elle même, comme la factorielle. Church a émis sa célèbre thèse que tous les formalismes algorithmiques généraux sont et seront à l'avenir équivalents à la machine de Turing et au  $\lambda$ -calcul. La dualité machine/langage ainsi introduite est toujours d'actualité. Ce qui est vraiment étonnant, c'est que le  $\lambda$ -calcul est devenu aussi la base des plus beaux langages de programmation généraux : ML, Caml, Scheme, etc.

**Les tout premiers ordinateurs.** Les choses se sont précisées après la Seconde Guerre mondiale, avec l'introduction des premiers ordinateurs commerciaux dans



▲ La Pascaline, première calculatrice mécanique. Elle fut inventée par Blaise Pascal en 1642, et ne pouvait effectuer que des additions et des soustractions.

► **Métier à tisser de Basile Bouchon (1725).** Bouchon inventa en 1728 la carte perforée, qui automatise le tissage des étoffes à motifs. Ce dispositif sera perfectionné par Falcon, puis par Jacquard, en 1790, qui révolutionnera l'industrie lyonnaise.



les années 1950, rapidement accompagnée de celle des premiers langages de programmation : Fortran (*formula translator*) et Cobol (*common business oriented language*). Les applications étaient de deux types : calcul numérique, surtout militaire, et applications de comptabilité et de gestion. On pensait alors que quelques gros ordinateurs suffiraient à résoudre tous les problèmes du monde. Pourtant, la mémoire et la puissance de calcul mondiales étaient bien inférieures à celles du plus petit des téléphones actuels !

Les années 1950-1960 ont vu la naissance de la théorie de l'information de l'Américain Claude Shannon, qui quantifie la notion même d'information, comptée en bits. Elle reste une théorie fondamentale, aussi bien pour les télécommunications que pour le codage efficace de sons, d'images ou de films. Ces années ont aussi été celles d'une progression considérable de la science des algorithmes, de la naissance de la théorie des langages et des automates, de la compréhension par le Néerlandais Edsger Dijkstra des mécanismes de calculs menés en parallèle sur plusieurs processeurs et des problèmes de synchronisation associés, enfin des débuts de l'intelligence

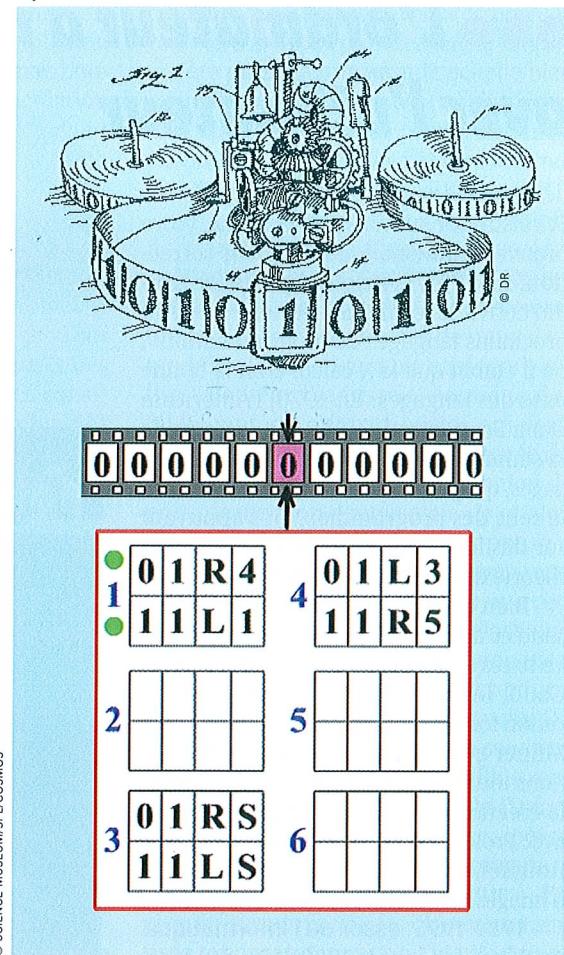
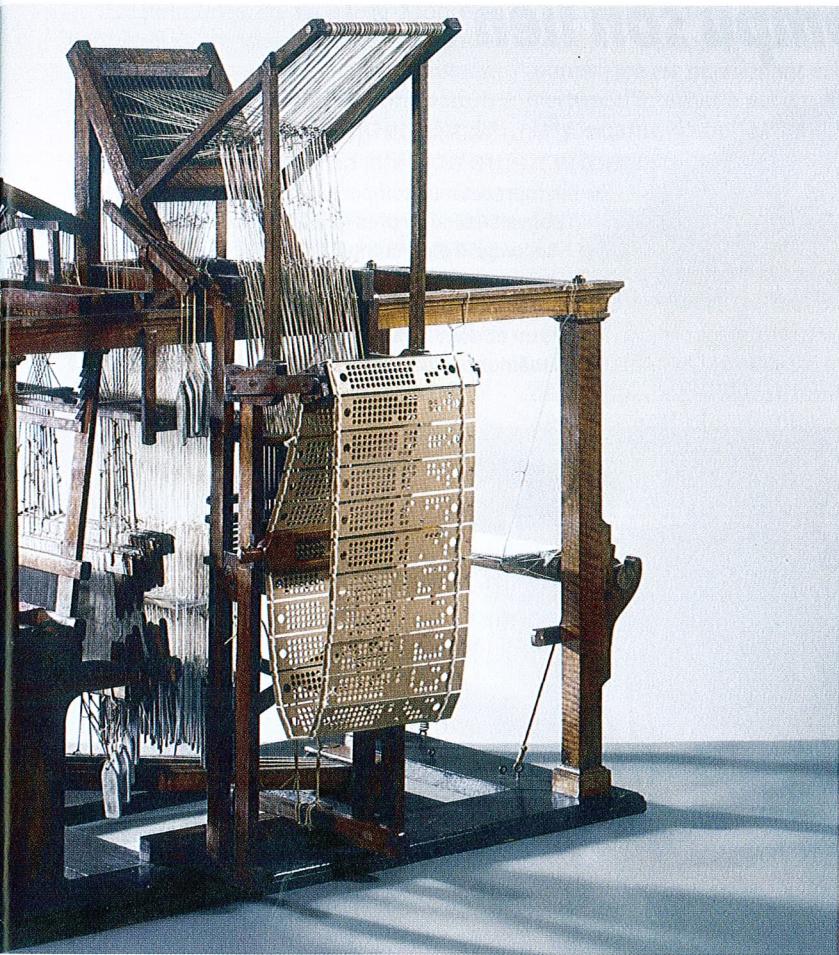
### CHURCH

INTRODUIT  
EN 1936 LE  
 $\lambda$ -CALCUL

artificielle avec le calcul symbolique introduit par l'Américain John McCarthy dans le langage Lisp.

Mais la révolution informatique n'a vraiment commencé que dans les années 1970. En 1971, avec l'Intel 4004, apparut le microprocesseur, brique de base de tous les ordinateurs actuels. En 1975, l'Américain Gordon Moore, cofondateur d'Intel, formula sa célèbre loi qui allait conditionner tout le futur de l'informatique : le nombre de transistors par unité de surface allait doubler tous les deux ans. Cette loi est restée parfaitement valable. On ignore souvent qu'elle ne résulte pas d'une constatation *a posteriori*, mais au contraire d'une décision *a priori* : l'industrie s'organise pour la respecter en construisant des robots et des usines adaptées, génération après génération. D'autres lois du même type sont valables pour la taille mémoire, la taille des disques, la vitesse des communications, etc.

Au milieu des années 1970 sont apparus les mini-ordinateurs, qui pouvaient être possédés par une entreprise ou une équipe de recherche de taille moyenne. L'informatique a ainsi changé de main et gagné une grande variété d'applications, l'ordinateur devenant un instrument



scientifique à part entière. Pour faire fonctionner ces petites machines, il a fallu créer des systèmes d'exploitation et des langages de programmation plus légers et plus efficaces; de là sont nés le langage C et le système d'exploitation Unix, qui a donné Linux, OS X d'Apple, etc. Sont aussi apparues les problématiques liées aux réseaux de grande échelle avec Arpanet, le courrier électronique et les transferts de fichiers.

**Zoom sur la théorie de la complexité.** Sur le plan théorique, la communauté s'est structurée en deux grands domaines: l'algorithmique et la théorie de la programmation.

L'algorithmique a d'abord bénéficié du travail encyclopédique inestimable de l'Américain Donald Knuth, qui a sistématisé l'étude de la complexité en temps et en mémoire de la plupart des algorithmes connus. Les problèmes les plus simples demandent un temps de calcul assez faible: un circuit met un temps  $\log(n)$  à additionner deux nombres de  $n$  chiffres; il faut un temps  $n$  pour balayer de gauche à droite une liste de taille  $n$ , un temps  $\log(n)$  pour trier une liste de taille  $n$ , un temps  $n^3$  pour multiplier deux matrices

$n \times n$  selon la méthode classique, etc. On dit qu'ils demandent un temps polynomial.

En 1973, l'Américain Stephen Cook a posé un jalon essentiel en caractérisant la notion centrale de NP-complétude. Un problème NP-complet est un problème

si compliqué qu'il ne semble pouvoir être résolu que par l'exploration combinatoire d'un nombre exponentiel de cas afin d'être sûr de trouver la solution, l'exploration de chaque cas étant possible en temps polynomial. Par exemple, comment minimiser le trajet d'un voyageur de commerce qui va de ville en ville, ou comment organiser les emplois du temps d'un lycée sont des problèmes NP-complets. Il semble qu'il faille explorer toutes les possibilités pour trouver la solution exacte de façon garantie. Il existe en pratique des centaines de problèmes essentiels de ce type. Ils sont tous de complexité équivalente, au sens où tout problème NP-complet X est tel que tout autre problème NP-complet Y peut se résoudre en le transformant de façon polynomiale en problème de type X. Ainsi, toute solution au problème du voyageur de commerce peut être transformée en un calculateur d'emplois du temps.

### L'ORDINATEUR EST DEVENU UN INSTRUMENT SCIENTIFIQUE

▲ La « machine » de Turing, ce modèle de machine théorique, formulé par Alan Turing en 1930, est capable de simuler toute opération réalisable par n'importe quel processeur.

Cependant, la nécessité d'examiner tous les cas possibles n'est pas encore prouvée; elle fait l'objet de la conjecture fondamentale  $P \neq NP$  de Cook, qui affirme que les problèmes NP-complets ne peuvent pas être résolus en temps polynomial. Cette conjecture reste l'un des grands problèmes mathématiques actuels. Son étude s'est récemment affinée. On sait maintenant qu'il y a plusieurs catégories de problèmes NP-complets: ceux pour lesquels il existe des heuristiques efficaces, c'est-à-dire des méthodes permettant de trouver rapidement de bonnes solutions approchées, et ceux pour lesquels on ne peut pas trouver d'heuristiques. Les premiers sont évidemment bienvenus partout; les seconds sont utiles en cryptographie.

La théorie de la programmation s'est consacrée à l'augmentation de la sûreté des programmes, toujours menacés par les abominables bugs. Elle a travaillé à la définition de meilleurs langages et ●●●

## ●●● L'ordinateur a rompu son lien avec l'utilisateur

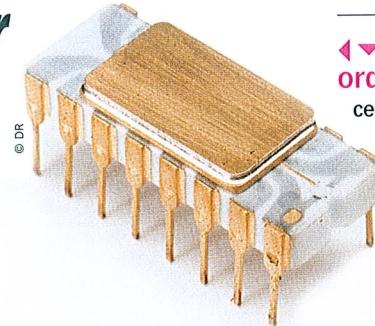
développé des méthodes pour manipuler et raisonner sur les programmes, voire prouver mathématiquement leur correction. En 1967, le Britannique Peter Landin écrivit le formidable article «Les 700 prochains langages de programmation», où il établit que le  $\lambda$ -calcul était la bonne base des langages. En 1970, l'Américain Dana Scott introduisit les fondements de la sémantique dénotationnelle des langages, qui permet de définir formellement le sens des programmes, en s'appuyant sur des idées mathématiques superbes : théorie des treillis, topologies faibles, etc.

Bien d'autres inventions ont été présentées au cours de cette période : citons les bases de données relationnelles d'Edgar Codd, les premiers systèmes de vérification formelle de programmes par Robin Milner et son équipe, des expérimentations multiples en intelligence artificielle, la création de la programmation logique avec Prolog d'Alain Colmerauer et Philippe Roussel, la naissance du traitement d'images, du graphique et de la robotique.

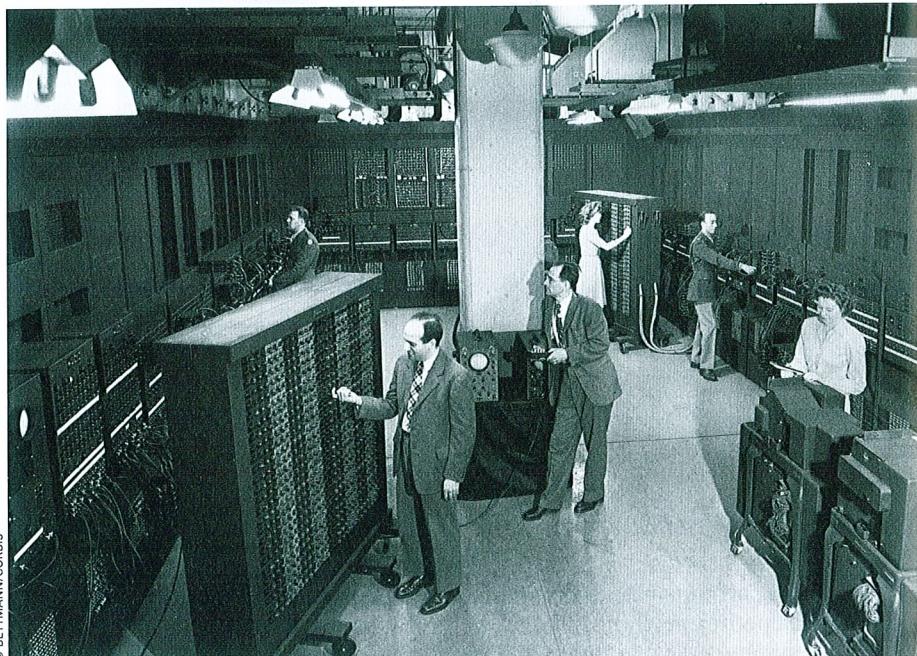
**1980-1995 : essor de l'informatique.** Au début des années 1980, la loi de Moore poursuivant ses effets, il a été possible de fabriquer des stations de travail individuelles, reliées par des réseaux rapides et équipées de nouvelles interfaces (souris, écrans graphiques), puis des ordinateurs personnels destinés à des amateurs éclairés. Cela ne s'est pas fait sans cahots : les premiers PC étaient lents, les systèmes d'exploitation frustres et frustrants, et les applications très imparfaites. Mais le mouvement a continué inéluctablement jusqu'à nos jours, où les micro-ordinateurs n'ont plus de gros problèmes de puissance ni d'accessibilité.

Dans le même temps, l'ordinateur a rompu son lien direct avec l'utilisateur pour s'enfouir dans les objets. Cela a commencé avec les avions de chasse dits instables, c'est-à-dire impossibles à piloter sans assistance informatique, puis s'est étendu aux avions civils avec l'Airbus A320, au contrôle des centrales nucléaires, à celui des usines par une race spéciale d'ordinateurs appelés « automates programmables », ou à celui des premiers robots autonomes.

Les applications ont explosé, touchant des secteurs jusque-là vierges : jeu graphique interactif, traitement de texte général ou scientifique, gestion d'entre-



◀ ▶ L'Intel 4004, brique de base des ordinateurs actuels. Conçu dans les années 1960, ce microprocesseur comportait 2 300 transistors et pouvait exécuter près de 60 000 opérations par seconde. Il était comparable à l'Eniac (ci-dessous), premier ordinateur électronique utilisé dans les années 1950, aux États-Unis, qui occupait plus de 20 m<sup>2</sup> et dont le programme n'était pas enregistré en mémoire : il devait être câblé manuellement.



prises avec le tableau et les logiciels pour PME, acquisition de données physiques par connexion de capteurs, conception assistée par ordinateur (CAO) des objets, simulation numérique des phénomènes

physiques, etc. L'algorithmique y a trouvé de nouveaux champs d'action : graphique 3D, manipulation d'objets géométriques, calcul symbolique sur les formules mathématiques, etc.

Mais les PC restaient difficiles à programmer, et les bugs commençaient à toucher d'autres personnes que les auteurs des programmes. Pour progresser, il fallait que les scientifiques amplifient leurs efforts sur la compréhension profonde de l'activité de programmation et de validation des programmes. Cette période a été fort riche en création de nouveaux concepts théoriques et pratiques : typage et vérification formelle pour trouver les bugs avant d'exécuter les programmes, programmation synchrone des systèmes embarqués, développement de l'algorithmique graphique et du traitement d'image,

perfectionnement des interfaces homme-machine avec les premiers concepts de liens entre informations aboutissant à la définition du Web, etc.

La limite de cette période était claire : les utilisateurs restaient soit des professionnels soit des amateurs éclairés. Ainsi, internet était déjà bien connu des chercheurs en 1980, mais pas du grand public, qui n'y avait pas accès. Les radiotéléphones existaient, mais ils étaient lourds et chers. Les CD régnaient en maîtres sur la musique, les cassettes sur la vidéo. Les appareils photographiques avaient encore des pellicules.

**1995-2009 : une période de grands bouleversements.** Effets maximaux de la loi de Moore conduisant les circuits vers le milliard de transistors, arrivée de l'internet grand public, début de l'informatisation systématique des objets et progrès majeurs en théorie de l'information (conduisant à l'ADSL, aux turbocodes atteignant les limites théoriques du codage, etc.), nombreux sont les progrès qui ont bouleversé notre vision du quotidien.

La poursuite de la loi de Moore a permis de s'affranchir des limites de vitesse et de taille des machines tout en baissant leur prix, de réduire la consommation d'énergie en intégrant de plus en plus de fonctions auparavant distinctes sur une puce, le basculement rapide de technologies classiques en technologies numériques nomades (notamment pour l'audiovisuel) en introduisant des mémoires permanentes de grande taille. À l'autre bout du spectre, les superordinateurs ont évolué vers les milliers de processeurs ; les grilles de calcul et le calcul «en nuage» sont nés de l'abondance des serveurs, et les fermes de données ont évolué vers des empilements de containers bourrés de microserveurs entourés de systèmes de climatisation.

La profondeur des transformations sociétales associées à cette explosion numérique est maintenant évidente. L'ouverture d'internet a permis une création foisonnante d'applications auparavant insoupçonnées : consultation de sites Web de toutes sortes, recherche généralisée d'informations, commerce en ligne, participation à des réseaux sociaux, à des échanges légaux ou illégaux de musique et de vidéo, développement collaboratif de logiciels libres, d'encyclopédies du savoir, etc. Tout cela a demandé une infrastructure et des développements algorithmiques considérables, utilisant de plus en plus des techniques probabilistes de «meilleur effort» à la place des techniques exactes jusque-là dominantes. Ce paradigme, où les

probabilités entrent en jeu pour assurer que cela marche «à peu près bien tout le temps», s'avère passer à une grande échelle bien mieux que les méthodes exactes classiques. Il est utilisé en particulier pour le téléphone portable, le routage des paquets internet et la transmission de films en pair-à-pair.

L'ouverture d'internet a aussi profondément transformé l'industrie, en propulsant au sommet de nouveaux acteurs comme Google, en créant de très nombreuses entreprises dotées de nouveaux modèles économiques, et en détruisant littéralement l'industrie classique des télécommunications filaires, qui était restée intellectuellement très séparée du reste de l'informatique et de la notion de nomadisme. Son dogme était «la qualité garantie», donc exactement opposé à celui de «meilleur effort» qui sous-tend l'internet et le portable. Malgré une certaine obstination, téléphone classique et Minitel n'allait pas résister bien longtemps, et l'industrie allait aboutir progressivement à la situation actuelle. De tels basculements telluriques ne sont évidemment pas dus au hasard. Ils illustrent bien à quel point la puissance de l'informatique peut être sous-estimée, même par de grands acteurs dans ce domaine.

**L'ère du «tout-informatique».** En plus du grand réseau, les objets informatisés font maintenant partie intégrante de notre paysage, du lecteur MP3 à l'appareil photographique numérique, ●●●



◀ **L'enfance du Mac.** Lancé en 1984 par l'Américain Steve Jobs, l'Apple Macintosh est le premier ordinateur utilisant une souris et une interface graphique au lieu d'une interface en lignes de commande où la communication utilisateur-ordinateur s'effectue en mode texte.

▼ **Tableau de bord de l'Airbus A320.** Aujourd'hui, l'informatique se dissimule dans les objets ou appareils les plus divers, comme dans cet avion, dont les commandes sont reliées à un ordinateur de gestion de vol.

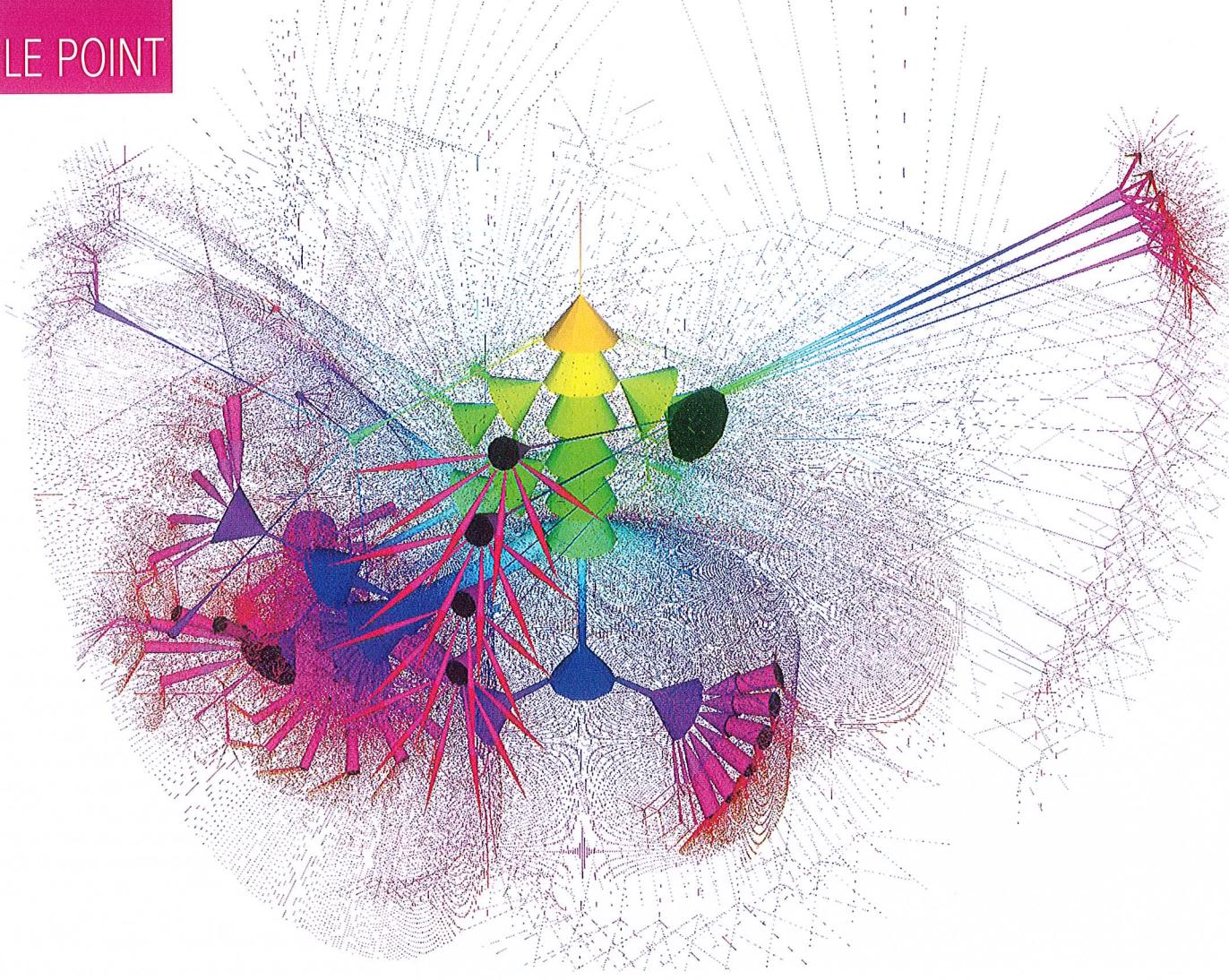


## FOCUS

# De la sécurité des logiciels

Si l'écriture a permis de moins utiliser la mémoire en la gravant dans la matière et l'imprimerie de la diffuser, internet offre la possibilité de délocaliser complètement la mémoire du monde, et donc de ne plus en avoir besoin chez soi. Les conséquences sont encore largement inconnues, mais la seule certitude est qu'elles seront considérables.

Dans tous les domaines, la qualité et la sécurité des logiciels deviendront de plus en plus essentielles. Certaines industries, comme l'avionique ou le ferroviaire, utilisent déjà des processus de certification externe bien établis. Cependant, la certification actuelle ne concerne que le processus de développement, non les circuits ou les programmes eux-mêmes. Il faut espérer que les progrès considérables de la vérification formelle de ces derniers pourront être exploités afin de certifier vraiment la fonctionnalité des circuits et des logiciels.



## ••• L'informatique ubiquitaire deviendra réalité

en passant par la voiture moderne ou la télévision en haute définition. Leurs interfaces vont devenir tactiles, gestuelles, vocales, et, qui sait, bientôt être directement reliées à l'activité cérébrale. Pourtant, le public ne réalise pas qu'il y a maintenant beaucoup plus de microprocesseurs dans les objets de tous les jours que dans les ordinateurs classiques.

Pour réaliser tous ces nouveaux systèmes, il a fallu beaucoup de nouvelles idées jointes à un approfondissement considérable de la science et de la technique informatique, qui n'est plus du tout artisanale et passe de plus en plus de temps à calculer sur elle-même. Les objets mécaniques sont conçus en CAO industrielle, avec des algorithmes géométriques et des simulations numériques extrêmement sophistiqués. Leurs circuits sont conçus et vérifiés en CAO électronique par des programmes informatiques extrêmement complexes, avec des coûts de développement devenus gigantesques. Les programmes sont écrits dans des langages de haut niveau et transformés graduellement en programmes machines très efficaces par des couches successives et très délicates de compilation et d'optimisation. Ils

sont parallélisés pour être exécutés encore plus efficacement sur des multiprocesseurs. Leur mise au point se fait à l'aide de débogueurs graphiques évolués, voire à l'aide de systèmes automatiques de preuves de propriétés. Les bases de données ou les moteurs de recherches gèrent des nombres gigantesques de requêtes, en utilisant de grandes fermes de machines organisées pour tolérer les pannes. La sécurité des communications et des données commence enfin à être prise au sérieux, même si elle laisse encore beaucoup à désirer, comme on le voit à travers les nombreuses attaques de sites ou de machines, qui exploitent savamment des bugs minuscules dans des systèmes informatiques réalisés «à l'arrache» pour gagner du temps de développement et de diffusion (navigateurs Web, par exemple).

**Une époque charnière de l'évolution informatique.** De nouveaux effets d'échelle apparaissent au niveau de toutes les composantes. La conception et la fabrication des circuits se transforment progressivement en vrai cauchemar. Même si la physique n'interdit toujours pas la poursuite de la loi de Moore, la capacité humaine de l'exploiter devient moins

claire. Le prix des usines croît avec l'intégration des circuits ; il atteint des sommets tels que de plus en plus d'acteurs historiques renoncent. Concevoir un circuit à des milliards de transistors devient aussi complexe que de concevoir un avion, avec la contrainte supplémentaire que *tout* doit être prêt le même jour, celui où la première fabrication est lancée. Les bugs de conception ou de fabrication des circuits sont devenus inévitables, et ils ne peuvent pas toujours être physiquement corrigés ; il faudra apprendre à vivre avec. Pour simplifier la conception, on s'oriente vers l'intégration d'un grand nombre de processeurs déjà maîtrisés sur chaque puce, sans vraie garantie qu'on saura les exploiter efficacement. De plus, si on est encore assez loin des limites physiques de taille des fils, on a atteint le mur de la chaleur : on ne peut plus augmenter la vitesse d'horloge des processeurs, sous peine d'atteindre des températures insupportables ou de vider rapidement les batteries. Ainsi, la réduction de la consommation électrique est devenue le sujet principal dans ce domaine.

L'ordinateur traditionnel pourrait bien être un objet en voie de disparition. La tour de bureau devient rare, l'ordinateur por-

### APPRENDRE À VIVRE AVEC LES BUGS

table est concurrencé par les «ordiphones» ou les tablettes qui se relient bientôt à la télévision ou au projecteur, et l'on ne sait pas de quoi l'avenir sera fait. Les moyens de communication physiques et d'accès au réseau deviendront de plus en plus variés: ADSL amélioré, fibres optiques, courants porteurs sur les fils électriques classiques, liaisons sans fil de diverses portées, etc. Leur interopérabilité généralisée les rendra invisibles à l'utilisateur, non sans cahots sur le chemin. Les objets informatiques vont se banaliser et communiquer entre eux sans même que nous en ayons connaissance. N'est-il pas absurde que deux voitures se percutent à un carrefour, sans même s'être prévenues mutuellement de leur présence? Est-il encore raisonnable de se ruer dans un embouteillage parfaitement connu du système de gestion d'une ville, ou bien qu'un panneau «Verglas» ne soit pas équipé d'un réel système de détection de verglas et d'un système de communication d'alertes aux véhicules ? Ce ne sont que quelques

### UNE EXTENSION À D'AUTRES SCIENCES

exemples parmi des milliers d'utilisations possibles. En bref, l'informatique ubiquitaire deviendra une réalité.

**Vers les sciences numériques.** Au xx<sup>e</sup> siècle, le physicien américain Eugene Wigner déclarait que les mathématiques avaient eu «une réussite insolente en physique», car leurs équations permettaient d'expliquer la plupart des phénomènes, et même de les prévoir théoriquement avant de pouvoir les réaliser. L'informatique continuera dans le même sens, mais avec une amplification majeure des possibilités et une extension à bien d'autres sciences. Les capacités humaines limitent la modélisation mathématique classique à des équations assez simples avec relativement peu de variables. La puissance des ordinateurs, les progrès des mathématiques appliquées ainsi que de l'algorithmique permettent de repousser ces limites en traitant des millions d'inconnues dans des équations bien plus complexes, tout en manipulant des objets géométriques également complexes. L'informatique s'applique à la mécanique des solides, des fluides et de leurs interfaces, à l'aérodynamique, etc. Et les instruments de mesure deviennent tous numériques : télescopes, caméras, imagerie médicale, etc.

De façon plus radicale, le point de vue informatique s'intègre dans des disciplines autrefois immunes : la génomique travaille sur des mots et graphes de très grande taille ; la botanique numérique fait pousser des plantes virtuelles parfaitement réalistes en étudiant la dynamique du méristème ; la bioinformatique se pose des problèmes allant de la compréhension de l'interaction géométrique en 3D des

protéines à la simulation d'une cellule complète ; les neurosciences computationnelles cherchent à savoir comment le cerveau traite l'information à tous les niveaux. Dans toutes ces nouvelles approches pluridisciplinaires, l'informatique ne jouera plus seulement un rôle de calculette de grand luxe. Elle apportera une vision abstraite des transferts d'information réalisés par les systèmes mécaniques, électriques ou biochimiques sous-jacents, qui deviendra centrale pour la compréhension de la dynamique des phénomènes complexes.

Nous avons voulu illustrer ici l'essence, les possibilités et l'impact de l'informatique moderne. Mais qu'en est-il du système d'enseignement, du primaire jusqu'au supérieur ? Pourquoi un étudiant, en 2010, peut-il sortir de Mathématiques spéciales en ayant eu une exposition au cœur scientifique de l'informatique limitée à quelques heures ? Lors des découvertes de l'électricité et des ondes, leurs bases scientifiques ont été rapidement vulgarisées et enseignées, de sorte que les citoyens puissent vivre au sein de la nouvelle société industrielle sans être perpétuellement effarés des technologies qui y étaient omniprésentes. Et nous nous enorgueillissons encore de la place de notre pays comme créateur des technologies majeures de cette époque. Ne doit-on pas tout tenter pour garder la même place dans un domaine aussi profondément révolutionnaire que l'informatique actuelle ? ●

### ◀ Visualisation de graphe de très grande taille.

▼ Dans un cybercafé. Avec le développement mondial des cybercafés, l'internet est devenu un phénomène de société.

▲ Une ardoise magique. L'iPad, le dernier-né des micro-ordinateurs d'Apple, est dépourvu de clavier, mais possède un écran tactile multitouche.

### SAVOIR +

- BERRY Gérard. *Pourquoi et comment le monde devient numérique*. Paris : Fayard, 2008 (coll. Leçons inaugurales du Collège de France).

