TEXTES ET DOCUMENTS POUR LA CLASSE

BIMENSUEL > N° 997 > 1er JUIN 2010

TDC

L'informatique

LE POINT > Une révolution permanente

DROIT > Au service des citoyens

POSTER > L'informatique de A à Z

+ 16 PAGES DE SÉQUENCES PÉDAGOGIQUES

SCÉRÉN CNDP

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

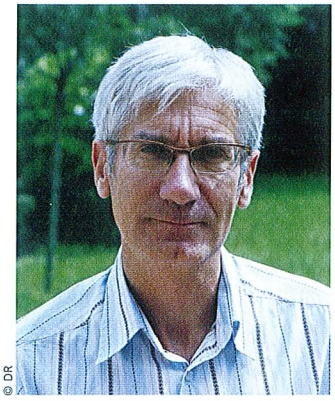
INRIA

4,70 € / ISSN 0395-6601

9 770395 660004 10997

755A3434

Soulever le capot...



© DR

> GUY BELZANE, RÉDACTEUR EN CHEF

Révolution numérique... Répétée *ad nauseam,* la formule a fini par perdre toute force, d'autant que la nouveauté portée par cette révolution s'est si vite et si aisément introduite dans nos vies quotidiennes que nous ne percevons plus guère, déjà, son caractère... révolutionnaire. Jamais sans doute les pratiques de masse (en dépit d'inégalités réelles mais somme toute marginales) n'ont été bouleversées avec une telle rapidité et une telle ampleur par un enchaînement lui-même fulgurant d'innovations technologiques. On s'épargnera ici la liste bien connue des offres et des usages inédits (les premières excédant d'ailleurs de beaucoup les seconds) introduits par la numérisation et la mise en réseau généralisées de l'information. On retiendra cependant un fait majeur : la disjonction entre une connaissance relativement confidentielle des bases scientifiques et techniques de l'informatique et une maîtrise très répandue de ses applications. Cet écart pose problème.

S'il n'est peut-être plus nécessaire aujourd'hui de connaître les rudiments de la mécanique automobile pour être un conducteur averti, il en va différemment de la circulation... de l'information, dont les enjeux – politiques, sociaux, culturels – sont tout autres. À l'heure où l'on déplore, non sans raison, la déficience d'une véritable culture scientifique en France, il serait opportun de se souvenir que l'informatique, avant d'être un formidable outil, est d'abord une science, justement. La question de son enseignement est heureusement (on allait dire : enfin) à l'ordre du jour. Elle suscite des débats, ce qui est bien normal. Certains clivages n'en paraissent pas moins artificiels, et il n'y a guère de sens à opposer approches scientifique et sociétale. C'est précisément à tenir ensemble ces deux fils que ce numéro de *TDC* s'attache et c'est tout le sens du partenariat mené, en cette occasion, avec l'Institut national de recherche en informatique et en automatique (Inria).

TDC

Directeur de la publication
Patrick Dion
Rédacteur en chef
Guy Belzane
Rédactrice en chef adjointe
Sylvie Gendrot
Rédaction
Marielle Chevallier,
Corinne Denailles,
Christiane Rebattet,
Christiane Yamada-Pédersen
Révision
Benoît Selleron
Iconographie
Pierre Philippon
Atelier graphique
Claire Salais

DIRECTION COMMERCIALE
Hélène Jourdain,
directrice,
Catherine Rastier
catherine.rastier@cndp.fr
Téléport 1 @ 4 – BP 80158
86961 Futuroscope Cedex

RENSEIGNEMENTS
SCÉRÉN-CNDP Abonnement
Tél. 03 44 62 43 98
Fax 03 44 12 57 70
abonnement@cndp.fr

CONCEPTION Agence 154
Création et direction artistique
Claire Salais
claire@liliklik.com

RÉGIE PUBLICITAIRE
Mistral Média
Directeur commercial
Luc Lehéricy

Directeur de publicité
David Bichot
365, rue de Vaugirard
75015 Paris
Tél. 01 40 02 99 00
Fax 01 40 02 99 01

IMPRIMERIE
Jouve
1, rue du Docteur-Sauvé
53100 Mayenne

PEFC

TDC est une publication du SCÉRÉN-CNDP
Téléport 1 @ 4, BP 80158
86961 Futuroscope Cedex
Tél. 05 49 49 78 78

TDC Rédaction
60, bd du Lycée 92170 Vanves
Tél. 01 41 23 04 21
tdc@cndp.fr

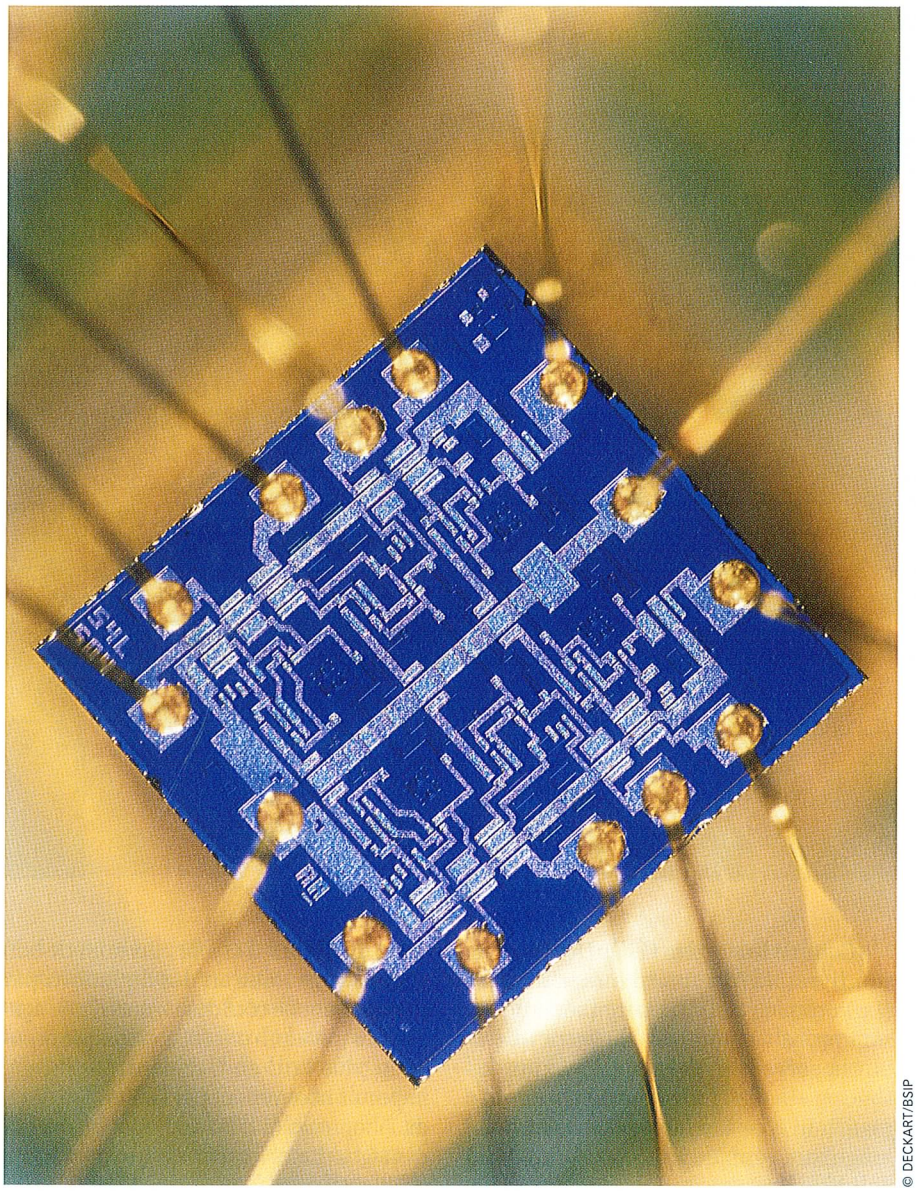
Les textes cités dans *TDC* le sont à titre documentaire : les opinions qu'ils peuvent exprimer doivent être appréciées de ce point de vue. Tous droits de reproduction, de traduction et d'adaptation réservés pour tous pays.
© SCÉRÉN-CNDP 2010
Dépôt légal juin 2010

UNE
Visualisation de champs de vitesse autour d'un avion Falcon (2009).
© Kaksonen/Inria

CE NUMÉRO A ÉTÉ COORDONNÉ PAR
Yannick Le Thiec,
François Rechenmann

ONT COLLABORÉ À CE DOSSIER
Jean-Pierre Archambault,
Emmanuel Baccelli,
Gérard Berry,
Marie-Christine Cuvelier,
Ionut Danaila,
Jean-Paul Delahaye,
Gilles Dowek,
Christine Leininger,
Daniel Le Métayer,
Thierry Viéville,
Benjamin Wack

L'informatique



© DECKART/BSIP

▲ **La puce électronique.** Souvent invisibles, les puces électroniques, encore appelées « circuits intégrés », n'en sont pas moins omniprésentes… au point que certains s'inquiètent déjà des dérives potentielles.

06/13 **LE POINT**
Une révolution permanente

14/15 **MATHÉMATIQUES**
Mathématiques et informatique

16/19 **DÉCRYPTAGE**
Un exemple de modélisation : le condensat de Bose-Einstein

20/21 **INTERVIEW**
« Un chemin initiatique vers l'abstraction »

22/24 **DROIT**
Au service des citoyens

25/28 **POSTER**
Zoom sur les sciences numériques

29/45 **PÉDAGOGIE**

INTRODUCTION

INFORMATIQUE
Tle S
Tout a un reflet numérique

Les ingrédients des algorithmes

46/47 **SUR LA TOILE**

48 **RESSOURCES**

50 **ABONNEMENT**

http://tdc.cndp.fr

Compléments en ligne sur le site internet de *TDC*

- L@ngues en ligne : japonais
- Ressources vidéo
- Liens avec l'actualité

Une révolution permanente

© C. LEBEDINSKY/INRIA

Née au milieu du XXe siècle avec les premiers calculateurs électroniques, l'informatique n'a cessé de se développer, touchant tous les secteurs d'activité.

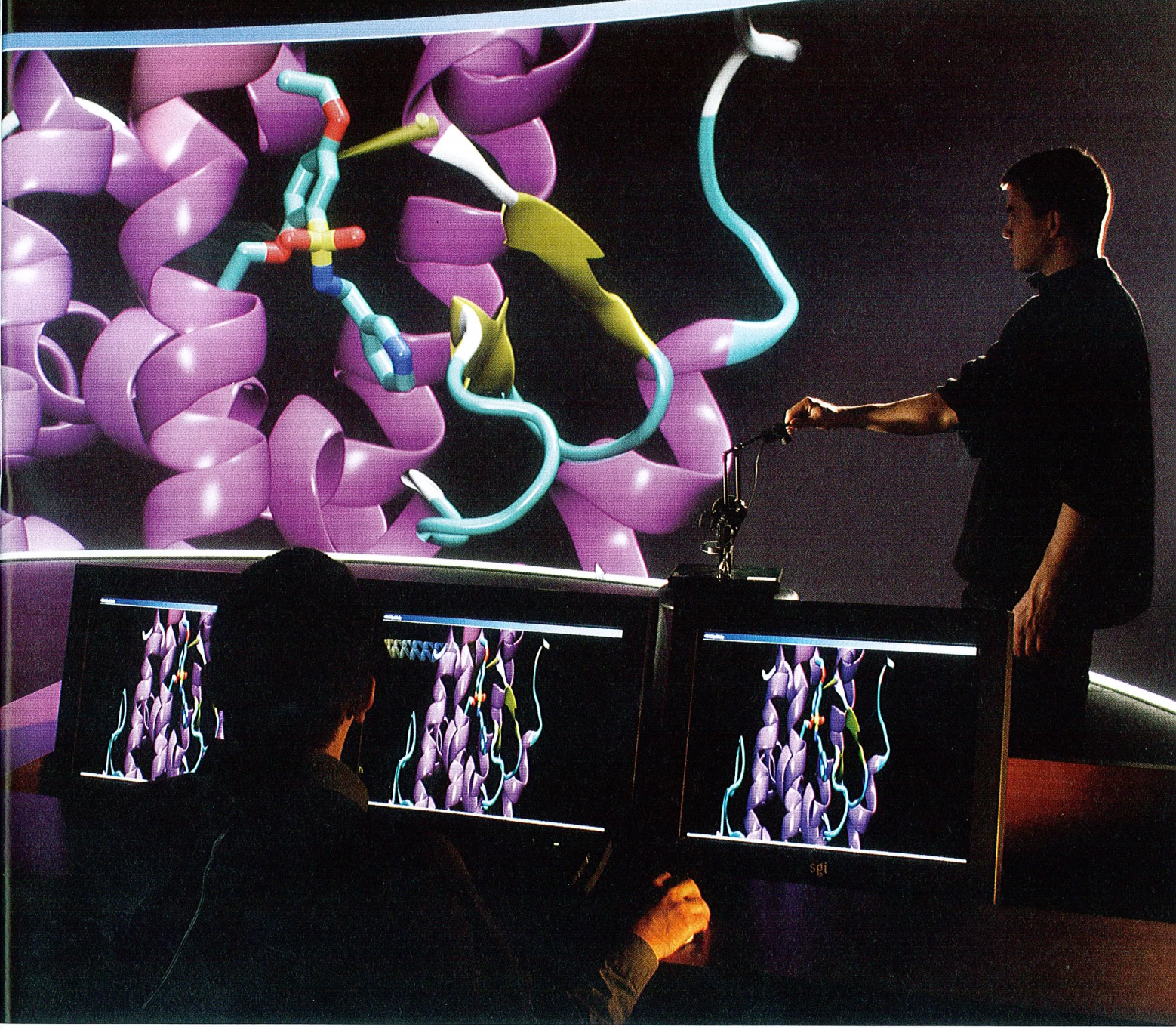
> PAR GÉRARD BERRY, INFORMATICIEN, MEMBRE DE L'ACADÉMIE DES SCIENCES, TITULAIRE DE LA CHAIRE « INFORMATIQUE ET SCIENCES NUMÉRIQUES » AU COLLÈGE DE FRANCE

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 semailles, 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 Fibonnacci (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 XVe siècle, les peu commodes chiffres romains étaient encore la règle.

Des abaques, 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.



▲ **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.**

Le XIXe 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 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 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.

ADA LOVELACE IDENTIFIA LA NOTION DE *BUG*

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'utile 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 λ-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 λ-calcul. La dualité machine/langage ainsi introduite est toujours d'actualité. Ce qui est vraiment étonnant, c'est que le λ-calcul est devenu aussi la base des plus beaux langages de programmation généraux : ML, Caml, Scheme, etc.

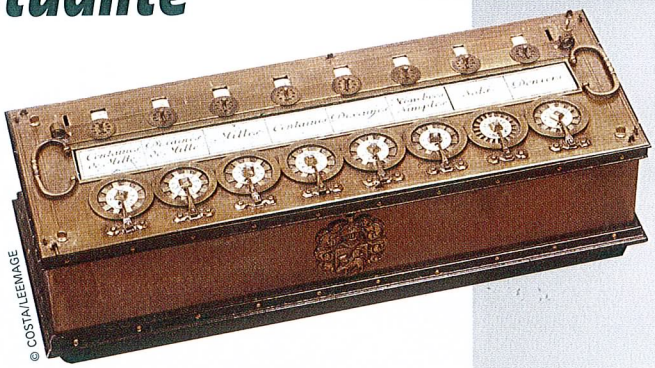
CHURCH INTRODUIT EN 1936 LE λ-CALCUL

Les tout premiers ordinateurs. Les choses se sont précisées après la Seconde Guerre mondiale, avec l'introduction des premiers ordinateurs commerciaux dans 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 Dijsktra des mécanismes de calculs menés en parallèle sur plusieurs processeurs et des problèmes de synchronisation associés, enfin des débuts de l'intelligence artificielle avec le calcul symbolique introduit par l'Américain John McCarthy dans le langage Lisp.

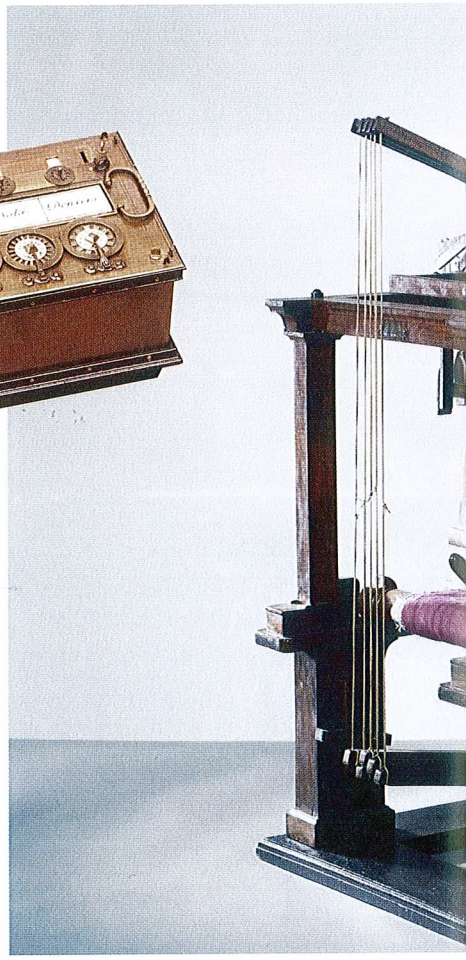
Mais la révolution informatique n'a vraiment commencé que dans les années 1970. En 1971, avec l'Intel 4004, apparut le microprocesseur, brique de base de tous les ordinateurs actuels. En 1975, l'Américain Gordon Moore, cofondateur d'Intel, formula sa célèbre loi qui allait conditionner tout le futur 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

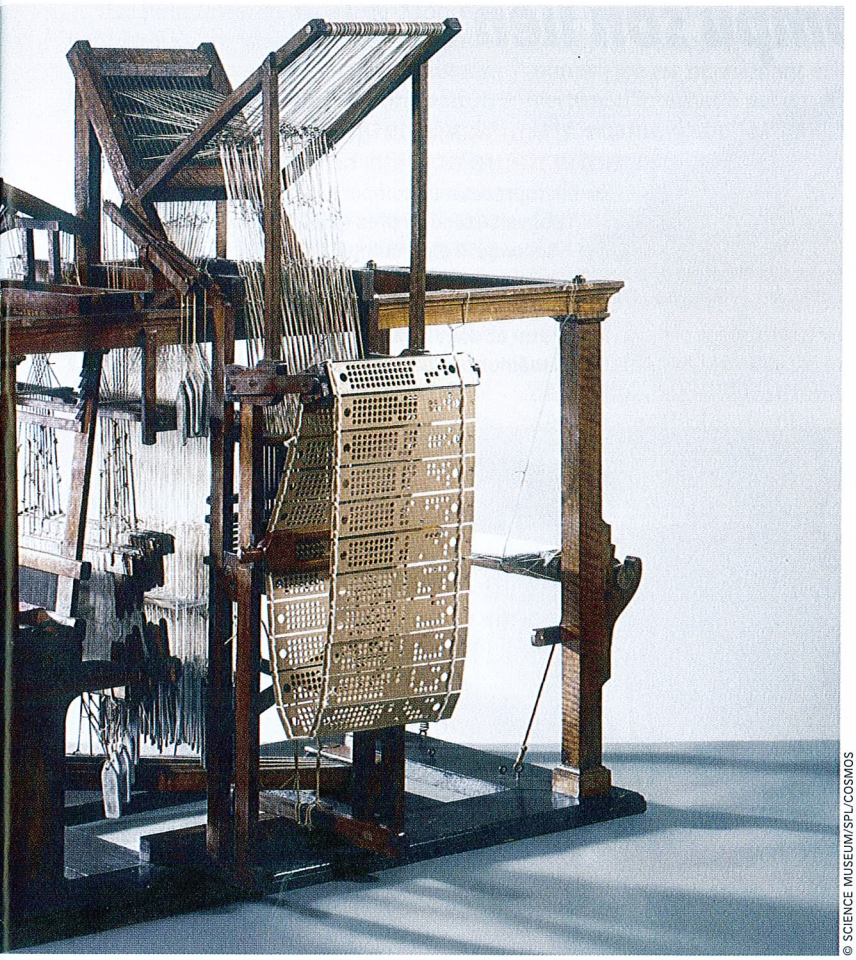


© COSTA/LEEMAGE

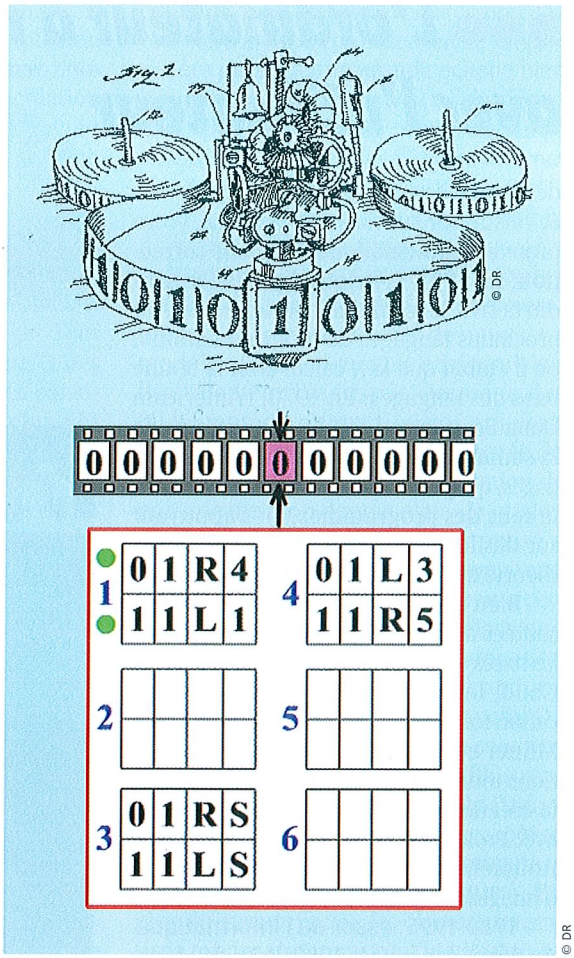
La Pascaline, première calculatrice mécanique. Elle fut inventée par Blaise Pascal en 1642, et ne pouvait effectuer que des additions et des soustractions.



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.



© SCIENCE MUSEUM/SPL/COSMOS



© DR

▲ **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.

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.

L'ORDINATEUR EST DEVENU UN INSTRUMENT SCIENTIFIQUE

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.

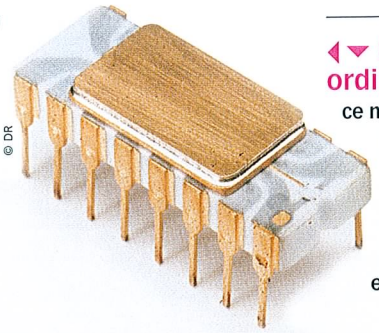
L'algorithmique a d'abord bénéficié du travail encyclopédique inestimable de l'Américain Donald Knuth, qui a systématisé l'étude de la complexité en temps et en mémoire de la plupart des algorithmes connus. Les problèmes les plus simples demandent un temps de calcul assez faible: un circuit met un temps 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 minimaliser 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.

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.

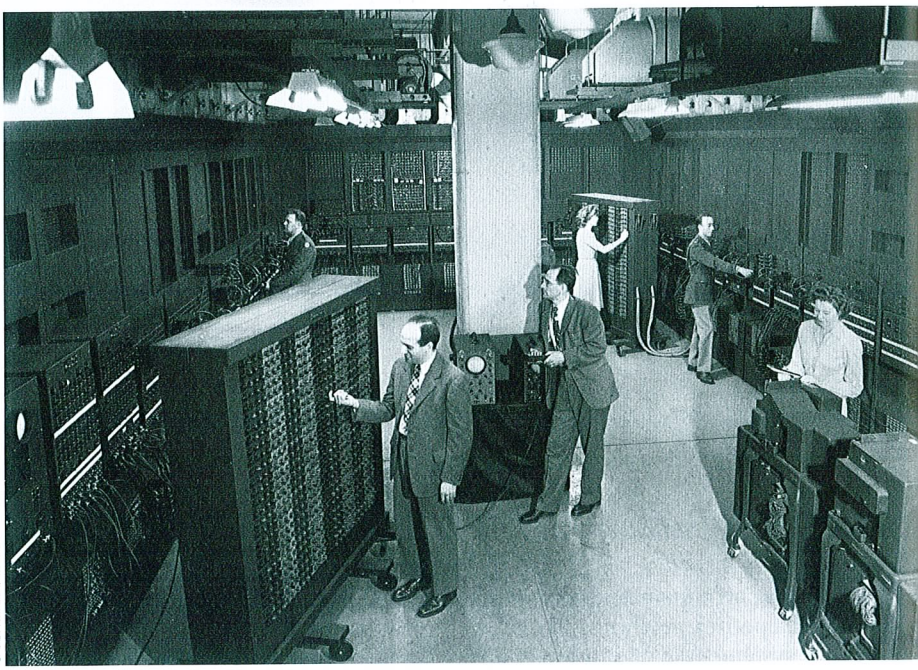
La théorie de la programmation s'est consacrée à l'augmentation de la sûreté des programmes, toujours menacés par les abominables *bugs*. Elle a travaillé à la définition de meilleurs langages et ●●●

L'ordinateur a rompu son lien avec l'utilisateur



© DR

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 m² et dont le programme n'était pas enregistré en mémoire : il devait être câblé manuellement.



© BETTMANN/CORBIS

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 λ-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.

LES APPPLICATIONS ONT EXPLOSÉ

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'entreprises avec le tableur 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'allaient 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, ●●●

© SCIENCE MUSEUM/SPL/COSMOS

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.

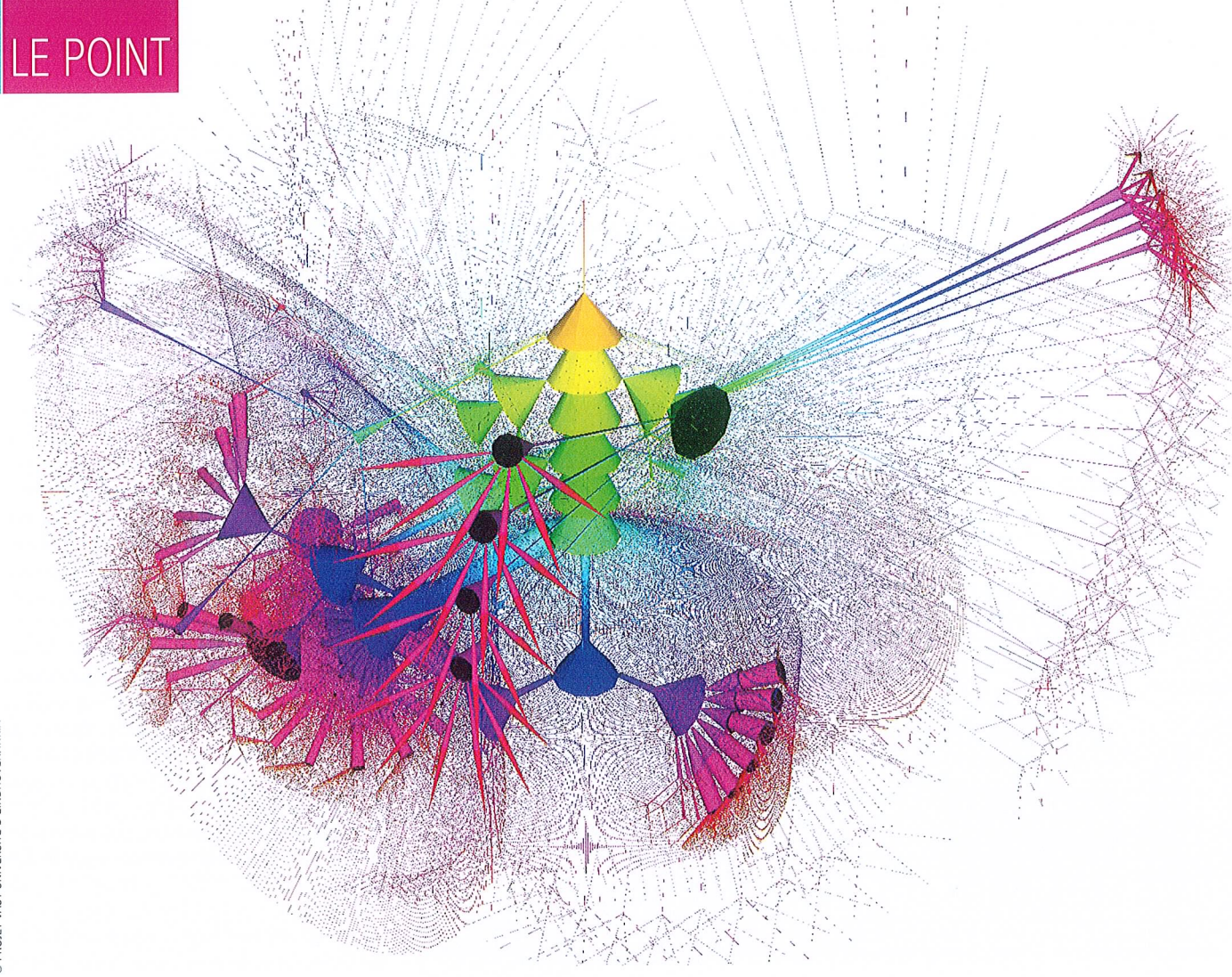
Tableau de bord de l'Airbus A320. Aujourd'hui, l'informatique se dissimule dans les objets ou appareils les plus divers, comme dans cet avion, dont les commandes sont reliées à un ordinateur de gestion de vol.

© AIRBUS

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.



© PROJET VASY-UNIVERSITÉ D'EINDHOVEN/INRIA

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).

APPRENDRE À VIVRE AVEC LES *BUGS*

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-

table est concurrencé par les «ordiphones» ou les tablettes qui se relieront 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 exemples parmi des milliers d'utilisations possibles. En bref, l'informatique ubiquitaire deviendra une réalité.

Vers les sciences numériques. Au XXe 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.

UNE EXTENSION À D'AUTRES SCIENCES

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? ●

SAVOIR +

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

◀ **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.**

© F. VIELCANET/URBA IMAGES SERVER

© DAVID BRABYN/CORBIS

Mathématiques et informatique

L'informatique a totalement transformé le paysage des mathématiques. Les liens à double sens entre les deux disciplines sont de plus en plus intenses et riches.

> PAR JEAN-PAUL DELAHAYE, UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE, LABORATOIRE D'INFORMATIQUE FONDAMENTALE DE LILLE, UMR-CNRS 8022, À VILLENEUVE-D'ASCQ

L'informatique connaît depuis plus d'un demi-siècle une évolution rapide qui a suscité une réflexion d'une extrême fécondité. Cela a conduit soit à utiliser des théories mathématiques existantes, qui se sont alors enrichies (l'arithmétique, la théorie des graphes, la théorie de l'information), soit à créer de nouveaux domaines mathématiques (par exemple, la théorie de la complexité liée à la théorie de la calculabilité). Une nouvelle sensibilité mathématique est née de l'usage des ordinateurs et des problèmes qu'ils posent à l'esprit théoricien. Si tout remonte à la décennie 1930 avec les travaux de l'Autrichien Kurt Gödel, du Britannique Alan Turing et de l'Américain Alonso Church sur la notion d'algorithme, cette sensibilité mathématique nouvelle a connu un essor considérable au cours des dernières décennies. À côté des mathématiques du continu, ces mathématiques du discret, du calcul et de l'information se sont épanouies, et l'on est très loin d'en avoir parcouru ne serait-ce que les allées principales.

LES ORDINATEURS BUTENT SUR DES LIMITES

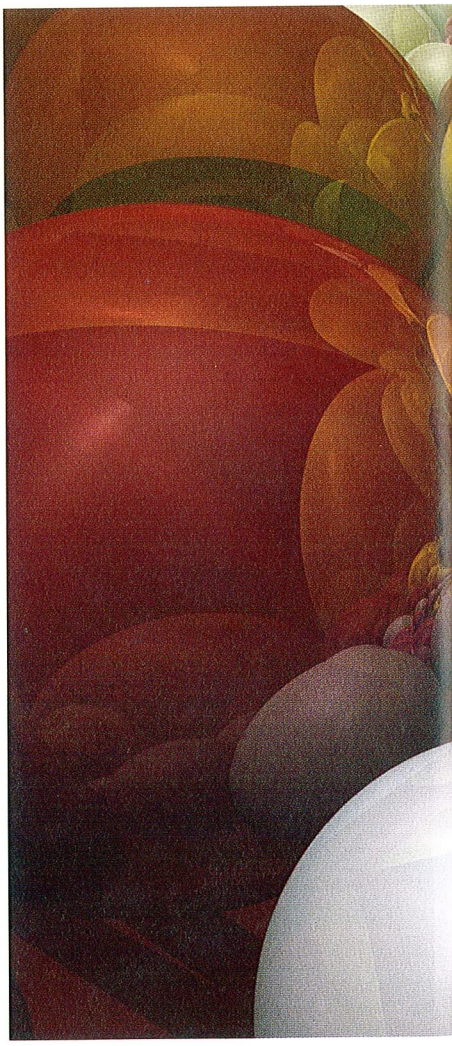
Organiser des milliards de calculs ?

Donner des ordres précis aux ordinateurs afin qu'ils fassent ce qu'on attend d'eux s'appelle « écrire des programmes ». Les mathématiciens conçoivent depuis toujours de telles méthodes de calcul, mais, en pratique, la règle implicite était qu'on utilise les algorithmes en opérant à la main quelques dizaines ou quelques centaines d'opérations élémentaires. Lorsque des machines capables de réaliser des millions, puis des milliards de calculs sans erreur furent disponibles, on découvrit que toute une science nouvelle, l'algorithmique, devait se développer et que cette science était d'essence mathématique. Commander des calculateurs automatiques n'est ni simple ni évident, et des méthodes au premier regard satisfaisantes se révèlent vaines dans le monde des ordinateurs. La résolution de systèmes d'équations linéaires par les formules du Suisse Gabriel Cramer (qui expriment les solutions comme des rapports de déterminants) ne permet pas de traiter des systèmes de taille 30, alors que d'autres méthodes iront bien au-delà. Les ordinateurs, quels qu'ils soient, butent sur des limites, et c'est à cette évidence qu'est confronté celui qui écrit des programmes et qui ne peut donc jamais raisonner en se disant que la puissance de la machine suppléera le manque d'analyse préalable.

La règle véritable est donc celle-ci : plus une machine est puissante, plus il y a de travail mathématique initial à produire pour la faire fonctionner correctement et en maîtriser la marche.

Un problème aussi simple que la multiplication des nombres entiers conduit au développement de méthodes complexes et délicates quand on doit manipuler des nombres de grandes tailles possédant plusieurs milliers, millions ou même milliards de chiffres. On a été étonné en s'apercevant que sur une question aussi élémentaire des progrès étaient possibles. Pourtant, dans la décennie 1970, des méthodes ont été mises au point, qui permettent de multiplier deux nombres de n chiffres en un temps proportionnel à $n \times ln(n) \times ln(ln(n))$. En pratique, cela est presque équivalent à un temps proportionnel à n et bien plus rapide que les méthodes classiques tirées des procédures pratiquées à la main depuis des millénaires qui, elles, demandent un

Une spirale de Doyle. Certaines structures géométriques complexes ne sont « visualisables » qu'à l'aide de programmes informatiques qui aident à les comprendre.



temps de travail proportionnel à n^2 ou n^3. Ce qu'il est possible de faire avec le bon algorithme de multiplication contemporain, sans lui et en ne comptant que sur la loi de Moore (qui prédit un doublement de la puissance des ordinateurs tous les 18 mois environ), il faudrait attendre 2050 ou plus pour y parvenir.

Des avancées surprenantes. Grâce à cette algorithmique sans cesse revisitée, complétée et perfectionnée, ainsi qu'aux progrès de l'électronique, on a pu progresser dans le calcul des constantes mathématiques et arriver à obtenir plusieurs milliards de décimales des principales d'entre elles (π, e, $\sqrt{2}$, le nombre d'or, etc.). Ces progrès se sont parfois accompagnés de surprises, comme la découverte par une équipe canadienne, en 1995, d'une nouvelle formule pour le calcul de π et d'une méthode exploitant cette formule, qui permet d'obtenir les chiffres binaires de π indépendamment les uns des autres.

Cet exploit théorique a été transformé en exploit pratique par Colin Percival qui, en septembre 2000, a obtenu une petite tranche de chiffres binaires autour du 10^{15e} chiffre binaire de π. Une telle performance semblait hors de portée pour des siècles. Ainsi, un bon algorithme rend possible ce qui, sans lui, ne l'est pas.

Comme autres conséquences de cette maîtrise algorithmique des calculs arithmétiques avec des nombres de grande taille, il faut citer la découverte de nombres premiers records : un nombre premier de deux millions de chiffres en 1999 et, récemment, un nombre de plus de 10 millions de chiffres. Là encore, la découverte des tests probabilistes de primalité (qui indiquent sans certitude absolue, mais avec un risque d'erreur infinitésimal, qu'un nombre entier est premier) fut une surprise théorique, par ailleurs d'une grande utilité en cryptographie.

UNE VISION NOUVELLE DES OBJETS ABSTRAITS

Au-delà des exploits informatiques parfois anecdotiques, c'est toute la science de l'organisation des calculs qui progresse et atteint la maturité, tissant des liens profonds avec les mathématiques et changeant parfois radicalement les points de vue. Des livres d'arithmétique d'un genre nouveau sont apparus dans lesquels les concepts introduits sont systématiquement associés à des outils algorithmiques permettant de les manipuler réellement avec un ordinateur. Les mathématiques sont nécessaires à la maîtrise des ordinateurs et contribuent en profondeur à leur compréhension, mais, en retour, les ordinateurs conduisent à pratiquer les mathématiques de manière différente et donnent une vision nouvelle des objets abstraits que la machine manipule mieux que l'esprit humain et avec une sûreté incomparable.

Des liens de plus en plus étroits. Ce n'est pas seulement l'arithmétique, mais une partie importante des mathématiques qui est concernée par les progrès de l'algorithmique. Le calcul formel (l'ordinateur calcule non plus seulement avec des nombres, mais aussi avec des symboles, des fonctions, des équations, etc.) s'est considérablement développé. Cependant, les rapports entre l'informatique et les mathématiques ne s'arrêtent pas là. Il faut citer notamment :

– l'analyse numérique, qui permet de résoudre des systèmes d'équations de grande taille ;

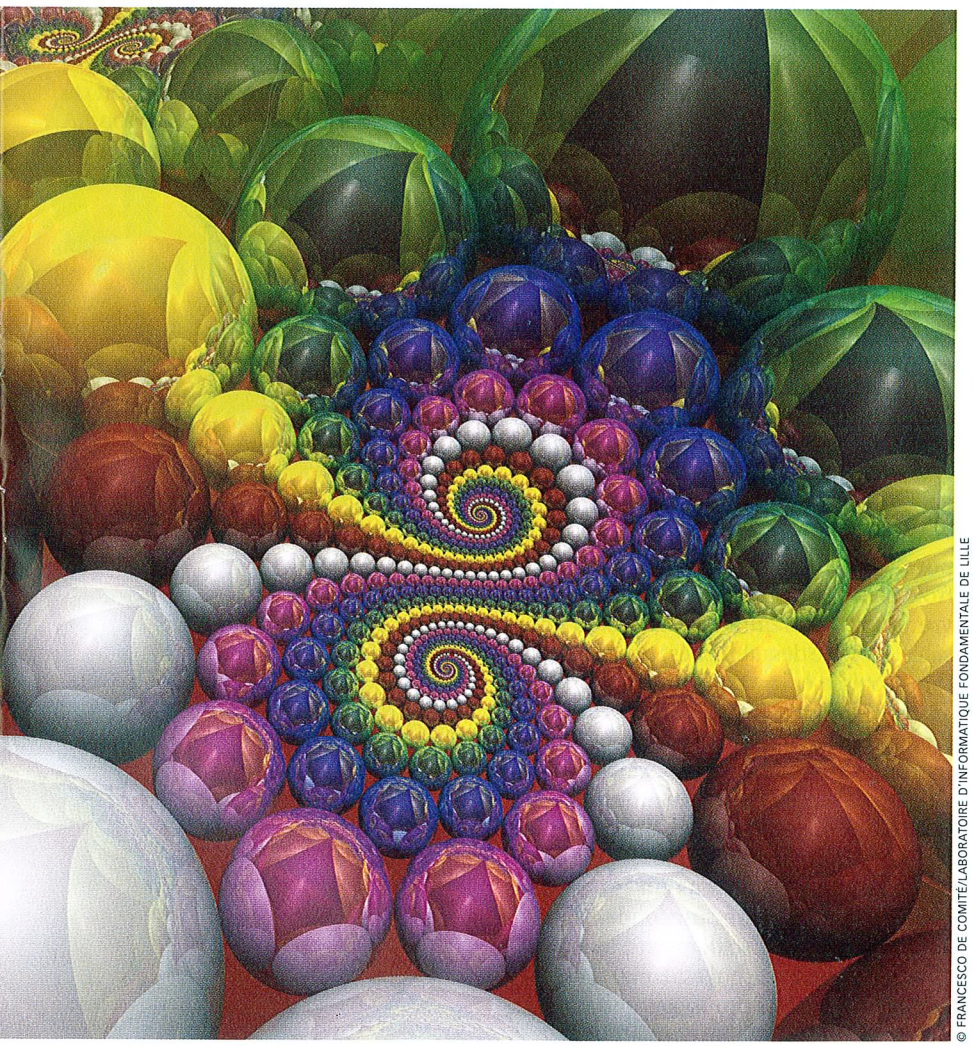
– la logique mathématique et ses extensions, utilisées en intelligence artificielle, dans le domaine des bases de données, pour la mise au point de méthodes de démonstration automatique et la certification de programmes ;

– la modélisation et la simulation qui, à l'aide de graphes et d'une grande variété de structures discrètes, permettent la reproduction dans l'ordinateur d'objets et de systèmes complexes provenant de la physique, de l'économie et de la biologie ;

– la sécurité informatique et la cryptographie, qui créent tout un ensemble de problèmes nouveaux, souvent rattachés à l'arithmétique ;

– l'étude des réseaux, qui suscite de nouvelles idées mathématiques.

L'avenir verra sans doute les liens entre l'informatique et les mathématiques se consolider et se resserrer encore. Il est devenu clair, aujourd'hui, que les deux sciences sont sœurs et que leur développement en synergie rend chaque jour plus profondes et intenses les relations qui les unissent. ●



© FRANCESCO DE COMITÉ/LABORATOIRE D'INFORMATIQUE FONDAMENTALE DE LILLE

SAVOIR +

- DEHORNOY Patrick. *Complexité et décidabilité.* Bâle : Birkhauser, 2000 (coll. Mathématiques et applications).
- DELAHAYE Jean-Paul. *Complexités : aux limites des mathématiques et de l'informatique.* Paris : Belin/Pour la science, 2006.

Un exemple de modélisation : le condensat de Bose-Einstein

La simulation numérique repose sur l'implémentation sur ordinateur de modèles théoriques. Elle sert à étudier les propriétés d'un système et à en prédire l'évolution.

> PAR IONUT DANAILA, MAÎTRE DE CONFÉRENCES, LABORATOIRE JACQUES-LOUIS-LIONS, UNIVERSITÉ PIERRE-ET-MARIE-CURIE

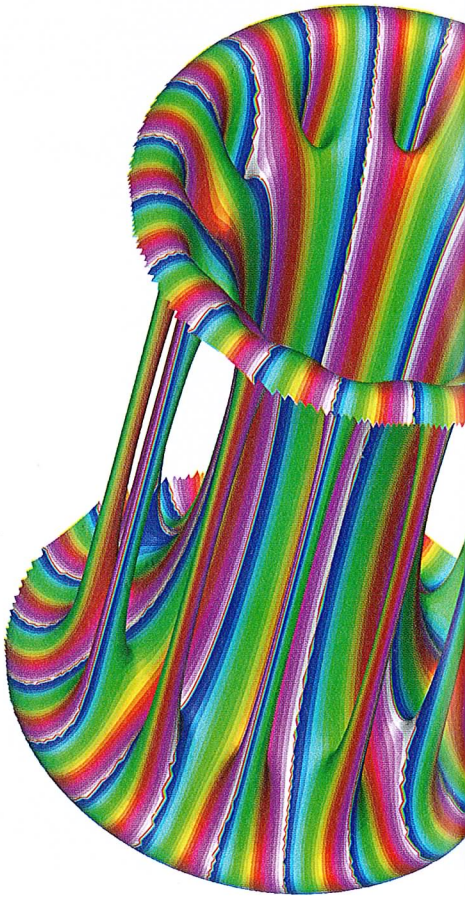
Imaginez qu'un matin, en effectuant le geste rituel de mélanger votre café avec une petite cuillère, le liquide refuse obstinément de tourner. Hautement improbable ? Pas tout à fait si le café, fluide de son état (comme tous les liquides et les gaz, il obéit aux lois fondamentales de la mécanique des fluides), est remplacé par un superfluide. Dans ce dernier, l'absence de toute forme de frottement entre molécules fait que la mise en rotation s'effectue suivant un scénario qui défie notre intuition (voir les deux photographies p. 18) : pour des rotations faibles, l'absence de forces de viscosité fait que le superfluide reste immobile ; pour des rotations importantes, des tourbillons quantifiés (ou vortex) apparaissent. Ces tourbillons sont des zones dépourvues d'atomes, autour desquelles le superfluide tourne localement. En augmentant progressivement la vitesse de rotation, de plus en plus de tourbillons apparaissent, s'organisant dans un réseau triangulaire qui porte le nom d'Alexei Abrikosov (Prix Nobel de physique en 2003).

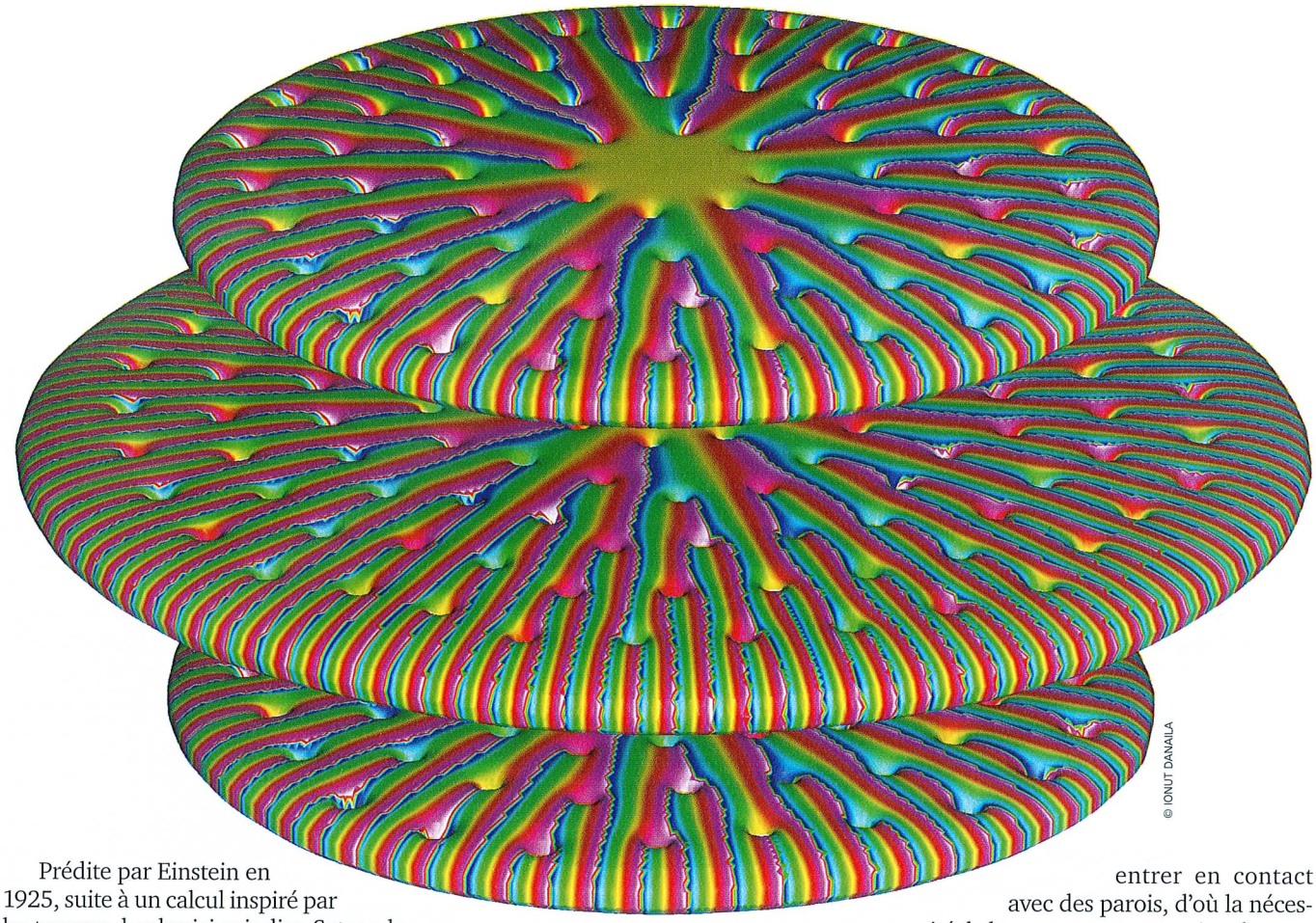
Comment obtenir un « café » superfluide ? L'idée est de réduire l'agitation moléculaire, et, par la même occasion, le frottement entre molécules, en diminuant drastiquement la température du fluide. Comme l'eau reste figée sous forme de glace solide en dessous de 0 °C, il va falloir la remplacer par un gaz pour atteindre des températures inférieures et garder le même état fluide de la matière. On sait que la température est une mesure de l'agitation atomique (ou moléculaire) d'un gaz et que le « zéro absolu » ou 0 kelvin (équivalent à - 273,15 °C) correspond au gel de toute agitation moléculaire. Ce qui est surprenant, c'est le fait qu'il est possible de discipliner le mouvement, habituellement désordonné, des atomes quand la température descend à quelques milliardièmes de degré au-dessus du zéro absolu, soit quelques nanokelvins (même l'espace interstellaire est maintenu par le rayonnement cosmologique à une température plus confortable d'environ 3 kelvins).

DANS LE FROID QUANTIQUE LES ATOMES VALSENT

Cette discipline s'impose naturellement par la « condensation » des atomes dans le même état quantique fondamental, caractérisé par la même longueur d'onde. Car la mécanique quantique ne regarde plus les atomes comme des petites billes rondes, mais comme des ondes. Et pour ces températures horriblement froides, les ondes individuelles se recouvrent pour n'en faire apparaître qu'une seule. La valse des atomes est parfaitement synchronisée et l'on parle ansi d'une onde géante ou d'un superatome pour décrire de manière imagée un nouvel état de la matière : le condensat de Bose-Einstein. Notons que les atomes forment un réseau fixe dans un solide ; ils se déplacent suivant des trajectoires chaotiques dans un gaz et glissent les uns sur les autres dans un liquide. Dans un condensat de Bose-Einstein, ils oscillent avec la même longueur d'onde, ce qui est l'une des rares manifestations des propriétés quantiques de la matière à l'échelle macroscopique.

▼ ▼ Configurations suggérées par calcul numérique. Réseau en rotation de trois condensats en forme de galette (en haut, à droite) et vortex géant (ci-dessous). Les couleurs mettent en évidence les discontinuités de phase introduites par les tourbillons quantifiés.





© IONUT DANAILA

Prédite par Einstein en 1925, suite à un calcul inspiré par les travaux du physicien indien Satyendra Nath Bose, la condensation d'un gaz ultra-froid a dû attendre soixante-dix ans pour être réalisée expérimentalement. Les Américains Carl Wieman et Eric Cornell et l'Allemand Wolfgang Ketterle ont obtenu pour la première fois, en 1995, un condensat de Bose-Einstein. Cette réalisation, attendue avec impatience par le monde scientifique, a été récompensée rapidement par le prix Nobel de physique en 2001. Quatre ans plus tôt, le même prix était attribué au Français Claude Cohen-Tannoudji et aux Américains Wiliam Phillips et Steven Chu pour la mise au point des techniques de refroidissement des atomes par laser, techniques qui ont ouvert la voie à la réalisation du condensat. Aujourd'hui, il existe dans le monde plus de trente laboratoires de recherche pouvant réaliser le condensat de Bose-Einstein, dont deux en France : le laboratoire Kastler-Brossel de l'École normale supérieure et l'Institut d'optique d'Orsay. Néanmoins, une telle expérience reste très coûteuse, avec un coût de l'équipement supérieur à 100 000 euros, ce qui devrait nous « refroidir » dans notre tentative, même imaginaire, de réaliser le « café » superfluide. Un autre système manifeste des propriétés superfluides : l'hélium liquide, quand il est refroidi « seulement » à une température inférieure à 2,2 kelvins (soit environ - 271 °C). La nature de la superfluidité, dans ce cas, est différente de celle décrite par Albert Einstein pour la condensation gazeuse.

© IONUT DANAILA

La simulation numérique comme outil d'exploration. La formidable cohérence du condensat de Bose-Einstein et sa haute contrôlabilité laissent présager des applications qui pourraient apporter une véritable révolution technologique. La construction d'un laser à atomes pour graver des circuits électroniques à l'échelle atomique, l'amélioration – d'un facteur inimaginable de 100 milliards – de la précision des instruments de mesure du temps (horloges utilisées, par exemple, dans les GPS) ou de la position (gyroscopes laser pour les avions et les sondes spatiales) font partie des applications envisageables. On parle aussi du condensat de Bose-Einstein comme étant une voie possible pour la réalisation d'un objet technologique qui tient actuellement de la science-fiction : l'ordinateur quantique.

LE CONDENSAT EST UN OBJET MINUSCULE

Il faut néanmoins savoir que le condensat produit en laboratoire est un objet minuscule d'environ 0,1 millimètre (ou 100 micromètres), entouré d'un nuage d'atomes non condensés. L'observation expérimentale est d'autant plus délicate que cette assemblée atomique est placée dans un piège magnétique. En effet, pour atteindre des températures proches du zéro absolu, les atomes ne doivent pas entrer en contact avec des parois, d'où la nécessité de les mettre en suspension dans un champ magnétique fort, généré par deux grosses bobines.

Pour prendre des photographies, le piège magnétique est coupé, puis un bref éclair lumineux est envoyé à travers l'assemblée atomique afin de mesurer son absorption et d'évaluer ainsi sa densité. Malgré la difficulté de la tâche, les photos expérimentales montrent remarquablement bien la structure du condensat, en particulier la présence de tourbillons quantiques (voir p. 18). Cependant, certains régimes, comme celui de la rotation rapide, sont difficiles à explorer expérimentalement. La simulation numérique peut alors apporter un complément d'information nécessaire à la compréhension des phénomènes et à la validation des théories physiques ; elle permet aussi une exploration détaillée de la structure tridimensionnelle (3D) du condensat et des lignes de tourbillon, ce qui est difficile – voire impossible – à visualiser dans les expériences de laboratoire.

Si l'on ajoute à l'argumentaire précédent le fait que le développement vertigineux des ordinateurs a rendu le coût de la simulation numérique très accessible, on peut se demander légitimement pourquoi l'expérimentation numérique ne remplace pas complètement l'expérience en laboratoire. Tout simplement parce que l'approche numérique est soumise à une série d'approximations ●●●

Un outil devenu incontournable

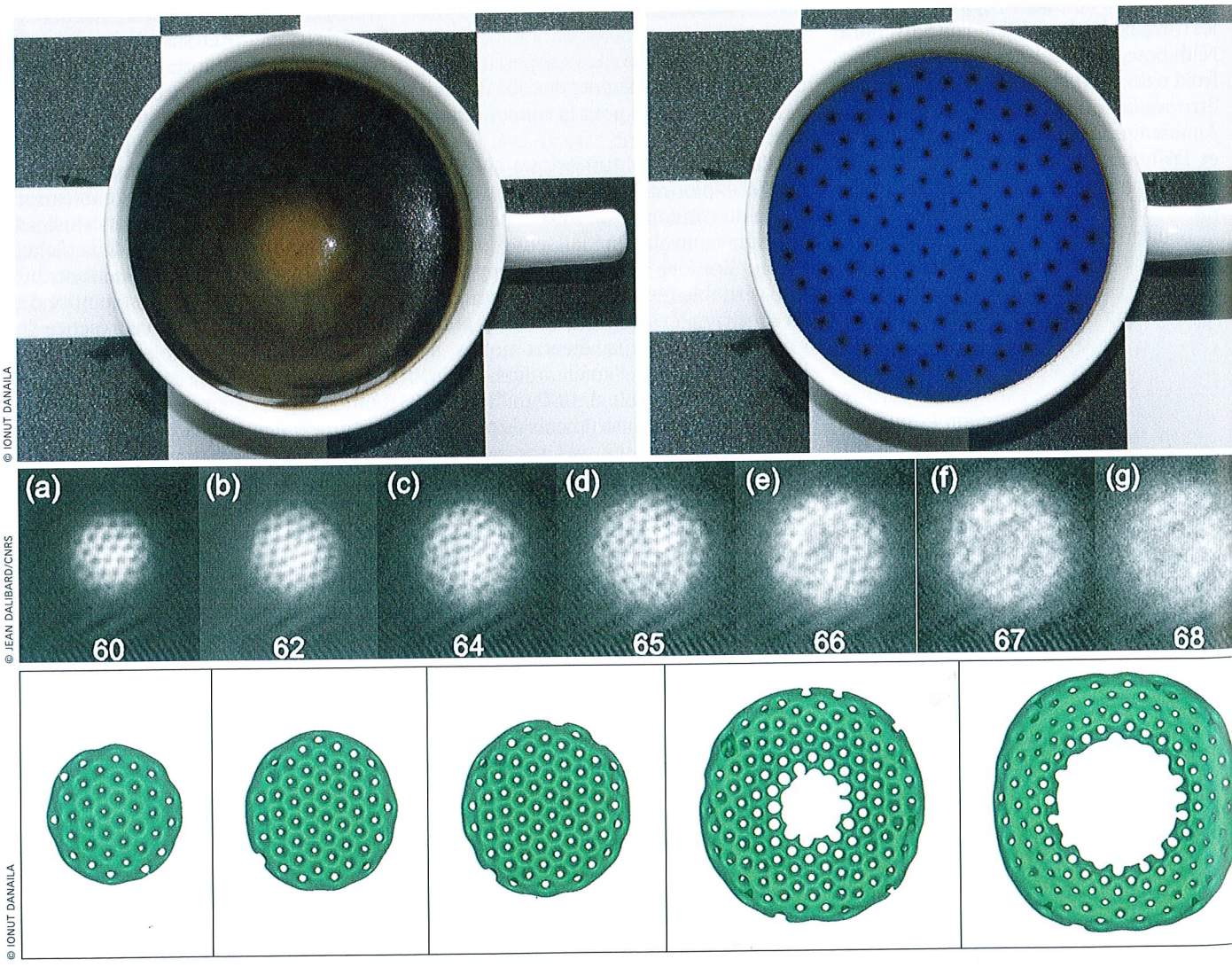
(modèles physiques et mathématiques), tandis que la vraie « physique » des phénomènes est observée en laboratoire. Parfois, même les plus puissants ordinateurs d'aujourd'hui ne peuvent pas offrir la capacité de calcul nécessaire à la résolution précise des modèles utilisés. Pour ces raisons, l'expérience en laboratoire et le calcul numérique restent des outils complémentaires dans l'exploration de la physique. En revanche, il existe des domaines où la simulation numérique est privilégiée par rapport à l'expérience et la remplace partiellement (conception des avions, automobiles) ou même totalement (essais nucléaires).

Au cœur du dispositif numérique. Le point de départ de l'aventure numérique est constitué par l'équation modèle qui décrit mathématiquement le phénomène physique. C'est habituellement le rôle des physiciens de mettre en équation les lois de la nature. Le modèle mathématique ainsi obtenu se présente généralement sous la forme d'une équation aux dérivées partielles, ou d'une équation intégrale. Les dérivées partielles sont généralement celles de la fonction inconnue par rapport aux variables d'espace (x, y, z) et de temps (t). La forme intégrale des équations est plus appropriée pour exprimer les lois de conservation de la physique (conservation de la masse, de l'énergie, etc.).

Les propriétés intrinsèques de l'équation modèle (existence, unicité des solutions, etc.) sont ensuite étudiées par les mathématiciens dans un cadre théorique abstrait. Les informations ainsi fournies sont utilisées en mathématiques appliquées ou en physique pour proposer des méthodes numériques adaptées à la résolution sur ordinateur de l'équation modèle. Les dernières étapes de la

Quand un superfluide remplace un fluide. Lorsqu'un liquide classique est mis en rotation (tasse de café, à gauche), les forces de frottement impriment une rotation en bloc. Le superfluide (simulé numériquement, à droite), se met en rotation par nucléation de tourbillons quantiques, qui apparaissent comme des petits trous désertés par les atomes du fluide (cercles noirs).

Du laboratoire à la simulation numérique. Au milieu : expérimentation en rotation rapide (plus la vitesse est importante, plus le contraste est faible ; voir figures e, f et g). En bas : simulation numérique en 3D de l'expérience en laboratoire, montrant l'évolution du réseau de tourbillons quantifiés vers une structure particulière, avec un trou au centre.

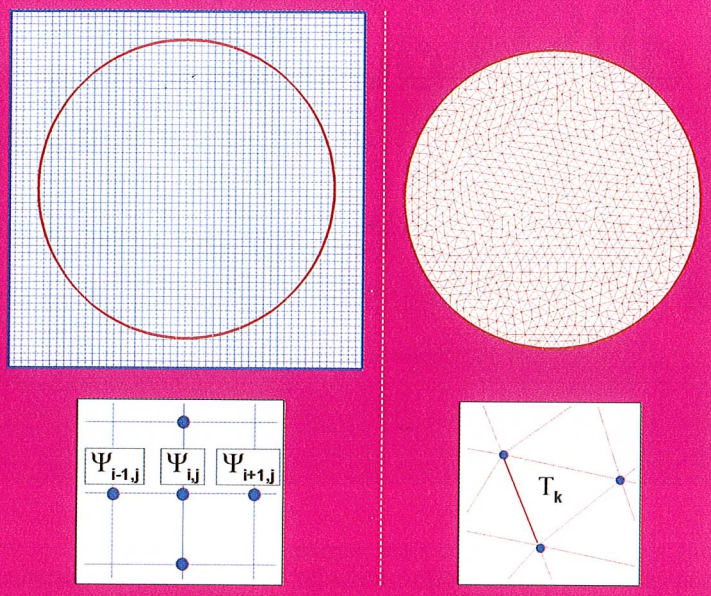


© IONUT DANAILA
© JEAN DALIBARD/CNRS
© IONUT DANAILA

FOCUS

Représentation d'un maillage en 2D

Il s'agit ici d'une représentation schématique d'une grille de calcul (ou maillage) en deux dimensions (*x, y*) de l'espace, pour deux méthodes représentatives du calcul numérique. La « méthode des différences finies » utilise un quadrillage rectangulaire du domaine de calcul (à gauche) qui englobe le domaine occupé par le condensat (cercle rouge). La variable inconnue Ψ sera calculée aux points discrets identifiés par deux indices (*i, j*), suivant les directions *x* et *y*. La forme différentielle de l'équation modèle est utilisée pour cette méthode.
La « méthode des éléments finis » offre plus de flexibilité dans la définition du domaine de calcul, qui peut avoir des frontières de forme complexe. Dans cet exemple, des éléments triangulaires sont utilisés pour couvrir exactement le domaine du condensat. La variable inconnue Ψ sera calculée aux sommets de chaque triangle T_k. La méthode utilise une forme intégrale de l'équation modèle.
Ces deux méthodes peuvent être généralisées en 3D : les cellules rectangulaires deviennent alors des briques parallélépipédiques et les triangles des tétraèdres.



© IONUT DANAILA

simulation numérique sont constituées par l'implémentation sur ordinateur (programmation de la méthode), le calcul proprement dit (qui peut varier de quelques secondes à plusieurs mois, en fonction de la complexité du problème) et l'exploitation des résultats.

Regardons ces étapes en détail dans le cadre de la simulation d'un condensat de Bose-Einstein. Nous nous rappelons que les atomes du condensat oscillaient avec la même longueur d'onde. Cette caractéristique permet de décrire le condensat en oubliant sa composition atomique (microscopique) et en utilisant une seule variable à l'échelle macroscopique. Cette variable est un nombre complexe, appelé « fonction d'onde » et noté par Ψ. L'équation qui décrit l'évolution de la fonction d'onde est due à deux physiciens, l'Américain Eugene Gross (1926-1991) et le Russe Lev Pitaevskii (actuellement professeur à l'université de Trente, en Italie). L'équation de Gross-Pitaevskii est une forme particulière de l'équation de Schrödinger, bien connue dans la mécanique quantique.

Une fois l'équation connue et son analyse mathématique réalisée, on passe à sa résolution numérique. Comme les ordinateurs fournissent des valeurs discrètes, on ne peut calculer qu'une approximation de la vraie solution. Seul un nombre fini de points du domaine du condensat sera donc calculé. Ces points sont distribués dans un « maillage » (voir **FOCUS**).

L'implémentation des méthodes et algorithmes passe par une phase de programmation qui génère des « codes de calcul ». Ces programmes fournissent les données numériques qui seront assemblées pour reconstituer l'image numérique du condensat. Comment savoir que les nombres fournis par l'ordinateur sont pertinents ? Tout d'abord, c'est l'analyse numérique – une branche des mathématiques appliquées – qui fournit la preuve que les méthodes et les algorithmes utilisés sont corrects. Ensuite, la validation ultime des résultats est donnée par la confrontation avec les mesures expérimentales.

SUGGÉRER DE NOUVELLES APPLICATIONS

Plus loin que l'expérience en laboratoire. La plus grande satisfaction d'un « numéricien » est de voir ses équations et ses lignes de code se métamorphoser en vrai banc expérimental. Une fois le système numérique validé, il peut être utilisé dans l'expérimentation numérique, c'est-à-dire pour calculer des situations non explorées en laboratoire, permettant ainsi d'aller plus loin pour approfondir la connaissance des phénomènes simulés. Les deux figures pp. 16-17 montrent des configurations simulées mais non réalisées, pour l'instant, en laboratoire. Imaginer et simuler de nouvelles configurations permet de stimuler les développements théoriques et de suggérer de nouvelles applications.

Si la simulation numérique est devenue un outil incontournable dans tous les domaines de la science, nous sommes néanmoins loin de l'image véhiculée dans les films de science-fiction, avec des programmes d'ordinateur intelligents qui renvoient immédiatement à l'écran une jolie image tridimensionnelle du problème simulé, comportant toutes les valeurs nécessaires à son analyse. Un important travail mathématique et informatique ainsi que beaucoup de savoir-faire sont encore nécessaires pour obtenir ce genre d'informations. Même si la recherche fondamentale dans ces domaines contribue, chaque jour, à se rapprocher de cet idéal, l'expertise du chercheur reste un facteur primordial dans la réussite d'une simulation numérique. ●

SAVOIR +

- DALIBARD Jean. « La condensation de Bose-Einstein en phase gazeuse », in *Images de la physique*, 2001. www.phys.ens.fr/~dalibard.
- DANAILA Ionut. *Tourbillons en fluides et superfluides : une exploration numérique*, mémoire de HDR (habilitation à diriger des recherches), 2008. www.ann.jussieu.fr/~danaila.
- « Le condensat de Bose-Einstein : la matière qui vient du froid », in *Science & vie*, hors série nº 228, septembre 2004.

« Un chemin initiatique vers l'abstraction »

Savoir se servir d'une souris ne suffit plus : il importe de se constituer un modèle mental correct de ce qu'est l'informatique et de comprendre la mutation sociétale qu'elle induit.

> INTERVIEW DE GILLES DOWEK, PROFESSEUR À L'ÉCOLE POLYTECHNIQUE, CHERCHEUR À L'INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE (INRIA), À SACLAY, PAR GUY BELZANE

TDC Beaucoup d'élèves ont envie de faire de l'informatique, alors que celle-ci n'est pas enseignée. Est-ce un paradoxe ?

Gilles Dowek. Beaucoup d'élèves sont des usagers de l'informatique : ils savent très bien se servir d'un logiciel, d'un blog, etc. Le but d'un enseignant en informatique n'est donc pas de leur apprendre ce qu'ils savent déjà, mais de leur faire comprendre que, derrière, il y a des concepts, des contenus. Et là, on est très surpris de la différence existant entre leur maîtrise des outils et leur compréhension. Cela tient beaucoup à une difficulté propre à l'informatique : les différents niveaux d'abstraction se mélangent. Par exemple, si vous montrez sur l'écran une icône sur laquelle vous devez cliquer pour lancer un navigateur, vous allez indifféremment dire que c'est une icône, ou que c'est un navigateur, en montrant le même objet. Si l'on n'explique pas à quel niveau d'abstraction on se place, il y a une possible confusion entre la signification du mot « navigateur » et celle du mot « icône ». Ce type de confusion, à des degrés divers, se retrouve souvent dans la tête des élèves… C'est pourquoi il est important qu'il y ait un enseignement de l'informatique. Ce n'est pas quelque chose que l'on apprend sur le tas, contrairement à ce que beaucoup pensent.

Pourquoi est-ce que l'informatique n'est pas enseignée ? Il y a deux raisons à cela. La première, c'est qu'elle est relativement récente ; elle n'existe que depuis les années 1930 et ne s'est véritablement développée qu'au cours des trois dernières décennies.

La deuxième raison, c'est qu'il y a un problème concernant la place des techniques dans l'enseignement général. On a souvent dévalorisé les savoirs techniques, notamment par rapport aux sciences. D'où les lycées spécialisés. Aujourd'hui, on s'aperçoit que des domaines scientifiques entiers sont totalement bouleversés par l'existence de l'informatique. Cette vision du monde – dans laquelle les techniques pouvaient être confinées dans un bocal que l'on appelait « lycée technique », parce qu'elles n'étaient pas en interaction avec le reste de la connaissance – est totalement obsolète. Malheureusement, elle a encore une certaine prévalence.

TDC L'informatique est-elle une technique ou une science ?

G. D. Les deux. Il s'agit d'un ensemble de connaissances extrêmement vaste. Or si l'on est prêt à accepter que la physique ou la chimie sont des domaines vastes et variés, on a un tout petit peu plus de mal à comprendre que c'est également le cas de l'informatique.

L'informatique est centrée autour de quatre concepts : le concept de machine, le concept d'algorithme, le concept d'information et le concept de langage. On voit que le concept d'algorithme, par exemple, est un concept partagé avec les mathématiques. Mais cela ne fait pas de l'informatique une branche des mathématiques, puisque seul un concept sur quatre est partagé avec cette discipline. La notion de machine est largement partagée avec les physiciens, car une machine, c'est avant tout un système qui obéit à des lois physiques. À côté de cela, il y a le concept de langage, comme les langages de programmation. Il est bien sûr partagé avec la linguistique, même si la notion de langage, en informatique, est un peu différente. La notion d'information, elle, est propre à l'informatique. On voit donc que celle-ci est très au contact d'autres sciences, auxquelles on ne peut cependant pas la réduire.

TDC Que serait un bon enseignement de l'informatique ?

G. D. Le B2i offre un enseignement qui n'a ni queue ni tête, pour deux raisons : la première, c'est qu'il réduit l'informatique à ses usages. Cela peut constituer l'une des phases de l'apprentissage, mais il faut aller au-delà, en particulier au lycée.

PROFIL

© DR

GILLES DOWEK
Enseignant à l'École polytechnique et chercheur à l'Institut national de recherche en informatique et en automatique (Inria), il est l'auteur des *Métamorphoses du calcul : une étonnante histoire de mathématiques* (Le Pommier, 2007), qui a obtenu le grand prix de philosophie de l'Académie française, et de *Ces préjugés qui nous encombrent* (Le Pommier, 2009).

© FRÉDÉRIC MAIGROT/RÉA

▲ **L'informatique à l'école primaire.** **Contrairement à une idée répandue, l'informatique n'est pas une science que l'on apprend sur le tas.**

La deuxième raison, c'est cette idée absurde qui consiste à faire enseigner l'informatique par une équipe pédagogique, et non par un enseignant en particulier. L'enseignement transdisciplinaire ne doit pas être une dissolution des disciplines. Au contraire, je crois que plus les élèves recevront un bon enseignement spécifique en informatique, plus les questions comme « Est-ce que l'âme est une machine ? » pourront être abordées, et même totalement renouvelées. Et si l'on peut comprendre que, dans une phase transitoire, il n'y avait pas d'enseignants, que l'on devait faire avec les moyens du bord, cela manifeste clairement un amateurisme qui n'est pas traditionnel dans l'Éducation nationale. Il faut donc aller au-delà et former aujourd'hui de véritables enseignants en informatique.

Un bon enseignement dans ce domaine est un enseignement qui accorde un équilibre aux quatre notions que j'ai citées plus haut. Veiller à cet équilibre est la règle d'or… qui n'est pas toujours comprise par les collègues des autres disciplines. Mais c'est grâce à cela que l'on suscite l'intérêt des élèves. Voilà pour la dimension épistémologique.

TDC Et en ce qui concerne la dimension pédagogique ?

G. D. J'aime bien décomposer le cursus d'un élève en deux étapes. La première – et là, le B2i n'était pas totalement faux –, c'est que l'on commence, en informatique, par apprendre à se servir de logiciels, non dans le but de devenir des experts du traitement de texte, mais afin de permettre le questionnement. Par exemple, concernant le navigateur : où sont stockées les informations que l'on trouve sur Wikipédia ? Comment ce réseau peut-il changer de géométrie ?… Cela permet de s'interfacer relativement bien avec les pratiques des jeunes. C'est ce que l'on devrait espérer de l'école primaire et des premières années de collège.

La deuxième étape est de faire comprendre comment ces logiciels sont fabriqués, de quoi ils sont formés, comment ils se décomposent. Tous les logiciels sont réductibles non pas à cent trois éléments chimiques, mais à quatre notions : l'affectation, la séquence, le test et la boucle. On atteint cet objectif en enseignant aux lycéens à écrire eux-mêmes de petits programmes et en leur montrant que tous les langages de programmation ont des invariants, des structures communes qui sont relativement simples et extrêmement puissantes.

On apprend un langage de programmation à la fois en écoutant un enseignant en parler, mais également en agissant par soi-même. Il y a ici la possibilité de réutiliser toute une démarche, celle de l'enseignement par projet, dans un cadre traditionnel.

Un certain nombre d'élèves sont rebutés par les sciences, précisément parce qu'elles sont enseignées d'une manière trop magistrale. La synthèse, la hauteur de vue que permettent ces cours sont nécessaires, mais comment y parvenir sans rebuter ces élèves ? Ici, en particulier pour les autres sciences, l'informatique peut jouer un rôle important. Les élèves commencent par une activité pratique – écrire de petits programmes – , puis ils se rendent compte que s'ils ne font pas preuve d'un peu de rigueur leurs programmes ne fonctionnent pas. On peut donc utiliser l'informatique comme un chemin initiatique vers l'abstraction.

TDC L'informatique est-elle la plus concrète ou la plus abstraite des sciences ?

G. D. Les deux à la fois. Les notions de machine, de langage sont des concepts extrêmement abstraits. À l'université, on aborde l'informatique comme une science, avec de nombreuses ramifications. Mais, au lycée, la dimension pratique doit être bien plus importante.

À côté de cela se pose la question de la place des interrogations du type : qu'est-ce qu'une licence ? Qu'est-ce que le téléchargement illégal ? Pourquoi la notion de propriété est-elle totalement transformée par l'arrivée de biens non rivaux, c'est-à-dire de biens que l'on peut dupliquer à l'infini et sans coût ? Réduire l'informatique à ses aspects sociétaux serait disserter sur un sujet que l'on ne connaît pas. Éviter ces aspects, ce serait aussi donner une image amputée de ce qu'est l'informatique. Il faut trouver un équilibre. J'ai tendance à penser que les cours sont faits pour être consacrés aux notions fondamentales, et que les questions plus sociétales doivent être abordées à travers les TPE, par exemple. ●

Au service des citoyens

Afin de protéger les libertés collectives et individuelles, il est indispensable d'intégrer des critères juridiques et éthiques dès la conception des systèmes informatiques.

> PAR DANIEL LE MÉTAYER, DIRECTEUR DE RECHERCHE À L'INRIA, RESPONSABLE DE L'ACTION EXPLORATOIRE LICIT (DROIT ET TECHNOLOGIES DE L'INFORMATION)

© PIERRE PHILIPPON

« L'informatique doit être au service de chaque citoyen. Son développement doit s'opérer dans le cadre de la coopération internationale. Elle ne doit porter atteinte ni à l'identité humaine, ni aux droits de l'homme, ni à la vie privée, ni aux libertés individuelles ou publiques » : en énonçant ce principe dès le premier article de la loi « Informatique et libertés », le législateur français faisait, en 1978, figure de pionnier. Cette loi fut votée à la suite de l'émotion provoquée par la révélation du projet de création de Safari (système automatisé pour les fichiers administratifs et le répertoire des individus), qui aurait permis d'interconnecter des données personnelles réparties dans des centaines de fichiers administratifs. L'informatique de l'époque, largement centralisée, prenait la forme d'ordinateurs d'une taille et d'un coût tels qu'ils restaient réservés aux administrations, aux universités et aux entreprises.

Tout autre est la situation actuelle avec l'avènement de l'« informatique ubiquitaire » : les moyens de calcul et de communication sont de plus en plus miniaturisés et omniprésents dans notre environnement, au point que certains ont qualifié l'ordinateur du futur de *disappearing computer*, ou « calculateur évanescent », fragmenté en multiples dispositifs invisibles dont nous dépendrons de plus en plus dans les moindres de nos activités quotidiennes. On peut penser notamment aux logiciels intégrés dans les appareils les plus variés (machines à laver, téléviseurs, automobiles, avions, etc.), aux capteurs, aux puces RFID (*radio frequency identification*) déjà utilisées pour contrôler, par exemple, l'accès à des bâtiments, verrouiller des voitures ou identifier des animaux (parfois même des humains), et qui seront peut-être présentes un jour sur tous les objets, reliées à internet pour créer l'« internet des objets ».

Aujourd'hui déjà, le formidable déploiement d'internet et notre dépendance croissante à ses services posent des questions de société majeures. Le principe énoncé en introduction n'a pas pris une ride et la loi « Informatique et libertés » demeure un instrument précieux pour encadrer les nouvelles technologies dans le souci du bien collectif. Il faut reconnaître cependant que les développements spectaculaires de ces technologies, s'ils rendent d'autant plus nécessaire l'affirmation de principes clairs, remettent aussi souvent en cause l'applicabilité même de la loi de 1978.

Deux sources de normativité parfois antagonistes. La protection de la vie privée est l'un des sujets de tension les plus perceptibles entre l'informatique et le droit. Il ne se passe plus une semaine, désormais, sans que la presse ne se fasse l'écho d'atteintes flagrantes à la vie privée liées aux nouveaux usages de l'informatique (réseaux sociaux sur internet, moteurs de recherche, cartes de transport, passeports biométriques, vidéosurveillance, etc.). Certains en viennent même à regretter que les technologies de l'information, malgré l'extraordinaire variété des services et des possibilités qu'elles offrent aujourd'hui, soient le plus souvent montrées du doigt, voire diabolisées par les médias.

DES QUESTIONS DE SOCIÉTÉ MAJEURES

De fait, le rapport entre le droit et les technologies est souvent présenté de manière binaire : doit-on adapter le droit, revoir nos principes pour les soumettre à la nouvelle donne technique ou plutôt renoncer à certaines facilités, brider les possibles, pour faire entrer les nouveaux services dans le cadre contraignant du droit existant ? Pour prendre quelques exemples concrets : doit-on interdire le déploiement d'un système de paiement de l'usage des routes qui permettrait différents types d'économies pour la société (diminution de la pollution) et pour les usagers (tarifs plus justes) parce qu'il rendrait possible un traçage complet des automobilistes ? Doit-on limiter l'usage de cartes de fidélité qui permettent d'obtenir quelques rabais au prix d'une divulgation de profils de consommation très révélateurs des modes de vie ? Doit-on protéger contre leur gré les adolescents qui mettent en ligne des informations sensibles qu'ils ne pourront plus effacer et qui risquent de réduire leurs chances de trouver un emploi dans l'avenir ? Doit-on filtrer internet pour empêcher l'accès à des sites illégaux ? Mais

◀▲ Le *boom* des puces… et le risque technologique. Face à la montée en puissance des nouvelles technologies (puces RFID, cartes et passeports biométriques, etc.), des voix de plus en plus nombreuses s'élèvent pour défendre le droit à la confidentialité et obtenir une meilleure sécurité informatique.

réduire ainsi le rapport entre le droit et les technologies à une simple opposition est à la fois stérile et fallacieux : même s'il ne s'agit pas ici de nier les tensions qui peuvent exister entre la volonté d'offrir des services et des produits innovants et celle de respecter le bien commun et les droits des individus, il convient d'examiner une troisième voie, plus constructive, celle de la prise en compte des dimensions légales et éthiques dès la phase de conception des systèmes informatiques.

Réconcilier la technique et le droit. Un projet en cours d'étude aux Pays-Bas vise à équiper les véhicules de boîtiers incorporant des fonctionnalités de localisation géographique (GPS) et de communication (GSM) de sorte que le tarif d'imposition actuel, uniforme, puisse être remplacé par une taxe variable dépendant de facteurs comme le nombre de kilomètres parcourus, les heures de circulation, les routes empruntées, etc. Un tel système contribuerait à préserver l'environnement en décourageant la circulation aux heures de pointe ou lors de pics de pollution. Il apporterait également des bénéfices aux usagers, notamment une circulation plus fluide et des tarifs plus justes. Cependant, la solution proposée initialement par les pouvoirs publics se révélait très intrusive, puisqu'elle supposait l'envoi régulier (par exemple, chaque minute) à un serveur central de la position de chaque véhicule : tous les parcours des automobilistes se verraient ainsi enregistrés pour une période relativement longue (pour des besoins de facturation), et cela à l'échelle d'un pays entier !

DES ATTEINTES FLAGRANTES À LA VIE PRIVÉE

Mais cette atteinte au droit de circuler de manière anonyme n'est en aucune façon une fatalité. Il résulte simplement de choix d'architecture du système informatique, choix qui peuvent très bien être remis en cause. Des chercheurs néerlandais ont ainsi proposé une solution plus satisfaisante : au lieu d'être déporté sur le serveur central, le calcul du tarif est réalisé par le calculateur de bord de l'automobile et envoyé périodiquement (par exemple, une fois par trimestre) au serveur.

Une telle solution permet donc d'obtenir le même service sans mettre en péril la vie privée des automobilistes, puisque leurs parcours ne sont pas envoyés au serveur. Ainsi décrite, elle ne serait pas complètement satisfaisante pour l'opérateur routier, cependant, puisque des automobilistes indélicats pourraient être tentés de trafiquer leurs équipements de bord afin de réduire le montant de leur taxe. Pour limiter ce risque, la proposition alternative prévoit également l'envoi régulier au serveur central d'un condensé (ou *hash*, en anglais) du parcours effectué. Le condensé possède deux propriétés importantes : il n'est pas possible de retrouver la valeur initiale à partir du condensé et deux valeurs différentes produisent forcément – en réalité à une très forte probabilité – deux condensés différents. De cette manière, l'opérateur peut effectuer des contrôles inopinés sur la route, saisir une portion de trajet sur l'équipement du véhicule, calculer son condensé et vérifier qu'il correspond bien à celui qui a été envoyé au serveur central par le véhicule. Cette solution permet donc de concilier les besoins de protection de la vie privée avec les exigences de sécurité de l'opérateur.

Vers une meilleure conception éthique. Le système de péage routier des Pays-Bas illustre le fait que la protection de la vie privée est un impératif qui doit être pris en compte dès la phase de conception d'un système informatique. Cette démarche a déjà été appliquée dans d'autres contextes, comme la mise en œuvre de dossiers médicaux personnalisés, de services reposant sur la géolocalisation ou encore pour la publicité ciblée. ●●●

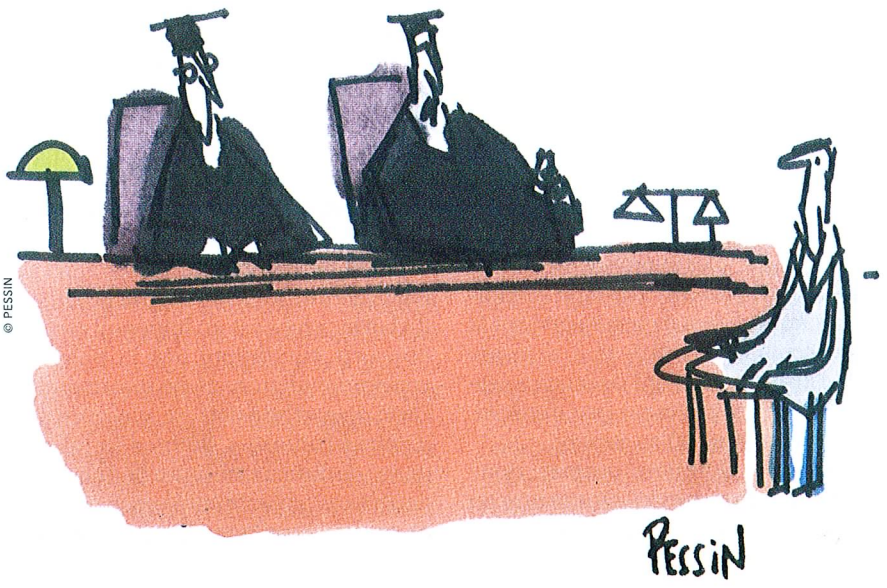
Protéger la vie privée est un impératif

De manière plus générale, ces exemples montrent ce qu'ont déjà affirmé un certain nombre d'auteurs, à savoir que l'architecture d'un système informatique déployé à grande échelle reflète des choix de société. En d'autres termes, l'informatique, en se diffusant aussi largement, en pénétrant ainsi tous les aspects de notre vie quotidienne, est devenue une source majeure de normes.

Le juriste américain Lawrence Lessig, qui fut l'un des premiers à en prendre conscience et à s'en inquiéter, a proposé une classification des modes de régulation en quatre catégories : la loi, les normes sociales, le marché et l'architecture informatique. D'après Lessig, l'architecture est la modalité dominante sur internet – pour simplifier, le code informatique devient la loi – et cet état de fait est préjudiciable pour diverses raisons, notamment le manque de transparence et les effets indésirables de la technique. De fait, la question qu'il convient de poser à ce stade est celle des critères de validité d'une norme. Le juriste Yves Poullet, pionnier en matière du droit de l'informatique, a notamment proposé trois critères de validité principaux : *la légitimité,* qui porte sur la source de la norme et mesure sa représentativité ainsi que la transparence de ses procédures ; *la conformité,* qui porte sur le contenu de la norme et mesure son respect des valeurs de la société, notamment sur le plan éthique ; *l'effectivité,* qui porte sur la mise en pratique de la norme et le contrôle de son respect. Il paraît clair qu'à l'aune de ces trois critères, la validité des normes imposées par les techniques soulève de nombreuses questions. Le code informatique est produit par des ingénieurs, des programmeurs sans légitimité particulière pour édifier des normes ; ces codes sont souvent gardés secrets pour des raisons commerciales (les algorithmes de classement de pages des moteurs de recherche fournissent l'un des exemples les plus édifiants à cet égard) . L'élaboration d'une solution informatique répond généralement à un cahier des charges dicté par les besoins du marché, sans considération particulière pour les questions d'éthique. L'effectivité elle-même est problématique : à l'inverse de la norme juridique dont on regrette parfois l'ineffectivité, la norme technique peut se voir reprocher sa sureffectivité dans le sens où elle peut rendre simplement impossibles certains comportements (alors que la norme juridique agit plutôt en sanctionnant les comportements interdits).

RAPPROCHER LA RÉALITÉ DE LA LOI DE 1978

La norme technique peut ainsi conduire à une réduction des libertés et à ce que certains ont appelé un « paternalisme technologique ». En effet, comme le note Antoinette Rouvroy, chercheuse en droit, « il n'est pas neutre de rendre techniquement impossibles certaines actions ou certains comportements illégaux. N'est-il pas nécessaire que les citoyens puissent, dans certains cas, désobéir au droit ? » On l'aura compris, les relations entre l'informatique, le droit et l'éthique ne sont pas simples et une meilleure prise en compte des dimensions juridiques et éthiques dans les projets informatiques est un véritable défi, qui pose de délicates questions à tous les niveaux, des fondements éthiques et juridiques aux nouvelles solutions techniques à inventer, en passant par les problèmes d'organisation interdisciplinaire et de conduite de projets. Il s'agit là d'un nouveau champ de recherche en soi, que certains ont appelé *value sensitive design,* et que l'on pourrait traduire par « conception éthique ». Il serait exagéré de dire que cet état d'esprit et cette démarche sont largement répandus, mais ils commencent à se traduire dans les faits, comme on l'a décrit plus haut, pour la protection de la vie privée. On peut penser que cette prise de conscience s'amplifiera au cours des années à venir et permettra de rapprocher la réalité du principe de la loi de 1978, selon lequel « l'informatique doit être au service de chaque citoyen ». ●



© PESSIN

SAVOIR +

- LE MÉTAYER Daniel. *Les Technologies de l'information au service des droits : opportunités, défis, limites.* Bruxelles : Émile Bruylant, 2010.
- LE MÉTAYER Daniel, ROUVROY Antoinette. « STIC et droit, défis, conflits et complémentarités », sur le site Interstices, novembre 2008, http://interstices.info/jcms/c_34521/stic-et-droit-defis-conflits-et-complementarites
- POULLET Yves. « Technologies de l'information et de la communication et "co-régulation" : une nouvelle approche ? », in *Liber Amicorum Michel Coipel.* Bruxelles : Kluwer, 2004.

Zoom sur les sciences numériques

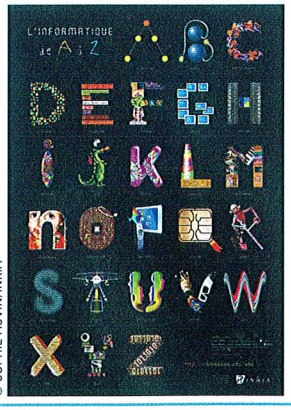
Algorithme, bogue, cryptographie, données, équation… Découvrez, sous la forme d'un abécédaire, un panorama du vocabulaire de l'informatique !

> PAR CHRISTINE LEININGER, CONCEPTRICE-RÉALISATRICE DANS L'ÉQUIPE MULTIMÉDIA DE L'INRIA, RÉDACTRICE EN CHEF DU SITE INTERSTICES

Le poster présenté ici récapitule une série de vingt-six affiches – déclinées aussi sous forme de cartes postales – qui ont été réalisées à l'initiative de Christine Genest dans le cadre d'un partenariat entre l'Inria et le Centre de vulgarisation de la connaissance (université Paris-Sud-XI), soutenu par le CNRS. Vous pourrez retrouver sur le site *Interstices* (voir **SAVOIR +**) l'ensemble des termes, des illustrations et des textes qui y sont associés.

Chaque lettre illustrée de ce poster renvoie à un mot choisi par les chercheurs pour évoquer leurs domaines de recherche : de l'**algorithme**, assurément la notion la plus fondamentale de la science informatique, jusqu'au **zéro**, qui nous rappelle que le codage binaire est indissociable du fonctionnement physique de nos ordinateurs. Plus qu'un inventaire à la Prévert, cet ensemble de mots constitue un témoignage de l'étendue du champ disciplinaire de l'informatique et des sciences du **numérique**.

POSTER



© SOPHIE AUVIN/INRIA

Qu'est-ce qu'un algorithme ? C'est tout simplement une méthode qui décrit une suite de tâches élémentaires et la façon de les enchaîner pour résoudre un problème. Il existe des algorithmes mathématiques, comme celui d'Euclide, mais aussi des méthodes pratiques et universelles pour rechercher un mot dans un dictionnaire, pour trouver la sortie d'un labyrinthe, etc. Pourquoi cette notion est-elle si importante ? Parce que, lorsqu'une bonne méthode – un algorithme approprié – est utilisée, on parvient à résoudre un problème bien plus rapidement. Ainsi, lorsqu'on se base uniquement sur la définition des nombres premiers pour en établir une liste, cela se révèle largement moins efficace que lorsqu'on utilise le crible d'Ératosthène, qui peut à son tour être amélioré. Le **temps** est mis en évidence comme un facteur primordial.

Celui-ci intervient également de façon critique dans les applications qui doivent évoluer en temps réel, que ce soit dans les domaines médical ou industriel ou dans ceux du **multimédia** ou des **jeux** vidéo. Des applications fort appréciées des **utilisateurs**, même si, bien souvent, ils ne comprennent ni l'ampleur de leurs enjeux – résumés d'une façon un peu lapidaire et provocatrice par la formule : « Le monde devient numérique » – ni la complexité de leur fonctionnement.

Des programmes à exécuter. Toutes les applications sont, en fait, des logiciels, des programmes qui mettent en œuvre les algorithmes définis précédemment. Ces programmes sont écrits dans des **langages** informatiques, afin de pouvoir être exécutés par les **ordinateurs**. Il existe une grande diversité de langages, qui répondent au besoin de programmer différents types de tâches.

TOUTES LES APPLICATIONS SONT DES LOGICIELS

Quoi qu'il en soit, le programme est ensuite traduit automatiquement par un autre programme appelé « compilateur » dans le langage machine, directement exécutable par les circuits électroniques de l'ordinateur. Comme les **données** sur lesquelles il travaille, le programme en langage machine est une suite de 0 et de 1, qui ne sont plus lisibles par les humains mais sont interprétés par les systèmes informatiques. Les différents composants électroniques entrent alors en action : processeurs, mémoires, sans oublier l'**horloge**, qui agit comme un chef d'orchestre.

Les programmes doivent être soigneusement vérifiés, voire prouvés formellement, pour éviter les **bogues** – ou *bugs*, en anglais –, les erreurs qui provoquent parfois des catastrophes. Songez, par exemple, au vol inaugural d'Ariane 5, lorsque la fusée a explosé et détruit pour 500 millions de dollars de satellites. Le bogue était dû à une partie de programme créée pour Ariane 4 et réutilisée telle quelle, ce qui a donné •••

un résultat aberrant conduisant à un ordre de braquage maximal.

La **qualité** des logiciels ainsi que celle des interfaces par lesquelles les utilisateurs interagissent avec les programmes sont indispensables pour garantir le bon fonctionnement des systèmes informatiques.

Ces systèmes sont présents dans les ordinateurs, mais pas seulement. Ils pilotent toutes sortes de dispositifs, comme les **robots**. Mais ne vous fiez pas à l'image du robot dont l'aspect se rapproche de celui de l'être humain ; de tels robots restent rares. Dans l'industrie, les bras robotiques sont bien plus nombreux. Ils servent notamment à assembler avec précision des pièces mécaniques ou à effectuer des interventions dans un environnement hostile. Les robots sont souvent dotés d'un système de vision qui s'apparente d'assez loin à nos **yeux**. Une ou plusieurs caméras enregistrent des images de l'environnement, qui sont analysées par un système de reconnaissance de **formes**. Les objets à saisir sont ainsi repérés, les obstacles détectés et évités.

Les « autoroutes » de l'information. En informatique, la sécurité, c'est aussi la protection des données grâce à la **cryptographie**. Cela s'avère particulièrement crucial lorsque des informations confidentielles doivent transiter sur un réseau. Schématiquement, le message est crypté par une méthode de chiffrement, à l'aide d'une clé. En général, c'est une autre clé, liée à la première par une formule mathématique, qui sert à récupérer les informations sous une forme lisible. La formule mathématique employée est telle qu'il est très facile à une personne de générer sa clé publique, qu'elle diffuse pour permettre de crypter les messages qui lui sont destinés. En revanche, il est très difficile à un tiers d'en déduire la clé privée du destinataire, celle qui lui sert à décrypter les messages.

Lorsque des fichiers sont envoyés par internet, le temps est là encore déterminant. À l'ère du multimédia, qu'il s'agisse d'images, de sons, de vidéos, la taille des fichiers a énormément augmenté, passant, en peu d'années, de quelques **kilobits** à plusieurs gigabits. Heureusement, les protocoles de transmission permettent d'acheminer les fichiers non pas entiers, mais par petits blocs ou « paquets », qui circulent indépendamment les uns des autres et sont remis dans le bon ordre une fois parvenus à destination. Ainsi, lorsque vous cliquez sur un lien, la page Web que vous avez demandée s'affiche rapidement sur votre écran. **Web** et **internet** sont liés, mais attention à ne pas les confondre. Le Web est l'une des applications construites sur internet, sans doute celle qui a connu le plus grand succès. À tel point que, aujourd'hui, c'est un véritable casse-tête pour naviguer dans la jungle des pages Web ! Grâce à la structuration permise par le langage **XML** (*extensible markup language*, « langage à balises étendu »), il devient plus facile de s'y retrouver.

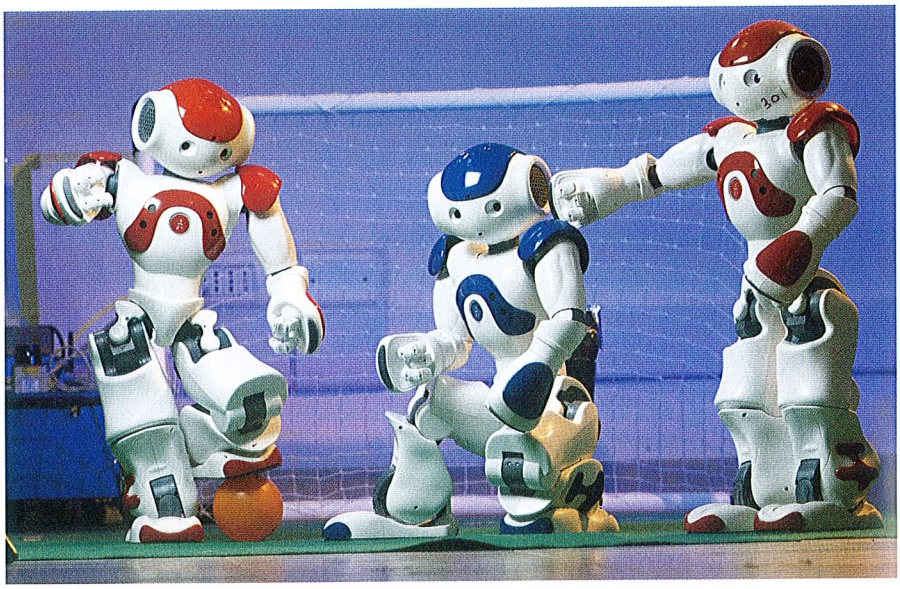
NE PAS CONFONDRE WEB ET INTERNET

Un incomparable outil de modélisation… Par-delà son influence sur notre vie quotidienne, l'informatique joue un rôle sans cesse croissant dans les autres disciplines scientifiques, en offrant des outils pour la modélisation des phénomènes naturels ou artificiels.

Dans notre abécédaire, la lettre M étant déjà prise par le multimédia, la modélisation est évoquée d'une part par les **équations**, d'autre part par la **simulation**. En effet, un modèle n'est rien d'autre qu'un système d'équations qui rend compte, sous une forme mathématique, du phénomène que l'on souhaite étudier. Par exemple, une masse accrochée à un ressort oscille autour de sa position d'équilibre. La modélisation d'un tel système physique simple se ramène à une équation différentielle.

Plus intéressants sont les cas où le système est imparfaitement connu. On construit alors un modèle du système dynamique que l'on souhaite étudier, en choisissant les éléments pertinents et en décrivant leurs interactions par des expressions mathématiques. Cela implique de simplifier ou de négliger certains aspects de la réalité.

Une succession de calculs numériques simule ensuite le comportement du système dynamique selon différentes hypothèses, en faisant varier les conditions initiales ou les valeurs des paramètres. L'astronomie, la géologie, mais aussi les sciences de la vie offrent de nombreuses possibilités de telles expérimentations « *in silico* ». Ainsi, en dynamique des populations, les modèles utilisés servent à étudier l'évolution des populations végétales et animales en interaction dans un même milieu naturel et à prévoir les effets d'une perturbation de leur environnement due au réchauffement climatique ou à la pollution.



© MASSIMO BREGA / LOOKATSCIENCES

▲ **Robots humanoïdes autonomes.** **Ces robots « footballeurs » ont été construits par une firme française, Aldebaran Robotics.**

… et d'optimisation des calculs. La simulation se révèle souvent extrêmement gourmande en temps de calcul, lorsque les paramètres à prendre en compte sont nombreux et complexes. C'est ainsi que des prévisions météorologiques très fiables peuvent être disponibles avec plusieurs jours de retard !

Pour optimiser l'exécution des calculs, une solution est de faire appel à des **grilles** informatiques. Grâce à des **protocoles** spécifiques, les calculs sont répartis sur de nombreux ordinateurs reliés en réseau.

Maîtriser la puissance de calcul est aussi un préalable pour faire fonctionner des environnements de réalité virtuelle. Par exemple, vous enfilez des gants tactiles, mettez des lunettes à cristaux liquides et entrez en interaction avec des objets **virtuels**, parfois superposés au réel. Ces outils et techniques sont exploités dans des jeux vidéo, mais servent aussi à l'entraînement de professionnels, comme les pilotes d'avion ou les chirurgiens, en leur proposant des simulations très réalistes.

Parmi les technologies issues de la recherche en informatique, certaines font rêver, d'autres peuvent faire réfléchir… toutes ont une incidence sur la société dans laquelle nous vivons. C'est pourquoi il est important d'aller au-delà des apparences, de comprendre le travail des chercheurs et de découvrir ce qui se cache derrière des technologies que nous utilisons souvent sans même y penser. ●

SAVOIR +

● http://interstices.info/abc (site de culture scientifique créé par des chercheurs, lancé à l'initiative de l'Inria, en partenariat avec le CNRS, les universités et l'Asti).

La discipline informatique en classe de Terminale S

> PAR JEAN-PIERRE ARCHAMBAULT, PROFESSEUR AGRÉGÉ DE MATHÉMATIQUES, CHARGÉ DE MISSION AU CNDP-CRDP DE PARIS

Ce numéro de *TDC* sur l'informatique s'inscrit dans le contexte de la création d'un enseignement de spécialité optionnel en Tle S : « Informatique et sciences du numérique ». Dans son discours devant le Conseil supérieur de l'Éducation, le 10 décembre 2009, s'exprimant sur la réforme du lycée, le ministre de l'Éducation nationale Luc Chatel a notamment déclaré : « À l'heure de la société de l'information et de la connaissance, la France a besoin plus que jamais de compétences scientifiques en informatique. Aujourd'hui, l'informatique représente 30 % de la recherche et du développement dans le monde. Aujourd'hui, l'informatique est partout. Nous ne pouvons pas manquer ce rendez-vous majeur et c'est la raison pour laquelle nous proposons, en série S, une spécialisation "Informatique et sciences du numérique". »

Le citoyen éclairé participe aux débats de société sur le nucléaire ou les OGM. Pour cela, il dispose d'un appareillage conceptuel que les enseignements de SVT et de sciences physiques lui ont fourni. Dans la société numérique, il doit pouvoir intervenir pleinement sur des problématiques comme celles des « droits d'auteurs et droits voisins dans la société de l'information » (directive européenne DADVSI) ou de la loi Hadopi. Ce sont des domaines complexes (interopérabilité, DRM, code source, adresse IP, etc.), inaccessibles si l'on ne s'est pas approprié le noyau de connaissances stables et transmissibles qui les sous-tendent, si l'on ne s'en est pas fabriqué une représentation mentale opérationnelle.

Une discipline scolaire en tant que telle

Si les sciences physiques sont devenues une discipline scolaire, c'est parce qu'elles fondaient les activités et réalisations de la société industrielle. Or le monde devient numérique : un enseignement de l'informatique est donc incontournable.

Le choix a été fait, lors des quinze dernières années, d'une formation par l'utilisation de l'outil. Cela s'est traduit, depuis 2001, par la mise en place du B2i. Mais l'expérience a montré qu'en matière de formation solide et durable, le simple usage « spontané » ne suffisait pas. L'informatique ne fait pas exception. Maîtriser sa langue maternelle, acquérir une culture mathématique sont des processus longs par lesquels les élèves acquièrent des savoirs que l'humanité a mis des siècles à élaborer. Les notions de lettre, de mot, de nombre sont des abstractions difficiles à appréhender pour un enfant. C'est pourquoi les cours de français et de mathématiques sont irremplaçables. Cela ne signifie pas que l'on ne puisse (ou ne doive) pas s'appuyer sur l'environnement des élèves et leurs pratiques du numérique, mais pour les dépasser, car une certaine dextérité se double souvent d'une faible autonomie, d'un manque de conceptualisation et de compréhension des mécanismes informatiques et d'une très faible verbalisation de ces pratiques.

Des statuts éducatifs divers

Si un enseignement de l'informatique en tant que tel est indispensable au XXIe siècle, d'autres « bonnes raisons » d'utiliser l'ordinateur à l'école ne manquent pas. Celui-ci vient ainsi enrichir la panoplie des instruments péagogiques de l'enseignant. Il se prête à la création de situations de communication « réelles » ayant du sens, notamment pour des élèves en difficulté. Il constitue un élément de motivation. Il favorise l'activité, l'initiative, la créativité. Avec internet, l'accès à des ressources pédagogiques semble infini. L'informatique s'immisce dans des objets, des méthodes et des outils des savoirs constitués, transformant leur « essence », et leur enseignement doit en tenir compte. C'est particulièrement vrai pour les enseignement techniques et professionnels. Mais, peu ou prou, toutes les disciplines sont concernées : les mathématiques, notamment par l'impact des outils de calcul ; les sciences expérimentales avec la simulation et l'expérimentation assistée par ordinateur (EXAO) ; la géographie avec les systèmes d'information géographique (SIG), etc. L'ordinateur est également un instrument de travail à la fois personnel et collectif pour les enseignants, les élèves et la communauté éducative, en particulier dans le cadre des espaces numériques de travail (ENT). Il existe une complémentarité entre l'outil pour enseigner et l'objet d'enseignement qui, loin de s'opposer, se renforcent mutuellement.

Quel programme ?

Il est en cours d'élaboration. Cela étant, chacun sait que l'informatique est structurée autour de quatre grandes dominantes : algorithme, langage, information, machine. On entend parfois dire que des concepts enseignés il y a vingt ans n'auraient plus cours aujourd'hui. Mais le monde bouge plus vite que les fondamentaux de la connaissance scolaire. Le théorème de Pythagore est vieux de vingt-cinq siècles, ce qui n'empêche pas les collégiens de l'étudier encore de nos jours ! Socrate et Platon n'ont pas été rendus caducs par Descartes et Kant. Molière est toujours très actuel, car universel et « éternel ». La programmation est partie intégrante de l'informatique, moyen irremplaçable pour comprendre l'intelligence de la science informatique et outil pertinent pour les autres disciplines. Si elle s'enrichit en permanence, pour autant, du point de vue de la culture scolaire en classe de Terminale, elle n'a pas vieilli.

On pourra se référer aux documents explicitant les grandes lignes d'un programme pour le lycée, élaboré par le groupe ITIC de l'ASTI (www.epi.asso.fr/revue/articles/a0712f.htm et www.epi.asso.fr/revue/editic/asti-itic-lycee-prog.htm), ainsi qu'aux deux séquences pédagogiques de ce numéro de *TDC*.

Tout a un reflet numérique

> PAR GILLES DOWEK, THIERRY VIÉVILLE ET EMMANUEL BACCELLI, CHERCHEURS A L'INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE (INRIA), JEAN-PIERRE ARCHAMBAULT ET BENJAMIN WACK, ENSEIGNANTS

A L'information et son reflet

À chaque instant, notre cerveau capte, enregistre et traite des informations. Ainsi pouvons-nous regarder, mémoriser, analyser, puis retranscrire une image, un texte ou un son. Depuis longtemps, l'homme a cherché à capter et à reproduire ce reflet de la réalité par un dispositif technique. Il a donc inventé le téléphone, la photographie, le cinéma, etc. Mais qu'en est-il aujourd'hui avec les machines que nous utilisons, qui sont des dispositifs de calcul programmé ?

Prenons l'exemple d'une calculette électronique : quand nous disons qu'elle « calcule », ses circuits électriques s'ouvrent et se ferment de telle sorte que l'affichage corresponde au résultat du calcul effectué. Mais le fait que cela prenne le sens d'un calcul tient à l'humain qui utilise cette calculette ; la machine ne « comprend » évidemment pas les différentes opérations, qui s'exécutent uniquement grâce aux propriétés électriques de ses circuits : elle ne possède que le reflet électronique du calcul.

INFORMATION, HASARD ET COMPLEXITÉ

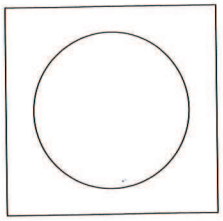
Regardons cette suite de chiffres : 14159 26535 89793 23846 26433 83279 50288 41971. Il y a sûrement plus d'informations dans cette suite-là que dans celle-ci : 00000 00000 00000 00000 00000 00000 00000 00000. Mais en quoi la première est-elle plus complexe ? Est-elle aléatoire, chaotique, sans ordre ni régularité ? Ou bien fortement structurée, riche en informations « utiles » ?

En fait, si nous regardons bien le début de cette suite, nous reconnaissons les premières décimales de π, peut-être le nombre le plus célèbre de toute l'histoire des mathématiques : autant dire que son information est très utile… du moins pour les géomètres et les mathématiciens ! En tout cas, il est bien difficile de retenir les chiffres de la première suite. Mais, pour la deuxième, il suffit de retenir : « huit fois cinq zéros ». Cet exemple, cité par Jean-Paul Delahaye dans son ouvrage *Information, complexité et hasard* (Hermes Science, 1999, coll. « Langue, raisonnement, calcul »), est très instructif, car il nous montre que nous devons bien distinguer le contenu *brut* d'une information, qui ne dépend pas du sens, et la *valeur* d'une information qui, elle, dépend du but fixé.

B Fabriquer un reflet numérique à partir d'un objet réel

DÉCRIRE UN DESSIN PAR UNE DÉFINITION STRUCTURÉE

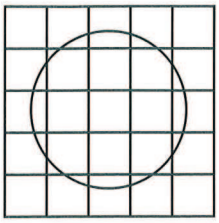
Comment décrire le dessin ci-dessous ? Une solution est de dire : « Cette image est formée d'un cercle. » Nous pouvons même être plus précis et indiquer les coordonnées du centre du cercle, son rayon, sa couleur, l'épaisseur du trait, etc. À partir de cette description, il est possible de reconstituer le dessin.



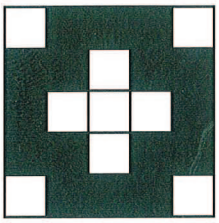
Mais n'utiliser que des mots serait bien moins pratique pour le dessin suivant. En effet, comment décrire facilement et en détail ces traits qui semblent tracés au hasard ? Qu'en serait-il avec un dessin représentant un objet réel ou un visage ?

DÉCRIRE AVEC DES NOMBRES BINAIRES

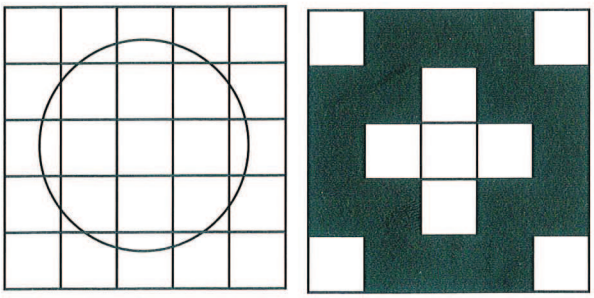
Une autre méthode, qui a l'avantage de pouvoir être utilisée pour n'importe quel dessin, consiste à superposer un quadrillage au tracé. Chaque case de ce quadrillage est appelée « pixel » (contraction de l'anglais *picture element*).



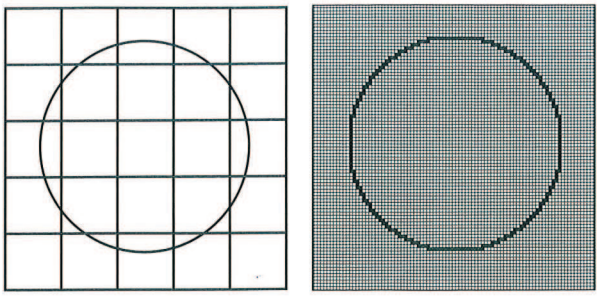
On noircit ensuite les pixels qui contiennent une portion de trait :



Puis, il suffit d'indiquer la couleur de chacun des pixels, en les lisant de gauche à droite et de haut en bas, comme un texte. Cela donne : blanc, noir, noir, noir, blanc, noir, noir, blanc, noir, noir, noir, blanc, blanc, blanc, noir, noir, noir, blanc, noir, noir, blanc, noir, noir, noir, blanc. Cependant, cette description est assez approximative, et l'on constate une grande différence entre ces deux images :



Il est possible de rendre la description plus précise en utilisant un quadrillage, non plus de 5 x 5, mais de 100 x 100 pixels :



À partir de quelques millions de pixels, nous ne serions plus capables de faire la différence entre les deux images. Cette méthode est approximative, mais universelle : n'importe quel dessin, même très compliqué, se décrit exactement comme un dessin simple. Celui que nous venons de voir se décrit donc par une suite de mots : « blanc » ou « noir ». Comme seuls ces mots sont utilisés, nous pouvons être plus économes et remplacer chacun d'eux par un seul symbole : le mot « noir » par la lettre *n* ou le chiffre 0, et le mot « blanc » par la lettre *b* ou le chiffre 1. Le dessin ci-dessus, avec une grille de 5 x 5, se décrit alors par une suite de 25 chiffres : 1000100100011100010010001.

Ces différentes descriptions sont équivalentes : le dessin se décrit toujours par une suite de symboles choisis parmi deux. Leur nom importe peu, et nous pourrions les appeler « noir » et « blanc », *a* et *b*, 0 et 1, pile et face, - et +, Dupond et Dupont, etc. Cela ne changerait pas grand-chose. En général, on choisit 0 et 1. Ces symboles sont appelés « chiffres binaires » ou « bits » (abréviation de l'anglais *binary digits*, « chiffres binaires ») ; une suite de 0 et de 1, « suite binaire » ou, parfois, « mot binaire ». Notre dessin se décrit ainsi avec 25 chiffres binaires, ou encore 25 bits.

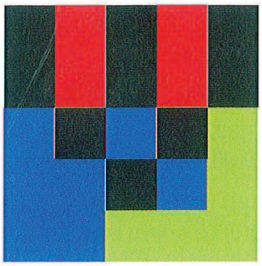
LE POINT-CLÉ : CONVENIR D'UN STANDARD

Bien entendu, il faut que nous soyons tous d'accord pour décrire l'ensemble des dessins de la même façon et décider, comme nous l'avons fait dans l'exemple précédent, que le noir est représenté par 0, le blanc par 1, et que les pixels se lisent de gauche à droite et de haut en bas. Il faut donc mettre au point un standard. Le codage de l'information est avant tout une affaire de convention. Regardons alors comment nous pourrions convenir de représenter d'autres objets.

C Décrire numériquement toutes sortes d'objets

DU DESSIN AUX IMAGES PHOTOGRAPHIQUES

Si nous voulons décrire une image en couleur – par exemple, une image dans laquelle chaque pixel peut être noir, rouge, vert ou bleu –, une première méthode consiste à décrire cette image par une suite dont chaque élément représente l'un des quatre mots. Prenons l'image ci-dessous :



Elle se décrit de cette manière : noir, rouge, noir, rouge, noir, noir, rouge, noir, rouge, noir, bleu, noir, bleu, noir, vert, bleu, bleu, noir, bleu, vert, bleu, bleu, vert, vert, vert. Comme pour les images en noir et blanc, on peut être plus économe et décrire chaque couleur par une seule lettre ou un seul chiffre. Par exemple, en utilisant la correspondance :

n	noir
r	rouge
v	vert
b	bleu

L'image se décrit donc par la suite : nrnrnnrnrnbnbnvbbnbvbbvvv. Cela fonctionne à condition, bien entendu, que celui qui code et celui qui décode cette image suivent parfaitement le standard proposé ici. Cette méthode utilise quatre symboles différents. Il est possible aussi de décrire l'image par une suite binaire, c'est-à-dire dans laquelle on utilise seulement deux symboles et non quatre. Mais comme il y a plus de deux couleurs, il est impossible de décrire chaque couleur par un seul chiffre binaire : 0 ou 1. En revanche, on peut décrire chaque couleur par une suite de deux bits. Par exemple :

00	noir
01	rouge
10	vert
11	bleu

Cette image se décrit alors par la suite : 00 10 00 10 00 00 10 00 10 00 01 00 01 00 11 01 01 00 01 11 01 01 11 11, donc 25 x 2 = 50 bits. Les espaces entre chaque couple de valeurs binaires sont inutiles, sauf pour en faciliter la lecture.

Une fois choisi le nombre de pixels (ici, 5 x 5 ; pour une photographie numérique, il en faudrait quelques millions) et le nombre de bits sur lequel on représente la couleur de chaque pixel (ici, 2 bits ; en réalité de 8 à 32 bits, selon la palette des couleurs), toutes les images se représentent par une suite binaire de la même longueur : cette longueur s'obtient simplement en multipliant le nombre de pixels de l'image par le nombre de bits utilisés pour décrire la couleur d'un pixel. C'est ainsi que se décrivent les photographies dans les ordinateurs et les autres machines numériques.

DÉCRIRE DES TEXTES PAR DES SUITES BINAIRES

Décrire un texte semble *a priori* plus simple que décrire une image, puisqu'un texte est déjà une suite de symboles, les lettres de l'alphabet. L'alphabet latin, que l'on utilise en français, contient 26 lettres. Mais, avec les majuscules et les minuscules, les lettres accentuées, les chiffres et les symboles de ponctuation, écrire en français nécessite plutôt une centaine de symboles.

Il faut aussi, cependant, pouvoir décrire toutes les autres langues. Par exemple, les 28 symboles de l'alphabet arabe, avec ses symboles complémentaires, comme on les voit sur une ancienne machine à écrire. Ainsi la phrase « Tout se code en binaire » s'écrira-t-elle « دوكلا ايئانث يف ءيش لك » (qui se lit de droite à gauche) et se codera-t-elle comme suit : 1011 1100 1001 1111 0000 1010 1111 0111 1101 0110 0101 1111 0110 1100 1011 1110 1000, en utilisant les codes binaires à 4 bits des 16 lettres suivantes :

ء	آ	أ	ؤ	إ	ئ	ا	ث	د	ش	ف	ك	ل	ن	و	ي
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Ce code de 16 lettres, nous nous sommes amusés à l'inventer uniquement pour cette présentation. Cependant, là encore, pour partager les informations, il ne faut pas utiliser un code particulier, comme celui-ci, mais un code standard, adopté par tous, et qui représente tous les caractères de la langue utilisée. Les caractères les plus courants en Occident, ceux que nous tapons au clavier, sont codés sur 8 bits, avec un code universel, dit ASCII (*american standard code for information interchange*) étendu. Pour coder tous les textes du monde, c'est l'Unicode, contenant plus de 120 000 caractères, qui a été retenu. Ce standard utilise le code ASCII et des codes complémentaires à 16 ou 24 bits.

D Décrire des nombres

Le codage des valeurs numériques est de nature différente, car ce n'est pas un code arbitraire « standard », mais un choix issu de l'arithmétique. En effet, il serait possible de représenter un nombre tel que 2741 comme la succession des chiffres 2, 7, 4 et 1, puis de coder chacun de ces chiffres par son équivalent ASCII pour obtenir : 00110001 00110111 00110100 00110001. Mais, si un tel codage représente bien la chaîne « 2741 », il ne représente pas le nombre 2741 et ne permet pas d'effectuer des calculs arithmétiques tels que l'addition et la multiplication. Pour obtenir une représentation utilisable dans des calculs, il suffit cependant de passer de la numération en base dix classique à la numération en base deux.

Le tableau suivant donne les dix premiers nombres en base deux dans la colonne du milieu et leur représentation binaire dans la colonne de droite :

$\mathbf{0}\times 8+\mathbf{0}\times 4+\mathbf{0}\times 2+\mathbf{0}\times 1=$	0	**0000**
$\mathbf{0}\times 8+\mathbf{0}\times 4+\mathbf{0}\times 2+\mathbf{1}\times 1=$	1	**0001**
$\mathbf{0}\times 8+\mathbf{0}\times 4+\mathbf{1}\times 2+\mathbf{0}\times 1=$	2	**0010**
$\mathbf{0}\times 8+\mathbf{0}\times 4+\mathbf{1}\times 2+\mathbf{1}\times 1=$	3	**0011**
$\mathbf{0}\times 8+\mathbf{1}\times 4+\mathbf{0}\times 2+\mathbf{0}\times 1=$	4	**0100**
$\mathbf{0}\times 8+\mathbf{1}\times 4+\mathbf{0}\times 2+\mathbf{1}\times 1=$	5	**0101**
$\mathbf{0}\times 8+\mathbf{1}\times 4+\mathbf{1}\times 2+\mathbf{0}\times 1=$	6	**0110**
$\mathbf{0}\times 8+\mathbf{1}\times 4+\mathbf{1}\times 2+\mathbf{1}\times 1=$	7	**0111**
$\mathbf{1}\times 8+\mathbf{0}\times 4+\mathbf{0}\times 2+\mathbf{0}\times 1=$	8	**1000**
$\mathbf{1}\times 8+\mathbf{0}\times 4+\mathbf{0}\times 2+\mathbf{1}\times 1=$	9	**1001**

Si l'on observe bien ce codage, on constate quelque chose de plus : il est le résultat d'un calcul, montré dans la colonne de gauche. Le codage binaire d'un nombre (que nous écrivons quasiment toujours en base dix, en utilisant 10 chiffres) est ce même nombre, écrit en base deux.

DU CODAGE BINAIRE AU NOMBRE ENTIER POSITIF

Voici le calcul qui permet de retrouver un nombre à partir de son code binaire : additionner les bits, chacun multiplié par la puissance de deux qui lui correspond, comme ci-dessous, donne le nombre en base dix.

$$\mathbf{1} \times 8 + \mathbf{0} \times 4 + \mathbf{0} \times 2 + \mathbf{1} \times 1 =$$
$$\mathbf{1} \times 2^3 + \mathbf{0} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0 = \quad \mathbf{9}$$
$$((\mathbf{1} \times 2 + \mathbf{0}) \times 2 + \mathbf{0}) \times 2 + \mathbf{1} =$$

À la première ligne, nous avons multiplié respectivement par 1, 2, 4, 8 de droite à gauche les bits, qui représentent 9. À la seconde ligne, nous nous sommes aperçus que 1, 2, 4, 8 sont, en fait, les puissances de 2 : 2^0, 2^1, 2^2, etc. À la troisième ligne, nous factorisons 2 autant que possible ; cette écriture, avec ses parenthèses bien placées, nous montre qu'il suffit de multiplier par deux, puis d'additionner les bits de gauche à droite pour trouver le nombre en base dix. Voilà donc un calcul, une méthode mécanique appelée « algorithme », qui permet de passer du codage binaire au nombre dans sa représentation usuelle.

DU NOMBRE ENTIER POSITIF À SON CODAGE BINAIRE

Réciproquement, il existe un calcul pour obtenir le code binaire de n'importe quel nombre entier. Si nous notons 9 % 2 = 1, le reste de la division par 2 de 9, ou 4 % 2 = 0, le reste de la division de 4 par 2, etc., alors le calcul suivant, où nous calculons successivement le reste de la division par deux, puis divisons par deux, donne… le codage binaire du nombre 9 :

$$9 \% 2 = \mathbf{1}$$
$$(9/2 = 4) \% 2 = \mathbf{0}$$
$$(4/2 = 2) \% 2 = \mathbf{0}$$
$$(2/2 = \mathbf{1}) \% 2 = \mathbf{1}$$

REPRÉSENTER DES ENTIERS POSITIFS DE TOUTES TAILLES

Bien sûr, cela marche avec tous les nombres entiers positifs, et nous pouvons facilement calculer combien de bits sont nécessaires pour les coder :

Entiers	0 à 1	0 à 3	0 à 7	0 à 15	0 à 255	0 à 1023	0 à 4 milliards	0 à plus de 1,8 x 10^{20}
Codés sur	1 bit	2 bits	3 bits	4 bits	8 bits	10 bits	32 bits	64 bits

La plupart de nos ordinateurs utilisent aujourd'hui 32 ou 64 bits pour coder les nombres entiers ; ils peuvent donc coder directement des nombres très importants, et, si nécessaire, en utilisant davantage de bits… des nombres vertigineusement grands !

CALCULER DIRECTEMENT SUR LES NOMBRES BINAIRES

Nous pouvons donc représenter des nombres entiers positifs par la suite de bits de leur représentation binaire. Ce codage est très précieux, car il permet de faire directement des opérations sur ces nombres. Par exemple, additionner deux entiers positifs revient à additionner leur représentation binaire : on obtient directement le codage binaire du résultat. De même pour toutes les autres opérations numériques, pour comparer ces entiers entre eux, etc. Souvenons-nous que toutes les valeurs numériques sont non seulement stockées en binaire, mais aussi manipulées en binaire dans l'ordinateur.

REPRÉSENTER DES NOMBRES ENTIERS NÉGATIFS ET POSITIFS

Jusqu'ici, nous n'avons considéré que des nombres entiers positifs. Comment représenter des nombres entiers soit positifs, soit négatifs ? Choisissons de représenter le signe par un bit supplémentaire :

0	+
1	-

Avec cet ajout, nous représentons la valeur absolue du nombre comme un entier positif, et son signe avec ce bit en plus. Nous pouvons alors définir toutes les opérations numériques que nous connaissons en tenant compte de ce signe.

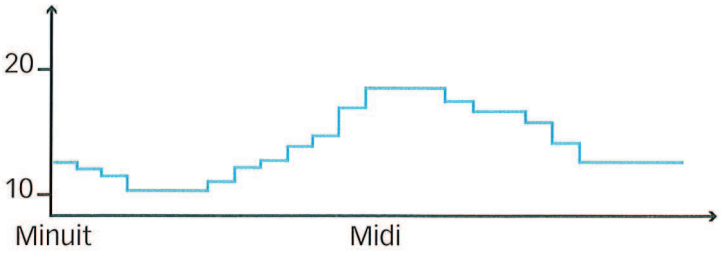
REPRÉSENTER LES NOMBRES RÉELS

Pour représenter un nombre réel – s'il s'agit d'un nombre décimal, par exemple 3,1416 –, on utilise la « notation scientifique » 31416 10^{-4}, qui code d'une part l'exposant (ici, $^{-4}$) et d'autre part les décimales du nombre, plus exactement sa mantisse (ici, 31416), sous forme de deux entiers, codés en binaire et mis bout à bout. C'est de cette façon que sont codés les nombres dans les calculatrices et les ordinateurs. Les calculs numériques se font alors directement sur les nombres binaires et sont ensuite transcrits pour afficher le résultat attendu. Mais attention : les fractions comme 10/3 = 3,333 333 333… ou encore les nombres réels comme π = 3,141 592 653 589 793 238 462 643 383 279 502 884 197 1… ont une suite infinie de décimales ! Cela veut dire que nous ne pouvons pas représenter leur valeur numérique, mais uniquement une *approximation* de cette valeur. De ce fait, *tous* les calculs que nous ferons avec un ordinateur ne seront jamais justes ! Ils ne seront pas vraiment faux non plus, ils seront « approximatifs ».

E Décrire toutes les informations humaines

CODER UNE MESURE PHYSIQUE

Comment coder la courbe de température de votre chambre au fil du jour ? On peut coder le nombre qui correspond à la mesure de température à chaque heure et les mettre bout à bout.



Ces 24 nombres vont donner une bonne approximation du chaud et du froid… et du fait que votre chauffage est visiblement coupé la nuit.

CODER UN SON

Le son est représenté par une courbe qui correspond aux vibrations de la voix, des bruits ou des instruments. Coder un son revient à coder une suite de valeurs numériques qui correspondent à cette courbe. Bien sûr, il faut beaucoup de valeurs (environ 10 000 par seconde) pour obtenir une bonne approximation de toutes les vibrations sonores ; il faut 16 bits pour coder correctement la valeur de l'intensité et de la fréquence du son à chaque instant. Et pour coder une minute de son : 16 bits x 10 000 valeurs/seconde x 60 secondes = presque 10 millions de bits.

C'est ce qui se passe dans les lecteurs MP3, par exemple, à une différence importante près. En effet, le codage du son à l'aide du standard MP3 est « astucieux » : il met en œuvre des méthodes de compression avec perte qui tiennent compte de la perception humaine. Cette approximation supplémentaire permet de gagner de la place dans la mémoire et du temps lors de la communication.

CODER DE LA MUSIQUE

Il suffit de choisir un code binaire pour chaque note – sa hauteur (aiguë ou grave) et sa durée –, chaque instrument. Ensuite, en mettant bout à bout ces codes, on obtient la séquences des notes. On code ainsi la partition musicale et son interprétation. Un tel codage existe ; il est standard et se nomme MIDI (*musical instrument digital interface*).

CODER UN FILM

Un film n'est jamais qu'une séquence d'images. Schématiquement, il suffit donc de coder celles-ci et de placer des suites de bits bout à bout. Sans oublier de coder aussi le son. Là encore, il existe des astuces pour regrouper les parties qui se ressemblent, pour gagner de la place et du temps. Mais le principe reste le même.

Qu'avons-nous découvert ici ? La polyvalence des ordinateurs et des autres machines numériques : ils traitent des données très diverses, mais toujours décrites de la même manière, par des suites binaires utilisant deux symboles.

F Pourquoi utiliser un codage binaire ?

IL FAUT AU MOINS DEUX SYMBOLES

Pouvons-nous utiliser moins de deux symboles pour décrire les images, les textes ou les nombres ? Non, car si nous avons un seul symbole, par exemple 0, une fois le nombre de bits fixé à 10, 100 ou 1 million, il n'y a plus qu'une seule suite de 10, 100 ou 1 million de symboles : 0000000000000000000000000... qui ne peut donc décrire qu'un seul objet ! L'information commence avec la dualité – opposition et complémentarité – du 0 et du 1, du chaud et du froid, du bas et du haut, etc. Quand on a un symbole unique, tout est uniforme ; il n'y a pas d'information.

EN UTILISER DEUX, C'EST BIEN PRATIQUE...

En revanche, nous aurions pu choisir d'utiliser plus de deux symboles : les 10 chiffres arabes, les 22 lettres de l'alphabet phénicien, les 24 lettres de l'alphabet grec ou les 26 lettres de l'alphabet latin. Mais n'utiliser que deux symboles a plusieurs avantages. Le premier est que ce choix est indépendant de la nature des informations décrites. Il aurait été dommage de concevoir des ordinateurs grecs qui utilisent 24 symboles et des ordinateurs latins qui en utilisent 26.

Par ailleurs, la description de l'information avec deux symboles est plus simple à mettre en œuvre dans un ordinateur. Cela se réalise électriquement avec un interrupteur ouvert ou fermé, un courant électrique présent ou absent, etc. Ce choix rend aussi les ordinateurs plus robustes : s'il y avait plus de deux états intermédiaires, le système physique pourrait les mélanger, et donc provoquer plus facilement des erreurs. En se limitant à 0 et 1, ce risque est minimisé.

... ET CELA PERMET AUSSI DE MESURER LA QUANTITÉ D'INFORMATION

Le nombre de bits est exactement la *quantité brute d'information qui a été codée.* Pour comprendre cette idée, jouons au jeu du portrait : devinez le nom d'une personne en ne posant que des questions où l'on répond par « oui » ou par « non », en suivant l'exemple ci-dessous :

Qui ?	Genre ?	*Look* emo ?	Moins de 15 ans ?	
Pierre	gars	oh non !	oui	'001'
Nadia	fille	good-good	oui	'111'
David	gars	énormément	non	'010'
Hamdi	gars	certes	oui	'011'
Marie	fille	surtout pas	non	'100'
Adèle	fille	évidemment	non	'110'
Ming Yu	fille	beurk	oui	'101'
Igor	gars	horreur !	non	'000'

En trois questions, « Est-ce une fille ?, Aime-t-elle le *look* emo ?, A-t-elle moins de 15 ans ? », vous allez forcément deviner, parmi les huit personnes, celle qui correspond au portrait, car vous aurez toute l'information utile ici. D'ailleurs, nous avons symbolisé ce fait en utilisant un bit – reporté dans la colonne de droite – pour chacune des questions binaires.

Deux questions binaires, c'est-à-dire deux bits d'information, ne suffisent pas ici. Si vous ne savez pas que la personne recherchée a moins de 15 ans, alors Adèle et Nadia ou Igor et Pierre, par exemple, ne peuvent pas être distingués, puisqu'ils pensent la même chose du *look* emo. On constate bien ici que le nombre de bits correspond au nombre de questions binaires à poser pour deviner toute l'information, et donc à la taille en information. Prenons un autre exemple : devinons l'âge d'un élève du secondaire, entre 11 et 18 ans. Nous pourrions poser huit questions, « A-t-il 11 ans ? A-t-il 12 ans ? A-t-il 13 ans ? A-t-il 14 ans ? A-t-il 15 ans ? A-t-il 16 ans ? A-t-il 17 ans ? A-t-il 18 ans ? », avec le risque, si nous manquons de chance, de devoir poser les huit questions avant de connaître la solution.

Voici une meilleure méthode : demandons d'abord si l'élève a moins de 15 ans. Si la réponse est « oui », alors nous savons que l'élève a entre 11 et 14 ans ; si la réponse est « non », alors nous savons que l'élève a entre 15 et 18 ans. Dans les deux cas, nous sommes passés d'une fourchette de 8 ans à celle de 4 ans. En divisant l'intervalle de recherche par deux, nous avons gagné un « atome » d'information, un bit d'information. Si nous continuons à poser une question qui divise l'intervalle de recherche par deux, nous arrivons à une fourchette de deux ans, puis à la troisième question, à une fourchette d'un an, etc., et nous avons trouvé la solution : l'âge de l'élève. Nous voyons bien ici que ce choix parmi 8 valeurs correspond à 3 questions, donc 3 bits d'information, chaque bit correspondant, de gauche à droite, aux réponses « oui » (= 1) ou « non » (= 0) aux trois questions.

11	12	13	14	15	16	17	18
'111'	'110'	'101'	'100'	'011'	'010'	'001'	'000'

Un «atome» d'information est donc un bit, la réponse «oui» ou «non» à une question binaire, le choix entre deux options, etc. La taille en information du choix entre plusieurs options correspond au nombre de bits pour le coder. Par exemple, choisir entre quatre couleurs : noir, rouge, vert, bleu nécessite deux bits. Choisir entre les sept couleurs que nous voyons aujourd'hui dans l'arc-en-ciel nécessite trois bits, tandis que le code 000 n'est pas utilisé ici, ce qui importe peu.

violet	indigo	bleu	vert	jaune	orange	rouge
'111'	'110'	'101'	'100'	'011'	'010'	'001'

La taille en information de deux informations indépendantes s'additionne. Par exemple, si nous codons l'âge d'un élève du secondaire sur 3 bits et son genre (fille/garçon), alors il faudra en tout 4 bits. Mais la taille en information de deux informations redondantes ne s'additionne pas. Par exemple, si nous codons l'âge d'un élève du secondaire sur 3 bits et le fait qu'il ait le droit ou non de conduire un scooter sur 1 bit, puisque nous savons que toute personne de 14 ans et plus peut conduire un scooter, il est inutile d'ajouter cette information, car elle est «contenue» dans l'âge : il y a donc toujours 3 bits d'information.

Nous avons ainsi découvert une mesure physique très intéressante : de même que la chaleur se mesure avec la température en degrés, la tension électrique en volts, la quantité brute d'information se mesure en bits. Cette mesure correspond tout simplement au nombre de bits minimal nécessaire pour stocker l'information sans la tronquer. Évidemment, elle ne dit rien de la valeur de l'information, ni de sa complexité. Par exemple, la séquence binaire 1100100100001111110110101010001000100001011010001100001000110100 est constituée de 64 bits. A-t-elle une grande valeur ? Aucune, si les 0 et les 1 ont été choisis sans raison. A-t-elle une grande complexité, c'est-à-dire est-elle difficile à calculer ? Non, si nous avons juste appuyé au petit bonheur la chance sur le 0 et le 1 du clavier. Mais cette séquence correspond au codage binaire, au milliardième près, du nombre le plus célèbre en mathématiques : le nombre π. Voilà qui le rend plus précieux et qui indique qu'il est plutôt complexe à calculer. Et pourtant, il est toujours constitué de 64 bits. C'est le contenu brut en information qui est mesuré ici, indépendamment de sa valeur ou de sa complexité.

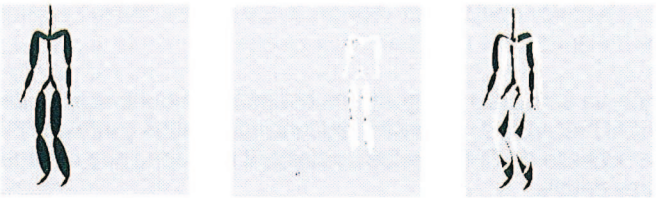
G Commencer à manipuler les objets numériques que sont les images

AJOUTER DEUX IMAGES : L'EFFET DE CALQUE

En ajoutant la valeur de chaque pixel de la première image à celle de chaque pixel de la seconde image, on obtient une image qui amalgame les valeurs des deux, un peu comme un effet de calque. Si nous avions considéré des pixels de couleurs, nous aurions mélangé les couleurs et obtenu des combinaisons diverses.

SOUSTRAIRE DEUX IMAGES : UN DÉTECTEUR DE MOUVEMENT

À quoi peut correspondre maintenant la « soustraction » de deux images, c'est-à-dire le fait de soustraire à chaque pixel de la première image la valeur du pixel correspondant à la seconde ? Il est facile de voir que si les pixels ont une même valeur, alors la différence sera nulle : si rien n'a changé d'une image à l'autre, il n'y aura pas de différence. Si, au contraire, quelque chose a changé, alors la différence sera ou positive (disons plutôt noire) ou négative (disons plutôt blanche), et nous détecterons ainsi les changements, comme nous le voyons ci-dessous pour deux images entre lesquelles la figurine a quelque peu bougé :



Les ingrédients des algorithmes

> PAR GILLES DOWEK, THIERRY VIÉVILLE ET EMMANUEL BACCELLI, CHERCHEURS À L'INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE (INRIA), JEAN-PIERRE ARCHAMBAULT ET BENJAMIN WACK, ENSEIGNANTS

A Introduction

Un ordinateur peut effectuer des opérations très variées, sur des types de données tout aussi variés : des nombres, des lettres, des images, des sons, des textes, des vidéos, comme le montre la séquence pédagogique précédente. Il peut être utilisé pour retoucher une photographie, la mettre sur un blog ou un site Web, la conserver dans un album, etc. Un ordinateur est une machine polyvalente : tout ce qui s'automatise peut être programmé sur un ordinateur. À l'inverse des machines à café ou des aspirateurs, qui ne servent qu'à une seule chose : faire le café ou aspirer la poussière.

En revanche, un ordinateur ne « sait » quasiment rien faire. Il doit être « programmé » pour retoucher une photographie, la mettre sur un blog ou un site Web. Cette notion de programme est également ce qui distingue l'ordinateur de certaines machines, comme les calculatrices non programmables, qui donnent une illusion de polyvalence dans la mesure où elles peuvent effectuer des tâches multiples. Cependant, ces tâches sont fixées une fois pour toutes : on ne peut pas programmer une calculatrice pour lire une vidéo, alors que l'on peut programmer un ordinateur pour faire tout ce que fait une calculatrice.

En quoi consiste l'écriture d'un tel programme ? Le plus important est de bien spécifier ce que l'on cherche à faire. Ce qui compte, ici, c'est la méthode mise en œuvre : l'algorithme. Il faut ensuite l'écrire de façon à le transmettre à l'ordinateur. C'est pour cela qu'on utilise un langage de programmation. Les documents mis en ligne sur le site *Interstices* (www.interstices.info), « Algorithme, mode d'emploi » et « Demandez le programme », offrent une introduction à ce sujet. Voir également http://javascool.gforge.inria.fr/proglet : le site propose une activité scolaire qui met en application les éléments décrits dans cette séquence.

B Algorithme... une recette de cuisine ?

On appelle « algorithme » la méthode, la façon systématique de procéder, pour faire quelque chose : trier des objets, situer des villes sur une carte, multiplier deux nombres, extraire une racine carrée, chercher un mot dans le dictionnaire, etc. Il se trouve que certaines actions mécaniques – peut-être toutes – se prêtent bien à cette décortication. On peut les décrire de manière générale, identifier des procédures, des suites d'actions ou de manipulations précises à accomplir séquentiellement. C'est cela, un algorithme : le concept qui traduit la notion intuitive de procédé systématique, applicable mécaniquement, sans réfléchir, en suivant un mode d'emploi précis.

Un algorithme, c'est donc « presque » une recette de cuisine. Prenons l'exemple d'un quatre-quarts au citron ou au chocolat. Il suffit d'utiliser deux œufs, le même poids de farine, de beurre et de sucre, d'ajouter le parfum, de mélanger le tout et de mettre au four une « petite » demi-heure. Mais que se passerait-il si nous confiions cette recette à un dispositif mécanique dépourvu d'intelligence (robot, ordinateur, automate) ? Il réaliserait notre recette *exactement* comme nous venons de le spécifier. « Une petite demi-heure dans le four » ? Voilà notre dispositif mécanique bloqué : petite ? Par rapport à quoi ? De combien ? Sans compter que personne n'a précisé d'allumer le four, ni de verser le mélange dans un moule !

En résumé, toute instruction ambiguë, incomplètement spécifiée, va échouer. Un algorithme est donc bien une recette de cuisine... mais c'est une recette à définir, à spécifier. En clair, un dispositif mécanique ne possède aucune information contextuelle, aucune connaissance préalable, aucun de ces éléments qui font la richesse de l'intelligence humaine. Il faut donc que chaque étape soit entièrement et explicitement spécifiée, dans ses moindres détails.

C De retour en cuisine

Nous allons demander au robot d'exécuter la séquence de toutes les opérations (par exemple : ouvrir le vaisselier, sortir le grand compotier, ouvrir le réfrigérateur, se tourner vers le compartiment supérieur de la porte, sortir le beurre, ouvrir le placard, se tourner vers la troisième étagère à droite, prendre la farine, etc.). Notre algorithme est donc un chemin à parcourir pas à pas, une **séquence d'instructions.** Pour peu que chaque instruction ait été bien programmée précédemment, nous commençons à voir apparaître la pâte du gâteau. À condition que tout se passe exactement comme prévu !

Que faire si le compotier n'est pas à sa place, ou s'il ne reste plus de beurre ? « S'il n'y a plus de beurre, il suffit de prendre de la margarine, sinon il vaut mieux arrêter de faire le gâteau… » Voilà que notre recette n'est plus une simple séquence d'instructions, ce n'est plus un simple chemin, mais un itinéraire avec des « carrefours », où le choix du chemin se fait en fonction d'une condition (pas de beurre, ou ni beurre ni margarine). L'algorithme est donc un réseau d'**instructions conditionnelles,** à parcourir au fur et à mesure, en bifurquant à chaque condition.

Le quatre-quarts est en bonne voie. Mais comment expliquer au robot qu'il faut tourner une trentaine de fois les ingrédients dans le compotier sans devoir recopier une bonne trentaine de fois l'instruction ? Car le but n'est pas de tourner trente fois, mais plutôt de tourner jusqu'à ce que la pâte soit bien homogène. Nous avons alors besoin d'une autre construction, une boucle de la forme : « Tant que la pâte n'est pas homogène, tourner les ingrédients dans le compotier. » Avec cette **boucle d'instruction,** il est possible de faire durer ou répéter une opération autant de fois que nécessaire. De même, nous pouvons écrire : « Cuire au four tant que la pâte reste liquide » ou « Fabriquer 200 quatre-quarts pour tout le quartier », en l'exprimant de manière concise.

Enfin, nous voilà avec un quatre-quarts au citron ou au chocolat. Va-t-il falloir réécrire la recette complète pour *chaque* parfum ? Alors que seule l'étape « ajouter trois zestes de citron » ou « ajouter 50 grammes de poudre de chocolat » change d'une recette à l'autre ? Certes pas ; pour l'éviter, nous avons besoin d'introduire la notion de **variable** ou de *paramètre*. La recette doit être *paramétrée* par la variable « parfum » et en fonction de sa *valeur* (« citron », « chocolat »). Une instruction conditionnelle changera juste la ligne de la recette liée au parfum. Nous obtenons ainsi, exprimée de manière concise, la recette de tous les quatre-quarts possibles, qui ne diffèrent que par leur parfum.

À présent, prenons un peu de recul. Nous avons utilisé des instructions comme « ouvrir le réfrigérateur » ou « tourner les ingrédients dans le compotier », qui ont sûrement dû être programmées par une autre personne, et nous avons produit l'instruction « faire un quatre-quarts au goût de $parfum » (le signe $ indique qu'il s'agit d'une variable), où « $parfum » prend la valeur « citron » ou « chocolat ». Bref, nous avons utilisé des fonctionnalités prédéfinies, des « briques de base », pour créer une autre fonctionnalité pouvant être utilisée par une personne qui prépare un menu, par exemple. Cette dernière construction, qui consiste à regrouper un bloc d'instructions dans une **fonction,** va permettre de réutiliser chaque fonctionnalité, comme dans un Lego, pour réaliser une construction logicielle. Ce qui est très intéressant, ici, c'est qu'il suffit de ces cinq ingrédients pour décrire tous les algorithmes de façon efficace. N'oublions pas toutefois une autre différence importante entre les recettes de cuisine et les programmes. Ces derniers sont destinés à être lus par des humains, mais aussi par des ordinateurs. Ils doivent donc être écrits dans des langages très particuliers : les langages de programmation, qui conditionnent la façon de présenter les ingrédients.

D Et les *bugs*, dans tout ça ?

Nous parlons de « recette de cuisine », mais que se passerait-il si l'on se trompait ? Le gâteau serait un peu trop mou ou un peu trop sucré, mais, pour un algorithme, ce serait très probablement la catastrophe absolue ! Voici un exemple : il s'agit de calculer la factorielle d'un nombre *n* (ici, comme souvent dans les notations informatiques, les instructions se placent entre des accolades et se terminent par un point-virgule, l'astérisque représentant la multiplication) :

```
factorielle (entier n) {
    si (n = 0) {
        renvoyer 1 ;
    } sinon {
        renvoyer n * factorielle (n - 1) ;
}
```

Cette formule sert bien à calculer la factorielle, puisqu'elle implémente la définition par récurrence de la factorielle, telle qu'elle est présentée dans nos manuels scolaires, à savoir : factorielle (0) = 1 et factorielle (n) = n * factorielle (n - 1). Si n = 0, le résultat renvoyé est donc 1, sinon le résultat renvoyé est calculé selon la deuxième formule. Avec les ingrédients précédents, on peut proposer une autre implémentation équivalente, c'est-à-dire noter *f* la variable intermédiaire qui va prendre les valeurs du calcul, et écrire le code suivant :

```
entier f = 1 ;
 tant que (n n'est pas nul)  {
    f = f * n ;
    n = n - 1 ;
 }
```

Cette méthode permet de calculer : factorielle (n) = n * (n - 1) * etc. * 1, donc exactement la factorielle que nous connaissons, en s'arrêtant lorsque $n = 0$, ce qui semble parfait… tant que n est positif. S'il est négatif, l'un ou l'autre code va se mettre à calculer factorielle (n) pour les valeurs suivantes, en partant de $n = -1$: $n = -1, -2, -3, -4$… sans s'arrêter ! C'est alors au hasard de décider de ce qui va arriver : ordinateur bloqué, calcul qui dure jusqu'à ce que les batteries soient vides, etc. Bienvenue devant ce qui s'appelle un « *bug* ». En effet, la fonction factorielle n'est que partiellement définie : pour $n \geqslant 0$, tout est bien spécifié, mais pour $n < 0$, rien n'est dit. L'élève reçoit un 0 à sa copie s'il commet l'étourderie ; le programme informatique, lui, boucle à l'infini. Dans ce cas précis, il y a plusieurs parades possibles : soit vérifier que $n \geqslant 0$ avant de lancer le calcul soit modifier le code en écrivant factorielle (entier n), si $n \leqslant 0$ alors renvoyer 1, sinon renvoyer n * factorielle (n - 1), qui renvoie la valeur 1 pour tous les entiers n négatifs ou nuls. En un sens, il s'agit d'étendre la fonction factorielle aux nombres négatifs.

Au-delà de cet exemple, comme nous l'explique Gérard Berry dans son cours au Collège de France sur les sciences numériques, et comme l'enseigne Gilles Dowek dans son cours d'informatique pour des professeurs de mathématiques de lycée, il existe des méthodes formelles qui adoptent une méthodologie plus proche des mathématiques (www.inria.fr/rocquencourt/ressources/multimedia/formation-informatique-et-objets-numerique). Elles permettent ainsi d'effectuer de nombreux calculs sur ces programmes, devenus à leur tour l'objet de manipulations informatiques.

Une technique fondamentale, nommée « interprétation abstraite » – développée à l'École normale supérieure par l'équipe de Patrick Cousot –, permet de détecter automatiquement des *bugs*, comme celui d'Ariane, identifié par Gilles Kahn : lorsque le premier vol d'Ariane 5, en 1996, a explosé et détruit pour 500 millions de dollars de satellites, la faute était due à un morceau de programme qui convertissait une valeur. Créé pour Ariane 4, il a été réutilisé tel quel. Or la valeur à convertir, beaucoup plus grande dans le cas d'Ariane 5, a donné un résultat aberrant et des ordres de braquage maximaux. L'interprétation abstraite évite surtout que de tels *bugs* ne se reproduisent ; elle a été appliquée avec succès sur les logiciels de vol de l'Airbus A380. D'autres techniques offrent la possibilité d'aller jusqu'à la vérification mathématique de propriétés dites de sûreté (« L'ascenseur ne pourra jamais voyager la porte ouverte. ») ou de vivacité (« L'ascenseur finira par prendre tous les passagers qui l'attendent. »). De telles vérifications formelles se font en calculant symboliquement le gigantesque ensemble des exécutions possibles, sans les effectuer explicitement.

Les méthodes formelles que nous allons évoquer permettent ainsi de diminuer sensiblement le nombre de *bugs* dès la conception, puis de trouver des *bugs* particulièrement sournois, hors de portée des tests manuels ou aléatoires ; même si elles ne suffisent pas à elles seules à assurer la sécurité requise pour tous les logiciels du commerce, à cause de leur taille et de leur nombre grandissants, puisque le problème est identifié comme étant indécidable mécaniquement. Il s'agit donc d'exécuter des algorithmes, de vérification et de preuve, sur les données que sont les algorithmes à vérifier ou à prouver. De ce fait, il est nécessaire que ces derniers puissent être exprimés sous une forme interprétable sans ambiguïté. Comment allons-nous donc définir formellement nos ingrédients ?

E De l'algorithme au programme… une notation rigoureuse ?

Les différents langages de programmation sont organisés autour d'un petit nombre de fonctionnalités présentes dans de nombreux langages, relativement stables depuis des décennies, et que l'on peut décrire simplement avec les outils adéquats (affectation, séquence, test, boucle, fonction, récursivité, enregistrement, cellule, module, objet, etc.). Le *noyau* de la plupart des langages est justement constitué des cinq ingrédients que nous détaillons pp. 41-42. Pour chacune de ces fonctionnalités, nous allons décrire :

- sa syntaxe : *la manière dont cette instruction s'écrit.* Ne perdons pas de vue que la question importante est de savoir de quoi est constituée cette instruction, quels sont ses paramètres, etc. Le fait de savoir comment cela s'écrit dans le langage particulier que l'on utilise est accessoire. De même, lorsqu'on apprend à conduire une voiture, l'important est de connaître la signification des mots « frein », « levier de vitesse », etc., et de pouvoir s'en servir. Que ces commandes aient tel ou tel aspect sur une Twingo ou une Mercedes est accessoire ;
- sa sémantique : *ce qui se passe quand on l'exécute.* Détaillons ce point.

Une instruction – ou un groupe d'instructions – a pour but de transformer un *état*, c'est-à-dire la *valeur* de chacune de ses variables. L'état indique donc la valeur de chaque variable à un instant donné. Par exemple, si le programme a trois variables x, y, z qui représentent trois « boîtes » dans lesquelles nous pouvons ranger des valeurs, une instruction va modifier les valeurs de ces boîtes lors de son exécution. De modification en modification, un programme fera passer l'état de sa valeur initiale à sa valeur finale.

Un programme a bien sûr des entrées et des sorties : les sorties sont des boîtes dont les valeurs sont visibles d'un dispositif externe ; les entrées sont des variables dont les valeurs ont été prédéfinies par l'utilisateur ou un dispositif externe. Pour formaliser la sémantique d'une instruction, les informaticiens et les logiciens utilisent habituellement des notations mathématiques, car elles ont l'avantage d'être concises et sans ambiguïté. Nous vous les présentons ci-dessous :

- l'état est représenté par une liste d'équations : [$x = 12$, y = « Hi », $z = 0$] qui donnent la valeur des variables ;
- pour une instruction p, la construction Σ (p, s) = s' signifie qu'elle transforme un état s en un état s'.

Nous sommes maintenant en mesure de détailler chaque instruction.

F Affectation d'une valeur à une variable

La première action est de donner des valeurs aux variables : c'est l'affectation.

MANIÈRE D'ÉCRIRE L'INSTRUCTION

L'affectation est constituée d'une variable x et d'une valeur v. Selon les langages, elle s'écrit $x = v$, $x := v$, $x = v$; etc. Par exemple : $x = 15$. Informellement : la variable x prend comme valeur v. Sa valeur précédente est effacée ; elle gardera toujours la valeur v tant qu'elle ne sera pas changée par une autre affectation de la variable x. L'affectation est donc une « action » : celle d'affecter à la variable x la valeur v. Il faut aussi bien distinguer l'affectation de l'égalité (souvent notée par un double signe ==). Écrire $x == 15$ revient à poser la question : « Est-ce que x est égal à 15 ? » $x == 15$ est vrai si x est égal à 15, faux sinon. Écrire $x = 15$ revient à donner l'ordre que « désormais x sera égal à 15 » !

SENS DE L'INSTRUCTION

Il est facile de comprendre ce que fait cette instruction : elle change l'état en lui ajoutant le fait que désormais $x = v$. Avec les notations mathématiques, nous pouvons le noter ainsi : $\Sigma\ (x = v, s) = s + [x = v]$. Cela signifie que l'on ajoute à l'état la nouvelle équation qui définit la valeur de la variable affectée.

G Fonctions et expressions

La valeur que nous donnons à une variable n'est pas forcément une constante, car nous voulons aussi évaluer des expressions : écrire, par exemple, $x = 12 + z / 2$ pour calculer x à partir de z.

MANIÈRE D'ÉCRIRE UNE EXPRESSION

Une expression numérique (c'est-à-dire qui porte sur des nombres) se construit comme sur une calculette, avec des additions (par exemple : 2 + 3), des soustractions, des multiplications (en utilisant le symbole *) et des divisions (en utilisant le symbole /). Il existe d'autres symboles algébriques que nous n'utiliserons pas ici : il suffit de retenir qu'ils s'écrivent comme pour une calculette. Par exemple, si nous écrivons $2 + (x / 3)$ ou $(2 + x) / 3$, ce ne sont pas les mêmes expressions, puisque le calcul se fait différemment selon les parenthèses.

Une expression logique (c'est-à-dire qui renvoie la valeur « vrai » ou « faux ») se construit avec :
- des comparaisons pour tester si deux valeurs sont égales ou non, si une valeur numérique est plus petite qu'une autre, etc. ;
- des opérateurs logiques comme « et » (noté &&), « ou » (notée ||), ainsi que la négation (notée !).

On peut aussi manipuler des chaînes de caractères, notées entre guillemets, et les concaténer (par exemple, « pa » et « py » se concatènent en écrivant « pa » + « py », ce qui donne « papy ») ou les comparer, en utilisant la fonction equal ( ) : l'expression equal (« pa », « py ») renverra la valeur « faux », puisque les chaînes de caractères « pa » et « py » ne sont pas égales, tandis que equal (« pa », « pa ») renverra la valeur « vrai », puisque ce sont les mêmes valeurs.

D'autres fonctions que les opérations numériques ou logiques peuvent être utilisées. Elles s'écrivent en utilisant des parenthèses : $f(x)$ correspondant à une fonction de nom f, qui prend comme argument en entrée x et va renvoyer un résultat. Par exemple : $f(x) = x * x$ calcule $x * x$, c'est-à-dire le carré de x. Les fonctions peuvent avoir plusieurs arguments, comme $g(x, y, z)$, qui en a trois. Il est d'ailleurs possible de définir soi-même de nouvelles fonctions.

SENS D'UNE EXPRESSION

Pour une expression e (par exemple : $12 + 6 / 2$), nous utilisons une nouvelle notation $\Theta\ (e, s) = r$ pour signifier que la valeur du résultat de son évaluation est r (ici, 15). Il faut donc bien distinguer une expression ($12 + 6 / 2$), une valeur (15) et une affectation ($x = 12 + 6 / 2$).

La valeur d'une expression dépend de l'état. Par exemple, l'expression $12 + z / 2$ dépend de la valeur de z. Ce qui donne, avec les notations mathématiques : $\Theta\ (12 + z / 2, [z = 0]) = 12$ ou $\Theta\ (12 + z / 2, [z = 6]) = 15$. Nous pouvons alors définir l'affectation d'une expression à une variable, avec ces notations : $\Sigma\ (x = e, s) = s + [x = \Theta\ (e, s)]$. Cela signifie que la variable prend comme valeur le résultat de l'évaluation de l'expression.

L'instruction « tourner les pages une à une jusqu'à trouver le pays » s'écrit comme suit :

```
trouver (pays) {
    page = début ;
    tant que page <= fin
        si dictionnaire (page) = pays alors renvoyer page
        page = page +1
    renvoyer « pas trouvé »
}
```

c'est-à-dire énumérer les pages du début à la fin du dictionnaire, ce qui est implémenté à travers la boucle :

```
    page = début ;
    tant que page < = fin
    .../...
    page = page +1
```

dont on sort en renvoyant la page du dictionnaire correspondant au pays si elle est trouvée, comme le spécifie ce test de comparaison :

```
    si
        dictionnaire (page) = pays
    alors
        renvoyer page
```

et dont on sort, à la fin, en signalant ne pas avoir trouvé la page.

Voilà un algorithme qui fonctionne parfaitement. Il coûtera une étape de calcul (ici, un test de comparaison) si le pays est au début du dictionnaire… et 195 étapes s'il se trouve à la fin. Dans le cas présent, le dictionnaire contient N items (N = 195), répartis de manière uniforme ; on voit donc que, de 1 à N étapes de calcul, il y en aura environ C = N / 2 à prévoir. Nous aurions pu énumérer les pages dans un autre ordre (de la fin au début) ou toutes les pages paires, puis les impaires, mais cela n'aurait rien changé : pourvu que nous énumérions toutes les pages, nous sommes certains de trouver le pays recherché, et ce sera alors toujours avec un coût de l'ordre de N étapes de calcul.

PEUT-ON ÊTRE PLUS EFFICACE ?

Si, dans le dictionnaire, les pays n'étaient pas classés par ordre alphabétique mais dans le désordre, la réponse serait négative : nous serions, de fait, obligés de tourner toutes les pages pour être certains de trouver le pays recherché. Mais voilà que le dictionnaire est ordonné de A à Z ; il s'agit là d'une propriété que nous pourrions exploiter. En effet, si nous ouvrons le dictionnaire au milieu, sur la page de la France, par exemple, tandis que nous cherchons l'Albanie, nous voyons que ce n'est pas la bonne page mais nous apprenons quelque chose de plus : puisque l'Albanie est avant la France, sa page est donc forcément située dans la moitié gauche du dictionnaire ; nous n'avons donc plus à chercher dans toutes les pages de droite, mais uniquement dans celles de gauche. Nous avons en une opération réduit notre espace de recherche de moitié. D'environ 200 pages, il ne nous restera plus que 100 pages à explorer. Puis, en reprenant le même procédé, 50 pages, 25 pages, 12 ou 13 pages, 6 ou 7 pages, 3 ou 4 pages, 1 ou 2 pages, et le pays sera trouvé.

Si nous calculons, nous voyons qu'il y a eu uniquement C = 7 ou 8 étapes de calcul ! Ce processus, qui consiste, à chaque étape, à couper en deux parties égales l'espace de recherche, est appelé « dichotomie » (« couper en deux », en grec). Intuitivement, on se rend compte que c'est bien en coupant en deux parties égales que nous sommes sûrs d'avoir des deux côtés un espace de recherche minimal à l'étape suivante (en parts inégales, la malchance pourrait nous conduire à rechercher dans un espace plus grand). Il faut comprendre que le gain est immense. Nous donnons dans le tableau ci-dessous le nombre C d'étapes de calcul en fonction de quelques nombres N d'items :

N = 2C	1	2	4	8	16	…	256	…	1024	…	un million	…	un milliard
C = log2(N)	0	1	2	3	4	…	8	…	10	…	20	…	30

Le fait de diviser en deux, de manière itérative, permet notamment de rechercher en un maximum de 26 étapes le nom d'une personne dans l'annuaire des 60 millions de Français. Il existe plusieurs manières de programmer ce mécanisme. En voici une où nous restreignons l'espace de recherche entre deux valeurs, « min » et « max » :

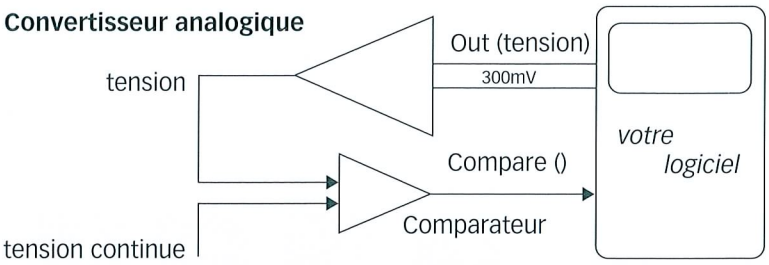
```
trouver (pays) {
    min = début ; max = fin ;
    répéter
        milieu = (min + max) / 2 ;
        si dictionnaire (milieu) = pays alors renvoyer milieu
        si min = max alors renvoyer « pas trouvé »
        si dictionnaire (milieu) < pays alors min = milieu
        si dictionnaire (milieu) > pays alors max = milieu
}
```

où l'on remarque que la quantité « max-min » vaut « fin-début » au démarrage, puis se divise par deux à chaque étape et devient plus petite que 1, donc égale à 0, puisque ce sont des nombres entiers. L'algorithme va donc s'arrêter au bout d'un nombre fini, logarithmique, d'étapes. Terminer en tournant la page si elle est trouvée ou sur « pas trouvé » si l'intervalle de recherche est de longueur nulle sans avoir trouvé la page recherchée.

Posons-nous maintenant une autre question : dans quelle mesure ce mécanisme algorithmique est-il générique ? Regardons un tout autre exemple pour s'en convaincre.

CONVERTIR UNE TENSION ÉLECTRIQUE EN VALEUR NUMÉRIQUE

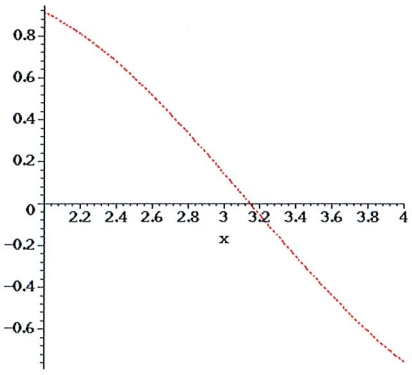
Pour « deviner » la valeur d'une tension électrique continue, un ordinateur numérique doit, en général, comparer cette valeur à une valeur de référence qu'il va produire en sortie pour, de proche en proche, cerner cette valeur, comme le montre le diagramme ci-dessous :



Là encore, la façon de fonctionner de ces convertisseurs à approximations successives est de procéder de manière dichotomique, en divisant l'espace de recherche de deux en deux. Cela permet d'atteindre très rapidement les précisions requises, le millième en dix étapes, le millionième en vingt, comme nous l'avons vu précédemment.

TROUVER LE ZÉRO D'UNE FONCTION MONOTONE DANS UN INTERVALLE

Tous ces problèmes sont reliés au problème mathématique suivant : résoudre une équation de la forme $f(x) = 0$, $\min < x < x \max$ où $f()$ est une fonction continue monotone dans l'intervalle [min, max]. Elle a donc une solution, puisqu'il y a bijection vers un intervalle réel. Si elle change de signe dans cet intervalle $f(\min)\, f(\max) < 0$, elle a une solution unique. C'est le cas de la fonction $\sin(x)$ dans l'intervalle [2, 4], où elle s'annule en π, comme le montre la figure ci-dessous :



Là encore, le même mécanisme de dichotomie permet de résoudre le problème, comme dans l'algorithme suivant :

```
zéro (f) {
    min = début ; max = fin ;
    si f (min) f (max) > = 0, alors renvoyer « pas de solution »
    répéter
        milieu = (min + max) / 2 ;
        si |max - min| < Σ, alors renvoyer milieu
        si f (max) f (milieu) < 0, alors min = milieu
        sinon, si f (min) f (milieu) < 0, alors max = milieu
        sinon renvoyer « pas de solution »
}
```

qui ne diffère du précédent que par la façon de détecter le fait qu'il n'y ait pas de solution et par celle d'effectuer les tests. Il s'agit donc bien d'un principe algorithmique, qui fonctionne à la fois pour des opérations numériques et des opérations symboliques (recherche de mots), comme Ada Lovelace le savait un siècle avant que ne fonctionne le premier ordinateur. Et c'est l'un des mécanismes algorithmiques les plus utilisés.

> PAR MARIE-CHRISTINE CUVELIER, PROFESSEURE DOCUMENTALISTE

Organismes institutionnels

Interstices, découvrir la recherche en informatique

HTTP://INTERSTICES.INFO

Interstices est un site de culture scientifique créé par des chercheurs pour rendre accessibles à un large public les sciences et technologies de l'information et de la communication. Lancé à l'initiative de l'Inria, *Interstices* se développe en partenariat avec le CNRS et les universités. Le site, très agréable à visiter, propose des informations riches et variées sous forme de petits dossiers thématiques qui traitent du langage informatique, des algorithmes, de la cryptographie, des virus, des réseaux, etc. Les rubriques « Découvrir » et « Approfondir » contiennent des articles pour comprendre les bases de la recherche en informatique ou en savoir plus sur des notions fondamentales : « Itinéraires » propose des portraits de chercheurs axés sur leurs parcours scientifiques ; « Débattre » aborde des questions de société liées aux évolutions scientifiques ; et on trouvera dans la rubrique « Ludique » des quiz, énigmes et jeux interactifs capables de distraire intelligemment l'internaute. ●

L'Institut national de recherche en informatique et en automatique (Inria)

WWW.INRIA.FR
WWW.YOUTUBE.COM/USER/INRIACHANNNEL

Le site de l'Inria regroupe de très nombreux documents et liens en rapport avec la recherche en informatique. L'amateur éclairé trouvera dans la rubrique « Ressources scientifiques » un accès à une bibliothèque de publications scientifiques particulièrement étoffée dans tous les domaines liés à l'informatique (voir « Archive ouverte HAL-INRIA »), ainsi que diverses ressources électroniques de l'institut (conférences, ouvrages, périodiques, etc.) disponibles à partir de « Ressources documentaires ». Le simple profane, quant à lui, visitera plutôt la rubrique « Science pour tous », qui lui permettra non seulement d'accéder à des ressources multimédias variées, mais aussi, en utilisant « Fuscia, bureau d'accueil », de poser aux chercheurs et experts de l'Inria des questions qui obtiendront des réponses détaillées, accompagnées de liens renvoyant aux ressources du site ou à des ouvrages de référence conseillés.
On trouvera également sur Youtube la chaîne « Inria sciences numériques », qui présente de nombreuses vidéos sur l'actualité de la recherche et de l'innovation informatiques. ●

Histoire de l'informatique

Les archives de Radio-Canada

HTTP://ARCHIVES.RADIO-CANADA.CA

Il est difficile d'imaginer aujourd'hui à quoi pouvait bien ressembler en 1955 un calculateur électronique, comment on utilisait en 1960 une carte perforée ou comment se déroulait en 1971 une classe d'informatique. Heureusement, grâce à ces 17 clips radio et 23 clips télé disponibles dans les archives de Radio-Canada, vous pourrez revivre en images et en son des scènes qui illustrent ces anciens moments de l'histoire de l'informatique, de 1955 à 2007. Les films et émissions sont complétés par des commentaires regroupés en trois rubriques : « Contexte », « Le saviez-vous ? » et « Infos », qui apportent les éléments utiles à l'analyse du document. ●

Musée de l'Informatique

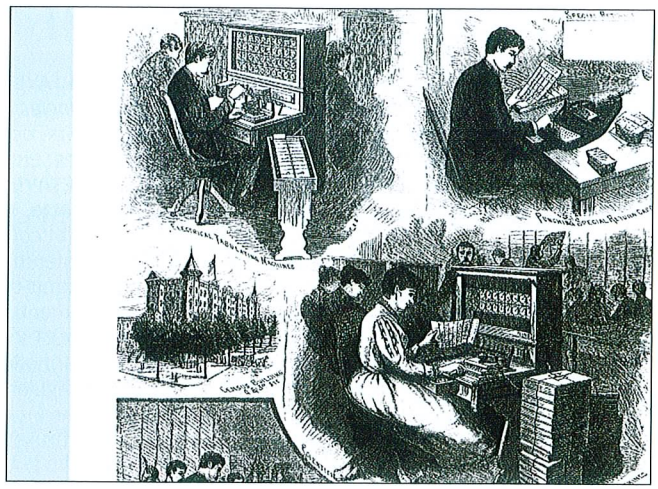
WWW.MUSEEINFORMATIQUE.FR

Ouvert en avril 2008 au sommet de la Grande Arche de la Défense, ce musée contient des collections permanentes et propose des expositions temporaires variées. À défaut d'une visite, l'internaute pourra consulter le « Blog du musée » qui permet de suivre l'actualité de l'informatique et présente des fiches sur de vieux calculateurs (IBM 1440, Univac 1100, ordinateurs français, etc.). Par ailleurs, l'intéressante rubrique « La télé du musée » offre de nombreux petits films présentant reportages, entretiens, chroniques et interviews sur le monde de l'informatique. Enfin, une photothèque présente une belle collection de 136 photographies d'objets exposés au musée, qui passionnera sans doute les pionniers de l'informatique. ●

Site personnel de François Guillier

WWW.HISTOIRE-INFORMATIQUE.ORG

Le site de cet informaticien rappelle les grandes dates et les principales étapes de l'histoire de l'informatique, retraçant l'évolution des théories, mais aussi celles du matériel, des langages et des supports de stockage. Une galerie de portraits recense les grands acteurs de cette épopée, un lien renvoyant généralement à une biographie plus détaillée. Enfin, la rubrique « Le musée » permet de « faire le voyage en images », en découvrant les principaux appareils ayant marqué l'histoire de l'informatique, du boulier chinois à l'iMac, en passant par la Pascaline, la machine de Charles Babbage et le premier IBM. La rubrique de liens, très riche, donne des pistes aux curieux pour aller plus loin. •

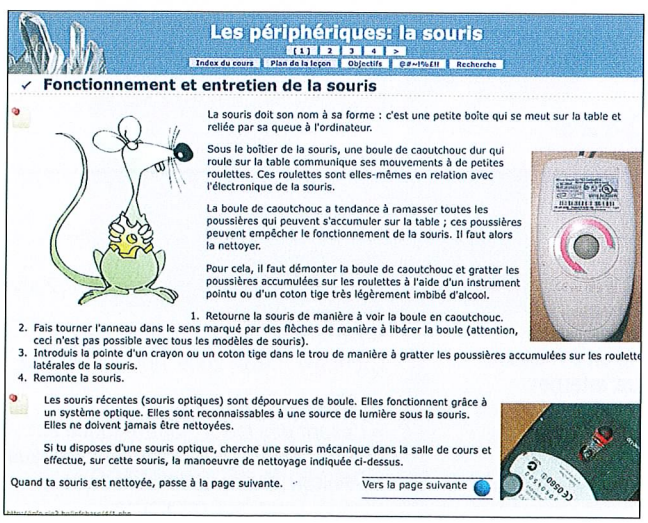


Sites pédagogiques

L'informatique, comment ça marche ?

HTTP://INFO.SIO2.BE

Un peu d'informatique et puis d'autres choses est un site simple et agréable à utiliser. Son créateur, Yves Mairesse, professeur belge d'informatique et de sciences, y propose une large palette de cours sur des thèmes variés autour de l'ordinateur. Sont disponibles des cours généraux (rubriques « Informatique de base » et « Informatique pour scientifiques »), qui traitent aussi bien du matériel que du logiciel. Des cours plus spécialisés permettent de découvrir la programmation en langage Python, mais aussi le Delphi Pascal ou le dessin vectoriel. Ces documents ont surtout une orientation pratique (mais néanmoins ludique) qui les rend bien adaptés à des élèves de collège ou de lycée, d'autant que beaucoup de leçons se basent sur l'emploi de logiciels libres. Centre sur les savoir-faire et décortiquant le fonctionnement de logiciels courants, ce site sera utile aux débutants en informatique. •



Introduction à l'informatique

WWW.GRAPPA.UNIV-LILLE3.FR/POLYS/INTRO-NFO/INDEX.HTM

Cette Introduction consultable en ligne (ou téléchargeable au format PDF), claire et accessible même à des non-informaticiens, a été conçue, à l'origine, à partir d'un cours donné en licence 1 non-informatique par Isabelle Tellier. Au menu : les principes de codage des données utilisées par les ordinateurs, les fondements théoriques de l'informatique, l'architecture de von Neumann, une introduction aux réseaux, etc. Ce document comporte également des exercices corrigés. Dans un domaine plus pointu, d'autres ressources sont aussi disponibles : par exemple, une « Introduction au TALN et à l'ingénierie linguistique » ou un ensemble de travaux dirigés pour l'apprentissage du langage Prolog. •

Informatique et enseignement

WWW.EPI.ASSO.FR/ACCUEIL.HTM

Depuis sa fondation en 1971, l'association EPI (Enseignement public et informatique), pionnière et militante, « veut faire de l'informatique, et des technologies de l'information et de la communication en général, un facteur de progrès et un instrument de démocratisation ». Son site, très régulièrement mis à jour, est une mine de ressources pour l'enseignant qui s'intéresse aux TICE.

Quant à l'autre grande rubrique du site : « Enseignement de l'informatique », elle témoigne de l'un des objectifs premiers de l'association : la formation des maîtres. Ceux-ci y trouveront des démarches et pratiques pédagogiques très nombreuses, appliquées à tous les domaines : matériel, réseaux, données, programmation, langages, didactiques, etc. Enfin, ceux qui s'intéressent aux sciences de l'éducation consulteront, dans la rubrique « Historique », un dossier intitulé « Historique de l'informatique dans le système éducatif français », qui rassemble tous les documents sur ce sujet publiés par l'association depuis sa création. •

1 OUVRAGES GÉNÉRAUX

- **BERRY Gérard.** *Pourquoi et comment le monde devient numérique.* Paris : Fayard, 2008. 78 p. (coll. Leçons inaugurales du Collège de France).
- **DENOYELLE Philippe, GEYNET Maurice, JACOB Michel.** *L'Informatique.* Lyon : EMCC, 2008. 120 p. (coll. Des objets qui racontent l'histoire).
- **GRIFFE Maurice.** *De la carte perforée à l'ordinateur et Internet.* Le Cannet : Tableaux synoptiques de l'histoire, 2006. 40 p.
- **GUÉDON Jean-Claude.** *Internet : le monde en réseau.* Paris : Gallimard, 1996. 128 p. (coll. Découvertes).
- **KALTENBACH Laure, ALEXANDRE Joux.** *Les Nouvelles Frontières du Net.* Paris : First Editions, 2010. 208 p.
- **LAPOINTE Pascal, DROUIN Josée Nadia.** *Science, on blogue !* Sainte-Foy : Multimondes, 2007. 295 p.
- **LEAVITT David.** *Alan Turing : l'homme qui inventa l'informatique.* Paris : Dunod, 2007. 273 p. (coll. Quai des sciences).

2 MATHÉMATIQUES

- **DEHORNOY Patrick.** *Complexité et décidabilité.* Bâle : Birkhauser, 2000 (coll. Mathématiques et applications).
- **DELAHAYE Jean-Paul.** *Complexités : aux limites des mathématiques et de l'informatique.* Paris : Belin/ Pour la science, 2006. 256 p. (coll. Bibliothèque scientifique).
- **DOISY Philippe.** *À la racine des nombres : une histoire du calcul numérique des origines à nos jours.* Paris : Ellipses, 2006. 496 p.
- **DOWECK Gilles.** *Métamorphoses du calcul : une étonnante histoire de mathématiques.* Paris : Le Pommier, 2007. 223 p. (coll. Essais).
- **RECHENMANN François, JONG Hidde de.** « Le vivant en équations », in *La Recherche,* n° 338, février 2005.

3 PHYSIQUE

- **DALIBARD Jean.** « La condensation de Bose-Einstein en phase gazeuse », in *Images de la physique,* 2001. www.phys.ens.fr/~dalibard.
- **DANAILA Ionut.** « Tourbillons en fluides et superfluides : une exploration numérique », mémoire de HDR, 2008. www.ann.jussieu.fr/~danaila.
- **VARENNE Franck.** *Du modèle à la simulation informatique.* Paris : Vrin, 2007. 249 p. (coll. Mathesis).
- « Le condensat de Bose-Einstein : la matière qui vient du froid », in *Science & vie,* n° 228, septembre 2004.

4 DROIT

- **LE MÉTAYER Daniel.** *Les Technologies de l'information au service des droits : opportunités, défis, limites.* Bruxelles : Émile Bruylant, 2010.
- **LE MÉTAYER Daniel, ROUVROY Antoinette.** « STIC et droit, défis, conflits et complémentarités », http://interstices.info/jcms/c_34521/stic-et-droit-defis-conflits-et-complementarites.
- **POULLET Yves.** « Technologies de l'information et de la communication et "co-régulation" : une nouvelle approche ? », in *Liber Amicorum Michel Coipel.* Bruxelles : Kluwer, 2004.

5 OUVRAGES POUR LA JEUNESSE

Documentaires

- **GIRARDET Sylvie, PUIG ROSADO Fernando.** *L'Écran et la Souris.* Paris : Hatier, 2006. 63 p. (coll. Citoyens en herbe). À partir de 5 ans. Thèmes : multimédia, internet, mail, blog, téléchargement.
- **ROQUEMAUREL Astrid de, VAUFREY Delphine.** *Mais non, je blogue ! : le petit guide des jeunes blogueurs pour bloguer vite et bien !* Toulouse : Milan Jeunesse, 2008. 80 p. À partir de 10 ans. Thèmes : multimédia, internet, informatique, ordinateur.
- **TORVALDS Linus, DIAMOND David.** *Linux, c'est gratuit ! Mais aidez-moi à l'installer.* Paris : L'École des loisirs, 2010. 288 p. (coll. Médium documents). À partir de 14 ans. Thèmes : biographie, Linus Torvalds, système Linux.
- **VIRIEUX Françoise, CHEBRET Sébastien.** *Internet : quel drôle de réseau !* Paris : Le Pommier, 2006. 63 p. (coll. Les petites pommes du savoir). À partir de 9 ans. Thèmes : ordinateur, site internet, multimédia.
- **ZEITOUN Charline, CHENOT Patrick.** *L'Informatique.* Paris : Mango Jeunesse, 2008. 23 p. (coll. Kézako ?). À partir de 7 ans. Thèmes : internet, informatique, langage binaire, microprocesseur.

Romans

- **CHARIF Hicham.** *Les Virus de l'ombre.* Saint-Hippolyte-du-Fort : Le Navire en pleine ville, 2007. 301 p. À partir de 14 ans. Thèmes : amour, informatique, intelligence artificielle, virus.
- **FÉRET-FLEURY Christine.** *Serial Blogger.* Paris : Hachette Jeunesse, 2009.158 p. (coll. Le livre de poche Jeunesse). À partir de 13 ans. Thèmes : disparition, informatique, internet.
- **LAMBERT Christophe.** *Console à haut risque.* Paris : Hachette Jeunesse, 2004. 186 p. À partir de 10 ans. Thèmes : enquête policière, jeux vidéo, informatique.

6 AU CATALOGUE DU SCÉRÉN

www.sceren.com

Imprimés

- *Introduction à la théorie des graphes.* Lyon : CRDP, 2005. 217 p. Réf. 690B3393.
- *Mathématiques, informatique en classe de première L.* Lille : CRDP, 2006. 104 p. (coll. Méthodes en pratique). Réf. 590M0008.
- *Maths en graphes : terminale.* Grenoble : CRDP, 2006. 270 p. Réf. 380GRA01.
- « Le numérique et les sciences du vivant », *DocSciences,* n° 8. Versailles : CRDP, 2009. Réf. 7800S008.
- « Les clés de la révolution numérique », *DocSciences,* n° 5. Versailles : CRDP, 2008. Réf. 7800S005.

TDC

- « Les nombres », n° 869, 1er février 2004.
- « Les créatures artificielles », n° 917, 1er juin 2006.

Multimédia

- *50 activités pour utiliser l'internet à l'école et au collège.* Toulouse : CRDP, 2008 (coll. Progrès). Livre + cédérom. Réf. 31081A52.
- *Éléments pour une pédagogie des écrans.* Toulouse : CRDP, 2003 (coll. Ressources formation vidéo/ Multimédia). Réf. 31000F02.
- *Informatique et Internet à l'école.* Grenoble : CRDP, 2007 (coll. Objectif multimédia). Livre + cédérom. Réf 380B5011.
- *Informatique et Internet au collège et au lycée.* Grenoble : CRDP, 2007 (coll. Objectif Multimédia). Livre + cédérom. Réf. 380B5012.

Cédéroms

- *Informatique et Internet : apprendre et s'évaluer.* Grenoble : CRDP, 2008. Réf. 380M042E.
- *Informatique et Internet : classeur électronique.* Grenoble : CRDP, 2003 (coll. Banques pédagogiques). Réf. 380MCE01.
- *Labo MPI (mesures physiques et informatique).* Paris : CRDP / Coulommiers : Micrelec, 2002. Réf. 750LOG01.
- *Pratiquer l'algorithmique.* Grenoble : CRDP, 2003 (coll. Progrès). Réf. 380M015E.
- *TICE à la carte.* Versailles : CRDP, 2002. Réf. 7800MI22.
- *WéM, Web éditeur mathématiques.* Grenoble : CRDP, 2005. Réf. 380M024M.

Vidéos

- *L'esprit des lois 5 : informatique et libertés. Service suspendu.* Chasseneuil-du-Poitou : CNDP, 2000 (coll. Galilée). Vidéo + livret. Réf. 002K3017.

+ de ressources

Derrière les objets numériques du quotidien se déploie une activité scientifique dynamique : l'informatique. Cette discipline récente entretient des liens étroits avec les mathématiques, la physique ou la biologie. Les spécialistes de l'**Inria** vous donnent des clés pour mieux comprendre et expliquer ces nouveaux champs de la connaissance à vos élèves.

+ d'échanges

Tout au long de l'année, les chercheurs des huit centres de l'**Inria** vont à la rencontre des enseignants et des élèves : visites de laboratoires, ateliers éducatifs, expériences interactives ou présentations en classe. Autant d'occasions de faire partager leur passion et de sensibiliser les élèves à ces nouvelles disciplines.

découvrir, approfondir, interagir...

L'**Inria** et ses partenaires mettent à la disposition des enseignants et élèves une grande variété de contenus culturels et d'informations sur les métiers de l'informatique.

Pour découvrir

Des revues réalisées avec Bayard Presse

Comment simuler une tempête ? une opération chirurgicale ?
À quoi ressemblent les mondes virtuels ?
Pour faire découvrir ou redécouvrir aux élèves différentes facettes de l'informatique.

La chaîne YouTube Inria

Tout ce que vous avez toujours voulu savoir sur le numérique sans avoir le temps de le lire : c'est tout en vidéo sur la chaine InriaChannel.

Pour approfondir

Deux numéros spéciaux avec le CRDP de Versailles

Des spécialistes décryptent le code binaire, les algorithmes, la programmation, la modélisation du vivant, la bioinformatique...

L'informatique à portée de clic

Le site de vulgarisation interstices.info propose près de 300 articles, podcasts, images, etc. pour les curieux de science.

Pour interagir

fuscia.info

+ une compilation de ressources pédagogiques (supports de cours, textes, sites, vidéos ...)

L'INRIA propose, à travers son initiative fuscia, un accueil en ligne individualisé.
Vous pouvez par exemple échanger directement avec un expert autour des TPE ou des TIPE.

Pour tout renseignement, demande de documentation, organisation de visites, de conférences ou ateliers avec vos classes,
contacter la direction de la communication au 01 39 63 56 84

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE
INRIA