numéro 1

1024

BULLETIN

de la société informatique de France

septembre *2013*

COMITÉ DE RÉDACTION

Société informatique de France
Institut Henri Poincaré, 11 rue Pierre et Marie Curie, 75231 Paris Cedex 05

Directeur de la publication : Colin de la Higuera
ISSN : en cours d'attribution.

SOMMAIRE DU Nº 1

Le mot du Président

Colin de la Higuera [1]

Il existe des moments de réel plaisir... de ces moments où on a la sensation qu'un projet collectif auquel on participe est réellement utile. J'ai le privilège de vivre un de ces moments en accompagnant aujourd'hui la sortie de ce premier numéro de 1024. Ce numéro est le résultat d'une énergie, et il faut saluer tout de suite celle de son rédacteur en chef et de son équipe ; c'est aussi le résultat d'une logique : la naissance de la Société informatique de France.

Ce premier numéro de 1024 vient en effet compléter une première année d'existence de la SIF particulièrement remplie. Pour mémoire, la SIF a été créée, en mai 2012, pour devenir la société savante française en Informatique. Le besoin d'une telle société savante avait été mis en évidence : les activités menées et les événements qui se sont produits depuis plus d'un an confirment s'il le fallait que ce besoin existait, et que la SIF arrivait dans un contexte où un tel interlocuteur avait toute sa place.

Que ce soit sur des sujets touchant à l'enseignement de l'informatique, à la place de l'informatique française dans le concert international, aux liens particulièrement importants entre la recherche et l'innovation, à l'image de la discipline elle-même, la SIF a été sollicitée, a discuté, argué, étudié, écrit...

Les informations et débats concernant l'enseignement de l'informatique s'accélèrent : là où il y a un an il semblait déraisonnable d'avoir d'autre ambition que de se réjouir de l'arrivée de l'informatique en tant que spécialité en Terminale S, il n'y a aujourd'hui plus de tabous à concrètement accompagner l'arrivée de l'informatique dans toutes les sections de lycée, à réfléchir à sa mise en place au collège et à se demander comment les professeurs des écoles vont aborder l'introduction de son enseignement dans le primaire !

1. Président de la Société informatique de France, Professeur à l'Université de Nantes, E-mail : cdlh@univ-nantes.fr.

Le chemin parcouru est le résultat du travail de nombreuses personnes qui essayent de convaincre (pour certaines depuis des dizaines d'années) notre société, les pouvoirs publics, à tous les niveaux, que le monde numérique qui se dessine a besoin de citoyens éduqués autrement, avec des compétences nouvelles qui doivent s'acquérir en profondeur, c'est-à-dire dans la durée, par un apprentissage tant expérimental que conceptuel, dispensé par des enseignants correctement formés pour cela.

C'est l'analyse qui a été défendue par l'Académie des sciences [2] dans son rapport, puis par le Conseil national du numérique [3]. C'est celle que nous partageons avec un nombre croissant d'acteurs du numérique. C'est aussi celle qui a motivé nos efforts pour accompagner la formation des enseignants d'ISN (Informatique et Sciences du Numérique), puis la rédaction et la remise d'un rapport sur cette question fondamentale de formation [4].

L'innovation est particulièrement concernée par cette question de formation à l'informatique : une idée reçue est que l'innovation est le fruit d'un besoin à identifier, et que l'innovateur est la personne qui, ayant eu l'intuition de ce besoin, n'a plus qu'à se retourner vers des ingénieurs pour que ceux-ci lui livrent l'objet numérique nouveau et innovant. Or l'innovation est d'abord le fruit de compétences technologiques et scientifiques solides qui permettent à l'innovateur de mesurer ce qu'il peut ou ne peut pas faire. Et dans le cas du numérique, les compétences en informatique sont essentielles. L'innovation est un sujet important pour la SIF qui lui consacre une journée en septembre 2013.

L'enseignement doit être accompagné par l'accès à la culture informatique, en tant que science et en tant que technologie. Et c'est sur cet aspect "accès à la culture" que 1024 trouve toute sa place. L'informatique, pour être une science comme les autres, doit être une *science partagée*. Les informaticiens, fussent-ils étudiants, enseignants, enseignants-chercheurs, chercheurs, ingénieurs ou techniciens, en milieu industriel ou académique, partagent-ils une même culture ? La culture scientifique s'acquiert traditionnellement au lycée : elle s'enrichit ensuite en fonction des informations que nous recevons. Dans le cas de l'informatique, nous avons deux handicaps à surmonter : pas (encore !) d'enseignement avant le baccalauréat et des médias qui peinent à informer sans image. La situation est donc largement améliorable : quand un des inventeurs de l'informatique au XX^{e} siècle vient à décéder, quelle place dans les médias ? Quand un Français se voit remettre l'un des prix les plus prestigieux de l'informatique, qui pour relayer l'information ? Quelle réponse, quand on demande à nos collègues informaticiens les raisons pour lesquelles Shafi Goldwasser, Silvio Micali, Judea Pearl, Leslie Valiant ont obtenu, ces 3 dernières années, leur Turing

2. http://www.academie-sciences.fr/activite/rapport/rads_0513.pdf

3. http://www.cnnumerique.fr/enseignementinformatique/

4. http://www.societe-informatique-de-france.fr/actualite/2013/Rapport_SIF_sur_ISN.pdf

Award ? Sait-on ce que sont les sujets de recherche en réseaux, ou concernant les interfaces homme-machine aujourd'hui ?

C'est pour que cette culture informatique soit accessible à tous les informaticiens et informaticiennes que 1024 existe, afin que demain nous puissions, à l'instar des mathématiciens qui veulent nous expliquer la teneur des travaux du dernier médaillé Fields, ou des physiciens qui prennent le temps de nous informer des enjeux du dernier prix Nobel, discuter de notre science, la raconter et la faire partager aux autres.

Il ne me reste plus qu'à souhaiter à Éric Sopena et au comité de rédaction tout le succès que cette aventure mérite, et au lecteur de trouver beaucoup de plaisir dans ce premier numéro de 1024.

Mesurer la complexité des objets numériques

Jean-Paul Delahaye [1]

« L'image d'un monde antique baignant dans le chaos n'est pas neutre. Elle accrédite la notion d'une croissance progressive de la complexité au cours des ères. Elle relie l'apparition de l'intelligence et de la conscience au déroulement de l'histoire. Elle restitue l'être humain dans ce vaste mouvement d'organisation de la matière à l'échelle cosmique. »

Hubert Reeves, Dernières nouvelles du cosmos, Éditions du Seuil, Paris, 2002, pp. 224 et 225.

Quand on considère les différentes étapes de l'évolution de l'univers depuis le Big Bang telles qu'elles sont décrites par la cosmologie contemporaine, on est tenté de dire — comme Hubert Reeves aime y insister — qu'il se *complexifie*. De même, on affirme que les écosystèmes terrestres comportent des éléments de plus en plus nombreux et variés interagissant de façon de plus en plus subtile et délicate. À une échelle de temps plus réduite, on parle encore de la *complexification* des sociétés humaines, des échanges sociaux et commerciaux, des objets technologiques et des théories scientifiques. Mais de quoi s'agit-il exactement ? Qu'est-ce qui précisément augmente quand on parle de complexité croissante ? Peut-on mesurer ou au moins définir de manière rigoureuse cette "complexité" et en faire un concept mathématique ? La question préoccupe les chercheurs depuis longtemps et une multitude de méthodes et d'idées ont été proposées pour définir et évaluer la complexité des objets artificiels et naturels, des processus et des interactions.

Dans cet article, nous nous attacherons uniquement au problème de la définition et de la mesure de la complexité des *objets numériques* (textes, images, contenus

1. Université des Sciences et Technologies de Lille, Laboratoire d'Informatique Fondamentale de Lille, UMR 8022 CNRS, Bât M3-ext, 59655 Villeneuve d'Ascq Cedex. E-mail : delahaye@lifl.fr.

de disques durs, de disques optiques, de clef USB, de téléphone portable, etc.). De manière indirecte, cela concernera tous les objets du monde réel, dès lors qu'on accepte de les représenter par des données numériques. Nous n'aborderons pas la question de savoir si certains objets sont impossibles à *numériser*, ni la question presque équivalente de savoir si une quantité infinie d'informations peut se trouver présente dans un volume fini d'espace. La question de la complexité des algorithmes (classes P, NP, PSPACE, EXPTIME, etc.) est un autre sujet qui n'a la plupart du temps qu'un sens asymptotique, alors que justement nous voulons parler d'objets finis.

Nous allons voir qu'à côté de mesures simples comme l'espace mémoire occupé par l'objet sur un support numérique (mesure très insatisfaisante), ou l'entropie de Shannon (guère meilleure contrairement à ce que certains imaginent), d'autres mesures sont plus sérieuses car elles prennent en compte la richesse combinatoire, ou mieux encore se fondent sur la théorie du calcul.

Nous soutiendrons que la *complexité de Kolmogorov* (ou *complexité de Chaitin-Kolmogorov*) est l'une des plus importantes notions introduites pour mesurer la complexité des objets numériques et qu'il est possible de la mettre en œuvre grâce aux algorithmes de compression (sans pertes). Nous défendrons cependant l'idée que la *profondeur logique de Charles Bennett* (notion liée à la complexité de Kolmogorov) est la piste la plus sérieuse pour définir et mesurer la *complexité structurelle* des objets numériques. Grâce à des résultats accumulés année après année depuis son introduction en 1977, la profondeur logique de Bennett apparaît être la définition scientifique la meilleure dès lors qu'on ne confond pas "contenu incompressible d'information" (mesurée par la complexité de Kolmogorov) avec "richesse d'organisation" ou "complexité structurelle". La mise au point d'outils qui réussissent, au moins dans certains cas, à évaluer la profondeur logique de Bennett conduit à en envisager des applications.

La taille mesurée en bits ou en octets

Bien sûr, la mesure la plus simple de la complexité d'un objet numérique est la taille de l'espace mémoire (mesuré en bits ou en octets) qu'on utilise pour le conserver (sans chercher à le compresser). Si on suit cette idée, une image de dix millions de pixels, chacun codé par un octet (au total donc : 10 Mo), sera considérée comme dix fois plus complexe qu'une image d'un million de pixels (1 Mo). C'est une première proposition naturelle et simple, mais on réalise immédiatement qu'elle n'est pas satisfaisante puisqu'une image toute blanche avec une telle mesure serait considérée aussi complexe qu'une image de même taille représentant un paysage ou une puce électronique ou n'importe quoi, ce qui n'est pas du tout conforme à ce qu'on attend d'une mesure de complexité (quel que soit le sens qu'on donne à ce mot).

L'entropie

L'entropie de Shannon d'un texte ou fichier s composé de n symboles pris dans un alphabet à k symboles est définie par :

$$H(s) = n\Big(-\sum_i p_i \log p_i\Big)$$

où i décrit les k symboles de l'alphabet et p_i est la fréquence d'utilisation de chacun des symboles dans s. Si $p_i = 0$, on considère que $p_i \log p_i = 0$. Le logarithme utilisé est celui en base 2, c'est-à-dire, celui pour lequel $\log 2 = 1$.

Reprenons l'exemple de l'image toute blanche. Dans le cas d'un alphabet à 256 caractères, pour coder une image de n pixels, on trouve que l'image blanche (chaque pixel est représenté par le même caractère) a pour entropie 0 : la fréquence du caractère codant le blanc est 100%, et les autres symboles ont pour fréquence 0%.

Dans le cas d'une image aléatoire, chacun des symboles a une fréquence proche de $1/256$, et donc l'entropie d'un fichier aléatoire de n symboles est approximativement égale à :

$$n\Big(-\sum \frac{1}{256}\log\frac{1}{256}\Big) = n\Big(-\log\frac{1}{256}\Big) = 8n.$$

C'est la quantité d'information (mesurée en bits : $8n$ bits = n octets) contenue dans l'image (sans compression), c'est-à-dire la taille de l'espace mémoire occupé par l'image.

Dans le cas d'une image représentant un paysage ou un objet réel, l'entropie de Shannon sera intermédiaire entre celle d'une image blanche, et celle des images aléatoires qui sont les images ayant une entropie maximale. Si une image est composée pour moitié de pixels blancs et pour moitié de pixels noirs, l'entropie sera

$$-n\sum_{i=0,1}\frac{1}{2}\log\frac{1}{2} = n$$

qui à nouveau représente la quantité d'information à laquelle on peut réduire l'image quand au lieu de coder chaque pixel par un octet, on le code par un seul bit d'information.

D'une façon générale, l'entropie de Shannon indique la quantité d'information mesurée en bits qu'il faut, au minimum, pour coder l'image en utilisant un code de type "remplacer chaque suite de p symboles par une suite de bits plus ou moins longue d'une manière réversible, c'est-à-dire de telle façon que le décodage ne présente pas d'ambiguïté" (codage de Huffman).

Exemple. Si le symbole 'a' est utilisé avec la fréquence $\frac{1}{2}$, 'b' avec la fréquence $\frac{1}{4}$, 'c' avec la fréquence $\frac{1}{8}$, 'd' avec la fréquence $\frac{1}{8}$, alors on codera 'a' par 0, 'b' par 10, 'c' par 110, 'd' par 111 (on réserve les codes courts pour les symboles les

plus fréquents). Avec les fréquences indiquées, un fichier de n pixels comporte $\frac{n}{2}$ ‘a’ (demandant chacun 1 bit), $\frac{n}{4}$ ‘b’ (demandant chacun 2 bits), $\frac{n}{8}$ ‘c’ (demandant chacun 3 bits), $\frac{n}{8}$ ‘d’ (demandant chacun 3 bits). Pour le fichier complet, on arrive à environ $\frac{n}{2}+\frac{2n}{4}+\frac{3n}{8}+\frac{3n}{8}=\frac{7n}{4}$ bits. Le codage proposé est optimal parmi les codages envisagés et ce qu’il donne est exactement ce qu’indique l’entropie :

$$-n\Big(\sum \frac{1}{2}\log\frac{1}{2}+\frac{1}{4}\log\frac{1}{4}+\frac{1}{8}\log\frac{1}{8}+\frac{1}{8}\log\frac{1}{8}\Big)=n\Big(\frac{1}{2}+\frac{2}{4}+\frac{3}{8}+\frac{3}{8}\Big)=\frac{7n}{4}.$$

Sur tout cela on consultera [16].

À première vue, l’entropie de Shannon apparaît comme une mesure assez satisfaisante de complexité. En y regardant de plus près, on découvre cependant que cette mesure attribue la même complexité

– d’une part, à une image composée “d’une moitié droite toute noire, et d’une moitié gauche toute blanche”,

– et d’autre part, à une image utilisant “50% de pixels blancs et 50% de pixels noirs disposés aléatoirement” ou représentant “un objet non trivial (paysage, personnage, machine, etc.) qui comporte 50% de pixels blancs et autant de pixels noirs”.

C’est absurde ! Ces images ont de toute évidence des complexités différentes : la première n’est pas complexe du tout, alors que les images d’objets réels ont certainement une assez grande complexité (quel que soit le sens qu’on donne à ce mot). L’entropie de Shannon ne convient pas comme mesure de complexité, car la fréquence d’utilisation des différents symboles ne détermine pas leur ordre dans un texte ou leur position dans une image, alors que la complexité d’un texte ou d’une image dépend évidemment de cet ordre ou de ces positions.

D’autres mesures inspirées de l’entropie de Shannon sont envisageables pour tenter de résoudre son utilisation directe insatisfaisante. On peut par exemple considérer l’entropie de Shannon des couples de deux symboles consécutifs, ou des carrés de 4 pixels extraits de l’image qu’on analyse. Ce qu’on obtient sera un peu meilleur, mais ne sera jamais pleinement satisfaisant car la complexité d’une image ne dépend pas seulement de la statistique sur les carrés de 4 pixels qui, comme les pixels, peuvent pour une statistique fixée être rangés simplement (par exemple en regroupant les carrés identiques) ou composer des motifs nombreux et variés, donc plus complexes.

Mesures combinatoires particulières

Pour les textes en langue naturelle, la richesse du vocabulaire est une marque naturelle de leur complexité. De même, le nombre de sous-mots d’une taille donnée, p, pris dans un fichier s en mesure la richesse, donc d’une certaine façon la complexité. Cette “variété combinatoire” qui cette fois prend en compte l’ordre des symboles est facile à mesurer puisqu’il suffit juste de mener un décompte.

Si p est un entier fixé et s une suite (finie ou infinie) de symboles pris dans un alphabet fixé (à k symboles), on appelle *fonction de complexité* (ou *complexité combinatoire*) de s, la fonction $p \to \mathrm{cc}_s(p)$ qui à p associe le nombre de mots différents de p symboles qu'on trouve dans s. Bien sûr :

$$0 \leq \mathrm{cc}_s(p) \leq k^p.$$

La fonction de complexité combinatoire de la suite `000....0...` est constante et égale à 1. Si une suite est périodique de période r alors $\mathrm{cc}_s(p) = r$ pour p assez grand. Les suites périodiques ne sont pas très complexes, et cela se voit sur leur fonction de complexité qui est bornée. On montre d'ailleurs qu'il y a équivalence entre les propriétés "être ultimement périodique" et "avoir une fonction cc_s bornée".

Un remarquable résultat récent dû à Boris Adamczewski et Yann Bugeaud [1] affirme que les nombres irrationnels algébriques ont des développements en base b ($b \geq 2$) dont la fonction de complexité combinatoire, cc, vérifie :

$$\lim_{p\to\infty} \frac{\mathrm{cc}(p)}{p} = \infty.$$

Ce résultat exprime que les développements en base b de ces nombres sont complexes et en particulier plus complexes que ceux des nombres rationnels qui sont ultimement périodiques, ce qui implique pour leurs développements décimaux que :

$$\lim_{p\to\infty} \frac{\mathrm{cc}(p)}{p} = 0.$$

Le résultat n'est pas une surprise, mais on attendait une démonstration et dans ce domaine tout résultat est difficile ! Notons qu'il ne peut pas y avoir de résultat du même type pour les nombres transcendants dont certains ont des fonctions de complexité combinatoire linéairement croissantes.

Même si ce type de mesure combinatoire de la complexité est intéressant, on ne peut pas s'en satisfaire. De toute évidence, la suite de chiffres décimaux suivante (les chiffres des entiers les uns derrière les autres) :

`0123458789101112131415161718912021222324 2526...`

introduite par Champernowne en 1933 n'est pas complexe. En effet, elle se décrit très brièvement, on la programme sans mal, et quelqu'un qui en voit les premiers termes devine aisément les suivants. Pourtant sa complexité combinatoire est maximale : non seulement $\mathrm{cc}_s(p) = 10^p$ pour chaque p, mais on prouve que toutes les séquences m de longueur p (de chiffres de 0 à 9) apparaissent une infinité de fois et ont pour fréquence limite d'apparition $\frac{1}{10^p}$ (on dit que le nombre réel dont l'écriture en base 10 est cette suite de chiffres est un nombre *normal*).

Mesurer la complexité d'une suite s (finie ou infinie) par la variété des sous-mots qu'on y rencontre ne convient pas lorsqu'on aborde des objets un peu subtils dont la suite de Champernowne n'est qu'un exemple parmi une infinité d'autres. La variété

des sous-mots d'une suite numérique, s, est un indice de sa complexité et évaluer cette variété donne une certaine indication de cette complexité (d'où l'intérêt du résultat sur les nombres algébriques irrationnels), mais pas d'une manière entièrement satisfaisante. Il faut aller plus loin !

La complexité cc est utilisée pour établir des propriétés des générateurs pseudo-aléatoires : si elle est élevée, on a une assurance que le générateur n'est pas trop mauvais. D'autres mesures combinatoires du même type sont considérées par les chercheurs de ce domaine et sont utilisées en cryptographie. Chacune pourrait se présenter comme une mesure de complexité, dont malheureusement on constaterait qu'elle est insatisfaisante dès qu'on a affaire à des séquences possédant des régularités un peu cachées ou subtiles (comme les chiffres du nombre π). Voir par exemple [5].

Pour tester les générateurs pseudo-aléatoires, une multitude de tests statistiques sont aussi utilisés. Chacun d'eux pourrait donner une mesure de complexité en utilisant le principe suivant : meilleure est la satisfaction au test, plus grande est la complexité de la séquence testée. Nous ne donnons pas de détails sur ces mesures et renvoyons à la publication [40].

Aussi utiles en pratique qu'ils soient, ces tests statistiques ne sont pas satisfaisants dans l'absolu. En effet, la suite des chiffres de la constante π (ou de n'importe quel nombre irrationnel algébrique) en base 2 (ou autre) passe ces tests de manière satisfaisante : aucune exception n'a jamais été trouvée. Elle est donc maximalement complexe à leurs yeux. Pourtant, du fait d'une définition simple et de l'existence de méthodes assez rapides pour calculer les chiffres de cette suite, elle ne peut pas être considérée comme la plus complexe possible : les tests statistiques se trompent au sujet de sa complexité. Nous allons y revenir.

Pour se sortir de cette profusion de méthodes de mesures de la complexité des objets numériques, dont pourtant aucune n'est vraiment satisfaisante, une idée un peu plus abstraite était nécessaire. C'est de la théorie de la calculabilité qu'elle est venue.

La complexité de Kolmogorov

Une idée implicite dans les tentatives de mesure évoquées jusqu'à présent est que "le simple est ce qui se décrit brièvement, et le complexe ce qui à l'inverse ne peut se décrire brièvement". C'est l'idée de la de complexité de Kolmogorov introduite en 1965 par Andrei Kolmogorov (et sous des formes presque équivalentes, indépendamment et à peu près au même moment, par Ray Solomonoff et Gregory Chaitin). Pour plus de détail sur l'histoire de cette notion, on consultera la référence absolue sur le sujet [35] ou, en moins encyclopédique, [17, 19, 20].

Donnons très rapidement quelques précisions sur cette mesure de complexité qui ne possède plus les défauts des mesures évoquées jusqu'ici.

La *complexité de Kolmogorov*, $K(s)$, d'une suite binaire finie s est par définition la longueur du plus court programme qui engendre s, c'est-à-dire du plus court programme qui produit l'impression de s sur l'écran de l'ordinateur, sur l'imprimante ou dans un fichier.

Le plus court programme en question s'appelle le *programme minimal pour s*, on le notera s^*. S'il y a plusieurs programmes de taille minimale donnant s, on prend pour s^* le premier dans l'ordre lexicographique. Les langages que mentionne la définition sont les langages informatiques de programmation : langage C, langage Java, langage Fortran, langage LISP, etc.

Plusieurs questions se posent à la lecture de la définition.

– Comment faire pour les suites qui ne sont pas binaires ?

Moyennant une conversion (équivalente au passage de la base de numération b à la base 2), il est clair qu'on se ramène facilement à des suites binaires et des programmes binaires.

– Est-ce que la définition dépend du langage de programmation choisi ?

Oui bien sûr, mais assez peu. En effet, si on ne considère que des machines puissantes (c'est-à-dire équivalentes à des machines de Turing universelles), alors le résultat suivant répond à la question et donne sa consistance à la notion de complexité de Kolmogorov en montrant que la définition de $K(s)$ varie faiblement quand on change de langage de référence dans la définition de $K(s)$.

Théorème d'invariance. *Si U et V sont deux machines de Turing universelles (machines qui pour la donnée $[i,d]$ calculent ce que la machine de Turing numéro i calcule pour la donnée d), et si on note $K_U(s)$ (respectivement $K_V(s)$) la complexité de Kolmogorov quand on utilise la machine U (respectivement V) comme machine de référence, alors il existe une constante $C_{U,V}$ indépendante de s, telle que pour toute suite finie s :*

$$|K_U(s) - K_V(s)| < C_{U,V}.$$

Ce résultat est essentiel puisqu'il signifie "qu'à une constante additive près", le changement de langage de programmation (ou de machine de Turing universelle, ce qui revient au même) ne modifie pas la complexité de Kolmogorov d'une suite s. Que vous utilisiez des programmes écrits en langage Java ou en Pascal, Fortran ou C (chacun est un langage universel, chacun est équivalent à une machine de Turing universelle), la complexité que vous définissez en mesurant la longueur des plus courts programmes restera sensiblement la même, et sera donc très sensiblement la même si s est une suite assez longue.

L'idée de la définition de Kolmogorov est la formalisation mathématique de : "est complexe ce qu'on ne peut pas définir brièvement". Avant les développements de la théorie de la calculabilité (Gödel, Church, Turing, etc.), cette idée ne pouvait pas être

mise en œuvre dans le champ mathématique. La notion de longueur d'une définition (dans un langage formel) n'est pas invariante, même à une constante additive près. C'est bien la mise au point de la théorie de la calculabilité qui a rendu mathématisable l'idée intuitive et naturelle que simple peut signifier "définissable en peu de symboles" et complexe "impossible à définir brièvement".

Donnons des exemples pour éclairer la définition.

(a) *Une suite répétitive élémentaire.*

La complexité de Kolmogorov d'une suite s, composée d'un milliard de '0' (ou d'une image toute blanche) est très petite. En effet, écrire un programme du type :

```
pour i variant de 1 jusqu'à 1000000000 imprimer 0
```

(programme traduisible dans tout langage de programmation) prend peu de symboles. Puisque la longueur d'un tel programme est de quelques dizaines de bits, 200 tout au plus, cela signifie que : $\mathrm{K}(s) < 200$.

Une suite binaire de moins de 200 bits définit une suite binaire d'un milliard de bits : la représentation par un programme (vue comme une définition) fait gagner un espace considérable. La définition de s par programme a une longueur inférieure à 0,00002% de la longueur de l'objet s représenté. Un milliard de fois '0' est une longue suite, mais conformément à ce qu'on attend, c'est une suite simple.

(b) *Un milliard de chiffres binaires de π.*

La complexité de Kolmogorov de la suite s composée d'un milliard de chiffres binaires de π est relativement faible. On peut en effet calculer s avec un programme assez court en utilisant une des nombreuses formules de série que l'analyse mathématique propose pour définir ou calculer π. Très concrètement, ce que nous savons de π montre que :

$$\mathrm{K}(s) < 2000.$$

Cette seconde suite est donc un peu moins simple que la première, mais au sens de la complexité de Kolmogorov, elle reste particulièrement simple puisque sa complexité de Kolmogorov est inférieure à 0,0002% de sa longueur.

Voici un programme en langage C de 158 caractères qui engendre 1600 décimales de π :

```
int a=10000,b,c=8400,d,e,f[8401],g;main(){for(;b-c
;)f[b++]=a/5;for(;d=0,g=c*2;c-=14,printf("%.4d",e+
d/a),e=d%a)for(b=c;d+=f[b]*a,f[b]=d%--g,d/=g--,--b
;d*=b);}
```

En Haskell le programme suivant donne autant de digits que la mémoire de la machine le permet :

```
pi = g(1,0,1,1,3,3) where
  g(q,r,t,k,n,l) = if 4*q+r-t<n*t
```

```
then n : g(10*q,10*(r-n*t),t,k,div(10*(3*q+r))t-10*n,l)
else g(q*k,(2*q+r)*l,t*l,k+1,div(q*(7*k+2)+r*l)(t*l),l+2)
```

Ce programme, dû à Jeremy Gibbons [25], comporte 200 caractères (en comptant les blancs et les retours à la ligne).

C'est donc bien un programme de moins de 2000 chiffres binaires qu'il faut pour calculer un milliard de chiffres binaires de π. Le programme obtenu par ce moyen n'est pas nécessairement le plus court possible permettant d'avoir un milliard de chiffres binaires de π, mais sa longueur est plus grande que celle du programme le plus court qui sert à définir K(s).

(c) *Une suite de chiffres au hasard.*

Une suite de chiffres binaires tirés au hasard — par exemple en lançant une pièce de monnaie non truquée — a toutes les chances d'être complexe au sens de Kolmogorov, c'est-à-dire d'avoir une complexité de Kolmogorov élevée. En effet, un argument de dénombrement montre précisément que :

parmi toutes les suites binaires de longueur n (n fixé), moins d'une suite sur 2^k a une complexité de Kolmogorov inférieure à $n-k$.

On note que le nombre de programmes corrects (c'est-à-dire dont la syntaxe est conforme aux exigences du langage choisi pour définir K) de longueur inférieure à $n-k$ est au plus 2^{n-k}.

Il y a au plus 2 programmes corrects de longueur 1, 4 de longueur 2, 8 de longueur 3, ..., 2^{n-k-1} de longueur $n-k-1$, et la somme de ces nombres vaut $2^{n-k}-2$. En conséquence, la proportion de suites de longueur n possédant un programme minimal de longueur inférieure à $n-k$ est inférieure à :

$$\frac{2^{n-k}}{2^n} = \frac{1}{2^k}.$$

Un peu plus de précision est même possible : une suite binaire s de longueur n, prise au hasard, possède, avec une probabilité très forte, une complexité de Kolmogorov, K(s), d'environ n.

En effet :

(1) d'une part, le résultat précédent montre que K(s), sauf cas exceptionnels (exponentiellement rares), ne sera pas nettement inférieur à n, et

(2) d'autre part, les programmes du type `imprimer "001001001..."` (leur longueur est approximativement n) montrent que jamais K(s) n'est sensiblement supérieur à n, la longueur de s.

Le programme minimal d'une suite s peut être considéré comme une version compressée de s et en pratique ce sont d'ailleurs les algorithmes de compression de données sans perte qui constituent les meilleurs outils d'évaluation de K(s).

(d) *Un exemple mixte.*

Voici un quatrième exemple qui fera comprendre un peu plus en profondeur ce qu'est la complexité de Kolmogorov. Il s'agit d'un exemple où l'aléatoire et l'organisation sont mélangés.

On prend une suite générée aléatoirement de 500 millions de chiffres binaires :

$$s = 011010101000010100001000\ldots$$

D'après ce que nous avons vu, sa complexité de Kolmogorov est environ 500 millions. On transforme cette suite de 500 millions de chiffres binaires en une suite s' de 1 milliard de chiffres binaires en doublant chaque 0 et en doublant chaque 1 :

$$s' = 001111001100110011000000001100110000000011000000\ldots$$

La suite s commence par 011, la suite s' commence donc par 001111.

La complexité de Kolmogorov de cette suite d'un milliard de chiffres binaires est à peu près 500 millions. En effet :

– Elle ne peut pas être moins, car si on disposait d'un programme de moins de 500 millions de bits produisant s', on pourrait en le modifiant légèrement obtenir un autre programme de moins de 500 millions de bits environ produisant s (le nouveau programme P' serait la copie du premier P, mais n'imprimerait qu'un digit sur deux, moyennant une petite modification qui n'en change que très peu la longueur). Or cela n'est pas possible car cela signifierait que sa complexité de Kolmogorov est inférieure à 500 millions, ce qui n'est pas le cas.
– Elle ne peut pas être beaucoup plus, car en mettant dans un programme la donnée des 500 millions de bits de s et en plaçant quelques instructions pour que le programme recopie chaque bit de s deux fois de suite, on a un programme de longueur 500 millions environ qui produit s'.

Ici une structure simple — le doublement de chaque digit —, mélangée à de l'aléatoire donne une suite de longueur 1 milliard, ayant une complexité de 500 millions. D'une manière générale, toute régularité (statistique, algébrique, arithmétique, etc.) dans une suite de longueur n, fait rapidement baisser sa complexité de Kolmogorov.

Trouver des programmes minimaux est très difficile et on montre que la complexité de Kolmogorov n'est pas calculable par algorithme. Cependant tout algorithme de compression de données peut être vu comme un calculateur approché de la complexité de Kolmogorov. S'il repère bien les régularités du fichier numérique qu'on lui soumet et les exploite au mieux la taille du fichier qu'il produit est une approximation satisfaisante de la complexité de Kolmogorov. Bien sûr, aucun algorithme de compression ne repère toutes les régularités possibles qu'on peut trouver

dans un fichier et le résultat d'incalculabilité mentionné signifie précisément qu'aucun algorithme ne réussira jamais à repérer toutes les régularités qu'on peut mettre dans une suite ou un fichier.

La méthode de calcul de la complexité de Kolmogorov par les algorithmes de compression de données — et plus spécialement la mesure de la complexité de Kolmogorov relative — a conduit à de nombreuses applications : classification de textes [14], de musiques, reconstitution d'arbres phylogénétiques [44, 29], détection de spam [10], étude des séries de cours boursiers, repérage de flux anormaux d'informations et d'intrusions, etc.

La complexité de Kolmogorov est très ambitieuse puisqu'elle revient à prendre en compte toute régularité présente dans un fichier (et pas seulement certaines régularités statistiques, comme le fait l'entropie de Shannon), on dit qu'elle est *universelle*. Cela a une double conséquence.

(1) Elle est impossible à calculer exactement pour tout fichier numérique (ce qui ne l'empêche pas d'être approchable). C'est ennuyeux mais pas si grave comme l'attestent les mises en œuvre pratiques nombreuses à l'aide d'algorithmes de compression.

(2) Elle n'a pas les défauts de toutes les mesures combinatoires, statistiques mentionnées précédemment qui ne captent qu'une partie — souvent assez réduite — de la notion générale de complexité en n'envisageant que des catégories limitées de régularités.

Les nombreuses applications mises en œuvre récemment prouvent que l'avantage de l'universalité, associée à l'effectivité des algorithmes de compression de données — qui sont de petites machines intelligentes travaillant à repérer des régularités — est aujourd'hui en train de l'emporter sur les conceptions ad hoc, souvent étroites et toujours faciles à prendre en défaut, qu'on a utilisées sans, bien souvent, en comprendre les insuffisances graves.

Cependant la complexité de Kolmogorov est une mesure de la complexité au sens de "contenu incompressible d'information". Elle n'est pas une mesure de "richesse en structure". D'ailleurs les suites aléatoires qui n'ont aucune structure sont pour la complexité de Kolmogorov les plus complexes.

C'est qu'en fait, il existe deux notions intuitives très différentes de complexité que le langage courant confond et associe à tort.

– Il y a une notion de complexité comme richesse en information que la variété combinatoire — ou mieux, l'incompressibilité — mesure et dont la version ultime est la complexité de Kolmogorov.
– Il y a une notion de complexité comme richesse en structure, comme quantité d'organisation, comme difficulté à être élaboré ou calculé, comme information de valeur. Cette notion est plus ou moins indépendante de la première

notion. C'est cette seconde notion qu'on évoque — sans nécessairement en avoir conscience — quand on parle de complexification.

La première notion est clairement celle que la complexité de Kolmogorov formalise et on peut considérer que le succès de cette formalisation est total. La seconde notion que pour l'instant nous n'avons évoquée que de manière informelle peut-elle être définie rigoureusement dans un cadre mathématique ? Si on y arrive, cela donnera une base mathématique à l'idée de la complexification. La réponse est que la profondeur logique de Charles Bennett (et une série de notions plus ou moins liées) se présentent comme de telles mesures de richesse, de valeur, d'utilité de l'information. Nous en donnons une présentation rapide plus loin.

L'image souvent défendue de l'univers passant d'un désordre total (complexité aléatoire maximum) à un ordre de plus en plus grand et donc à une forme de mort (parfois appelée "mort thermique") par uniformisation est une image fausse. L'univers évolue en créant, par des processus assimilables à du calcul, des objets de plus en plus complexes, mais la complexité dont il s'agit alors est celle liée à l'organisation, à la variété et la richesse des structures. Il n'y a pas deux pôles opposés dont les extrêmes seraient "l'aléa" et "l'uniforme" (la complexité aléatoire et la simplicité totale), mais trois pôles dont les points extrêmes sont "l'aléa", le "simple absolu ou le rien", et le "très organisé" (probablement sans maximum).

La profondeur logique de Bennett

Formulée en 1977 par Charles Bennett mais rendue précise en 1988 [11] la proposition de définition d'une mesure de la complexité structurelle se fonde sur l'idée d'un "contenu en calcul", qui s'oppose à l'idée d'un "contenu en information". Pour mesurer ce contenu en calcul, l'idée naturelle de considérer le temps de calcul du programme le plus rapide donnant s ne conduit qu'à une notion triviale (car ne dépendant pour l'essentiel que de la longueur de s). Il faut être un peu plus subtil et ne considérer que le programme minimal (ou les programmes courts dans les versions paramétrées de la notion).

La *profondeur logique* (ou *profondeur de Bennett*), $\mathrm{P}(s)$, de la suite binaire finie s est définie par :

$$\mathrm{P}(s) = \textit{temps de calcul du programme minimal } s^* \textit{ de } s.$$

L'unité de mesure de ce temps de calcul est le pas élémentaire de calcul pour le modèle de machine que l'on considère, ou une unité de temps d'exécution d'un programme si on se réfère à $\mathrm{K}(s)$ en se basant sur des langages universels de programmation.

L'article principal sur le sujet est celui publié en 1988 par Charles Bennett [11], mais on lira avec intérêt les autres textes de Bennett [12, 13]. L'article de 1988 donne les détails de la théorie de la profondeur logique. Il explore non seulement la définition donnée ci-dessus, mais plusieurs variantes.

La définition de profondeur logique proposée par Bennett est-elle satisfaisante et doit-on croire qu'on dispose avec elle d'une bonne mesure de complexité organisée (ou complexité structurelle), complémentaire de la notion de complexité de Kolmogorov qui, on l'a vu, mesure la complexité aléatoire ? La discussion est délicate, les arguments nombreux et variés et la conclusion n'est pas considérée unanimement comme acquise.

Commençons notre examen en reprenant l'exemple de la suite aléatoire ($K(s) \approx$ longueur(s)) qui nous a conduit à conclure que la complexité de Kolmogorov n'est pas le bon concept de complexité organisée : une suite aléatoire n'est pas organisée et possède pourtant une forte complexité de Kolmogorov. Que donne la profondeur logique de Bennett pour une telle suite s, obtenue par tirage au sort ?

Une telle suite, s, possède par définition une complexité de Kolmogorov $K(s)$ équivalente à sa longueur. Il en résulte que son programme minimal est sans doute équivalent à un programme du type "imprimer s" ou est proche d'un tel programme. Or, pour toute machine raisonnable et pour tout langage de programmation raisonnable exécuté par une telle machine, le temps d'exécution d'un programme du type "imprimer s" sera linéaire ou quasi-linéaire, c'est-à-dire petit (puisque le temps de calcul d'une suite de longueur n ne peut pas être inférieur à n en nombre de pas de calcul). Autrement dit, une suite aléatoire possède toujours une faible profondeur logique de Bennett : lorsqu'une suite n'a pas de structure et est totalement désordonnée, il n'y a rien de mieux à faire que d'écrire s dans le programme qui doit la produire (qui est donc du type "imprimer s") et alors, un tel programme fonctionne rapidement. Nous attendions cette propriété qui faisait défaut à la complexité de Kolmogorov.

Il ne s'agit pas là d'un argument définitif, mais il est particulièrement frappant que le principal problème de la complexité de Kolmogorov (elle est maximale pour les suites aléatoires pourtant sans organisation), défaut qui conduit à l'écarter comme mesure de complexité structurelle, disparaisse quand on prend en considération la profondeur logique de Bennett : la profondeur logique d'une suite aléatoire est minimale comme elle l'est pour une suite simple (par exemple composée uniquement de '0').

Une analyse heuristique de la profondeur de Bennett nous conduit aussi à la conclusion que, plus un objet est finement structuré, plus le temps nécessaire au passage de sa description la plus compressée (son programme minimal) à l'objet lui-même sera long. De ce fait — et dans le cas des chiffres de π les arguments peuvent être rendus précis — la profondeur logique de Bennett semble bien convenir : très faible pour une longue suite répétitive de 0 ou une suite incompressible ($K(s)$ très faible ou $K(s)$ maximum) ; moyenne pour une suite comme celle des chiffres de π (ayant un certain contenu en calcul) ; elle est élevée dans le cas d'un objet fortement structuré ou possédant une organisation complexe. C'est exactement ce que l'on attendait.

Schématiquement, il y aurait donc quatre sortes d'objets :

(1) les objets K-simples et peu profonds : une suite uniforme de '0', une suite répétant un même motif (cristal), etc. ;

(2) les objets K-simples et profonds : la suite des chiffres de π, la suite de Prouhet-Thue-Morse, etc. ;

(3) les objets aléatoires et peu profonds : les suites typiques de longueur n ;

(4) les objets aléatoires et profonds : microprocesseurs, ordinateurs, cellules vivantes, villes, organisations sociales, écosystèmes, etc.

Progrès récents

Bien qu'assez lentement, une série d'avancées dans la compréhension et l'utilisation de la profondeur logique de Bennett ont eu lieu depuis sa définition en 1977. Il faut les voir comme la confirmation que l'idée de Bennett est bonne et que, même si rien n'est très facile dans ce domaine soumis à des nombreuses "indécidabilités", il faut poursuivre son étude qui de toutes les façons, ne serait-ce que d'une manière abstraite, nous aide à saisir la nature de la complexité dont notre compréhension intuitive doit maintenant devenir mathématique et scientifique.

Des versions mathématiques plus avancées et plus précises de la notion de profondeur logique de Bennett ont été proposées par Bennett [11]. Ces versions ne se contentent pas de faire intervenir le programme le plus court produisant s, mais prennent en compte tous les programmes produisant s qui ne peuvent être compressés de plus de k digits ou même tous les programmes produisant s mais en accordant plus de poids aux programmes les plus courts qui sont considérés comme des origines plus probables de l'objet s. Ces versions possèdent de meilleures propriétés que P(s) dont celle d'autoriser la démonstration d'une "loi de croissance lente" qui indique que la complexité structurelle ne peut pas croître rapidement. Malheureusement, ces définitions plus satisfaisantes sur un plan abstrait sont plus délicates à mettre en application que la définition de base que nous avons notée P(s).

Parmi les tentatives de définition d'une mesure de complexité structurelle, certaines sont issues de la théorie du calcul tout en étant différentes de celle de Bennett. Ces autres notions sont en compétition avec l'idée de Bennett et il est donc nécessaire de comprendre s'il s'agit de notions liées les unes aux autres. La notion de "sophistication" introduite par Moshe Koppel et Henri Atlan [31, 32, 33], la notion de "complexité effective" de Seth Lloyd et Murray Gell-Mann [24] et la notion de "facticité" de Adriaans [4] se fondent toutes les trois sur l'idée d'une séparation possible entre la composante aléatoire et la composante organisée d'un objet. L'idée des définitions proposées par ces chercheurs sous des formes mathématiques qui en masquent parfois le côté naturel est la suivante. Quand on examine un programme produisant un objet, il est possible — au moins dans certain cas — d'y reconnaître

deux parties : l'une représente ce qui fixe et décrit la structure de l'objet, l'autre est ce qui habille la structure par des éléments de moindre importance ou même aléatoires.

Considérons par exemple une suite finie s de paires de chiffres binaires 00 ou 11 dont la succession est aléatoire :

$$s = \texttt{00 11 11 00 00 00 11 00 11 00 00 11 00 11 11}$$

Le plus court programme s^* produisant cette suite comportera deux parties assez faciles à identifier, car il sera de la forme :

`imprimer en doublant chaque chiffre la suite 011000101001011.`

La structure régulière de s est exprimée par la partie "imprimer en doublant chaque chiffre". La composante désordonnée se trouve quant à elle dans la partie descriptive et incompressible de 0 et de 1 qui termine le programme.

Lorsqu'on aura affaire à une suite s plus structurée que celle de notre exemple, la longueur de la partie non aléatoire sera plus grande. C'est cette longueur qui, selon Koppel, Atlan, Lloyd et Gell-Mann, Adriaans, mesure la richesse en organisation — la complexité structurelle — de la suite s.

La partie du programme où on a inscrit la séquence sans doubler les digits `011000101001011` est d'autant plus longue que la séquence s comporte beaucoup d'éléments. C'est la partie aléatoire de la séquence et sa longueur est en gros la complexité de Kolmogorov de s.

Des difficultés techniques rendent délicate la formulation mathématique de cette idée de séparation en deux parties d'un programme court donnant s, mais ce que nous venons d'expliquer est l'essence de l'idée de la sophistication, de la complexité effective et de la facticité. Puisque ces définitions sont aussi naturelles sans doute que la définition de la profondeur logique de Bennett, la question était posée de savoir si un lien existe entre ces deux visions, en apparence assez différentes, de la complexité structurelle. Un début de réponse a été donné en 2010 dans un remarquable article [8] de Nihat Ay, Markus Müller, et Arleta Szkola du Max Plank Institute de Leipzig. Les chercheurs y établissent formellement qu'une grande complexité effective entraîne toujours une grande profondeur logique, confirmant indirectement le bien fondé de la définition de Bennett.

Les recherches récentes ont conduit à un second progrès de la théorie qui provient de la mise au point de versions calculables de la profondeur logique [6, 7, 22, 34]. Ces versions font des hypothèses simplificatrices et restrictives sur les processus de calcul, mais aboutissent à des définitions débarrassées de la gênante incalculabilité de $\mathrm{P}(s)$.

Autre motif de satisfaction pour ceux qui pensent que le concept de profondeur logique de Bennett ou les concepts voisins sont une clef pour comprendre le monde réel a été de le voir reconnu dans des travaux scientifiques liés à ce qui dans le monde est une sorte de grand calcul : l'évolution biologique. Antoine Danchin [18] de l'Institut Pasteur, Guillaume Baptist [9] de l'Université de Grenoble et John Collier [15]

de l'université de Durban en Afrique du Sud défendent tous les trois indépendamment l'un de l'autre l'idée que la profondeur logique de Bennett est un concept utile et important en biologie. En effet, elle permet d'éviter certaines considérations trop naïves sur la complexification des êtres vivants et elle éclaire l'idée d'un progrès dans les lignées d'organismes vivants — progrès qui ne serait que l'accumulation de résultats de calculs.

Expérimentation

Parmi les dernières avancées dans la démonstration que la profondeur logique de Bennett est le bon concept mathématique permettant de définir et de mesurer la complexité structurelle, divers travaux de nature expérimentale ont conduit à des débuts de validation pratique de l'idée de Bennett dont les fondements intuitifs ne font pas de doute, mais dont il restait à montrer qu'elle possède aussi un potentiel d'application.

Un premier travail [45] a consisté à mesurer vraiment le temps de calcul du plus court programme produisant une image. Mené par Hector Zenil de l'Université de Sheffield, Cedric Gaucherel du Département d'Ecologie de l'Institut Français de Pondichéry en Inde et moi-même, la méthode évalue le temps de décompression nécessaire pour des images représentant divers objets plus ou moins structurés. Ce temps de décompression est assimilable à P(s) car le fichier compressé de s est une version approchée de s^* et il en résulte que le calcul pour passer du fichier compressé au fichier complet est assimilable au calcul qui s'effectue à partir de s^* pour revenir à s, c'est-à-dire P(s). Les résultats de ces mesures de temps de calcul ont été utilisés pour ordonner les images par complexité structurelle croissante. La classification obtenue est proche de celle attendue. Elle place au début du classement les images parfaitement simples ou totalement aléatoires, et à la fin du classement les images des objets les plus riches en structures.

Un second travail [42], fondé sur l'énumération massive de petites machines de Turing, a été mené par Fernando Soler-Toscano de l'Université de Séville, Hector Zenil, Nicolas Gauvrit de l'Université de Paris VII et moi-même. Il a permis d'évaluer pour des courtes séquences de '0' et de '1', leur complexité de Kolmogorov et leur profondeur logique de Bennett. Ces calculs montrent, comme on s'y attendait, que les deux notions de complexité sont indépendantes et que la profondeur logique de Bennett est faible aussi bien pour les suites simples (10 fois '0' par exemple) que pour les suites d'apparence aléatoire. À nouveau la coïncidence entre théorie et calcul approché montre que malgré l'indécidabilité théorique des notions, on peut tenter d'en faire quelque chose, et que ce qu'on trouve alors est conforme à ce que l'intuition attend. Ces expérimentations et applications à la classification des objets numériques selon leur complexité structurelle ne sont que balbutiantes, mais pourraient se révéler réellement utiles, pourvu que l'idée qu'il y a deux complexités bien

différentes (liées respectivement au *contenu en information* et au *contenu en calcul*) fasse son chemin, et que dans toutes les disciplines où on les rencontre, on cherche à les mesurer par les techniques de compression sans perte. Ces dernières sont susceptibles de progrès sans limite (on n'aura jamais d'outil parfait de détection des régularités, mais en disposant de capacité de calcul croissante on réussira de mieux en mieux) et devraient donc conduire à des outils de mesure de plus en plus fins et robustes des deux complexités identifiées et formalisées par la théorie de la calculabilité.

On le voit la définition et la mesure de la complexité sont des sujets difficiles dont on a sans doute pas encore identifié toutes les clefs. Les progrès récents sont cependant remarquables sur tous les aspects du problème :

(1) démonstration de la complexité (combinatoire) des développements décimaux de certains nombres réels ;

(2) meilleure maîtrise des générateurs de suites pseudo-aléatoires, grâce aux outils évaluant concrètement la complexité de leurs productions ;

(3) définition d'un notion universelle de complexité aléatoire (la complexité de Kolmogorov) et mise au point d'outils en permettant la mesure et l'utilisation ;

(4) identification d'une dualité longtemps restée confuse entre complexité aléatoire et complexité structurelle avec propositions de définitions mathématiques précises (dont la profondeur logique de Bennett) ;

(5) résultats et méthodes de calcul approché en cours de mise au point pour cette notion dont l'importance conceptuelle et pratique est devenue évidente.

Références

Il existe des centaines d'articles et de livres abordant le thème de la définition et de la mesure de la complexité. Nous nous sommes astreints à n'en mentionner qu'un peu moins d'une cinquantaine en choisissant ceux directement évoqués dans l'article ou ceux les plus utiles. Dans le texte de notre article, certaines notions et propositions de mesures de complexités que nous considérons comme trop liées à un domaine particulier, mal fondées ou conduisant à des impasses n'ont pas été mentionnées du tout. Sans pouvoir prétendre ici à l'exhaustivité (impossible sur ce sujet) nous indiquons quelques références concernant ces notions et autres propositions de mesure de complexité [2, 3, 23, 36].

[1] B. Adamczewski, Y. Bugeaud, On the Complexity of Algebraic Numbers I. Expansions in Integer Bases, *Annals of Mathematics, Second Series*, 165(2), 547–565, 2007.

[2] C. Adami, N. Cerf, Physical Complexity of Symbolic Sequences, *Physica D* 137, 62—69, 2000.

[3] C. Adami, Sequence Complexity in Darwinian Evolution, *Complexity* 8(2), 49–56, 2003.

[4] P. Adriaans, Facticity as the Amount of Self-Descriptive Information in a Data Set, arxiv.org/abs/1203.2245 , 2012.

[5] R. Ahlswede, L. Khachatrian, C. Mauduit, A. Sárközy, A Complexity Measure for Families of Binary Sequences, *Periodica Mathematica Hungarica* 46(2), 107–118, 2003.

[6] L. Antunes, L. Fortnow, D. van Melkebeek, N. Vinodchandran, Computational Depth : Concept and Applications, *Theor. Comput. Sci.* 354(3), 391-–404, 2006.

[7] L. Antunes, A. Matos, A. Souto, P. Vitányi, Depth as Randomness Deficiency, *Theory Comput. Syst.* 45, 724–739, 2009.

[8] N. Ay, M. Müller, A. Szkola, Effective Complexity and its Relation to Logical Depth, *IEEE Trans. Inform. Theory* 56, 4593–4607, 2010.

[9] G. Baptiste, Réseau de régulation génique chez Escherichia coli, Thèse Université de Grenoble, 2012.

[10] S. Belabbes, G. Richard, On Using SVM and Kolmogorov Complexity for Spam, Filtering Proceedings of the Twenty-First International FLAIRS, 2008.

[11] C. Bennett, Logical Depth and Physical Complexity, In : *A half-Century Survey on the Universal Turing Machine*, 227-257, Oxford University Press, Inc., New York, 1988.

[12] C. Bennett, How to Define Complexity in Physics, and Why, in W. H. Zurek (ed.), *Complexity, Entropy, and the Physics of Information*, 1991.

[13] C. Bennett, What Increases When a Self-organizing System Organizes Itself ? Logical Depth to the Rescue, http ://dabacon.org/pontiff/ ?p=5912, 2012.

[14] R. Cilibrasi , P. Vitányi, Clustering by compression, *IEEE Transactions on Information Theory*, 2005.

[15] J. Collier, Information in Biological Systems, *Handbook of Philosophy of Science*, V8, Philosophy of Information, eds. Pieter Adriaans and Johan van Benthem, Dordrecht, Elsevier, 763-787, 2008.

[16] T. Cover, J. Thomas, *Elements of Information Theory*, Wiley-Interscience, 2006.

[17] G. Chaitin, *Algorithmic Information Theory*, Cambridge, Cambridge University Press, 1987.

[18] A. Danchin, *La Barque de Delphes. Ce que révèle le texte des génomes*, Ed. Odile Jacob, Paris, 1998.

[19] J.-P. Delahaye, *Information, complexité et hasard*, Hermès, Paris, 1994, 1999.

[20] J.-P. Delahaye, *Complexité aléatoire et complexité organisée*, Editions Quae, 2009.

[21] J.-P. Delahaye, H. Zenil, Numerical Evaluation of Algorithmic Complexity for Short Strings : A Glance into the Innermost Structure of Randomness, *Applied Mathematics and Computation* 219, 63–77, 2012.

[22] D. Doty, P Moser, *Feasible Depth*, *Computation and Logic in the Real World*, 2007.

[23] D. Feldman, J. Crutchfield, Measures of Statistical Complexity : Why ?, *Physics Letters A* 238, 244–252, 1998.

[24] M. Gell-Mann, S. Lloyd, Information Measures, Effective Complexity, and Total information, *Complexity* 2, 44-–52, 1996.

[25] J. Gibbons, Unbounded Spigot Algorithms for the Digits of Pi, *The American Mathematical Monthly* 113(4), 318–328, 2006.

[26] P. Grassberger, Randomness, Information, and Complexity, arxiv.org/abs/1208.3459, 2012.

[27] P. Grunwald, P. Vitányi, Shannon Information and Kolmogorov Complexity, arXiv :cs/0410002v1, 2004.

[28] D. Juedes, J. Lathrop, J. Lutz, Computational Depth and Reducibility, *Theor. Comput. Sci.* 132, 37-–70, 1994.

[29] A. Kaltchenko, Algorithms for Estimating Information Distance with Application to Bioinformatics and Linguistics, *Electrical and Computer Engineering*, 2004.

[30] A. Kolmogorov, Three Approaches to the Definition of the Concept Quantity of information, *Probl. Peredachi Inf.* 1(1), 3-–11, 1965.

[31] M. Koppel, Complexity, Depth, and Sophistication, *Complexity* 1, 1087–1091, 1987.

[32] M. Koppel, Structure, In : *A half-Century Survey on the Universal Turing Machine*, 235–252, Oxford University Press, Inc., New York, 1988.

[33] M. Koppel, H. Atlan, An Almost Machine-independent Theory of Program-length Complexity, Sophistication, and Induction, *Information Sciences* 56(1), 23–33, 1991.

[34] J. Lathrop, J. Lutz, Recursive Computational Depth, *Information and Computation* 153(2), 139-–172, 1999.

[35] M. Li, P. Vitanyi, *An Introduction to Kolmogorov Complexity and Its Applications*, Third Edition, Springer Verlag, 2008.

[36] S. Lloyd, H. Pagels, Complexity as Thermodynamic Depth, *Annals of Physics* 188, 186–213, 1988.

[37] J. McAllister, Effective Complexity as a Measure of Information Content, *Philosophy of Science* 70, 302–307, 2003.

[38] M. Mitchell, *Complexity Guided Tour*, Oxford University Press, Chapitre 7, 2009.

[39] H. Reeves, *Dernières nouvelles du cosmos*, Éditions du Seuil, Paris, 2002.

[40] A. Rukhin et al., A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications, *NIST Special Publication* 800(22), 2010.

[41] C. Shannon, W. Weaver, *The Mathematical Theory of Communications*, Univ. of Illinois Press, Urbana, 1949.

[42] F. Soler-Toscano, H. Zenil, J.-P. Delahaye, N. Gauvrit, Correspondence and Independence of Numerical Evaluations of Algorithmic Information Measures, arxiv.org/abs/1211.4891, 2012.

[43] A. Teixeira, A. Matos, A. Souto, L. Antunes, Entropy Measures vs. Kolmogorov Complexity, *Entropy* 13(3), 595–611, 2011.

[44] J.-S. Varré, E. Rivals, J.-P. Delahaye, The Transformation Distance : a Dissimilarity Measure Based on Movements of Segments, *Bioinformatics* 15(3), 194–202, 1999.

[45] H. Zenil, J.-P. Delahaye, C. Gaucherel, Image Characterization and Classification by Physical Complexity, *Complexity* 17(3), 26–42, 2012.

Des algorithmes et du naturel...

Entretien avec Bernard Chazelle (*réalisé par Valérie Schafer*)

Pour inaugurer cette rubrique dédiée à l'histoire de l'informatique, nous vous proposons un entretien avec Bernard Chazelle, tant est vraie cette phrase de l'historien Nathan Ensmenger : « The future of the history of computing is not machines, but people » [1]. *Chercheur passionné et engagé, pionnier dans la recherche en géométrie algorithmique, installé depuis 1977 aux États-Unis, Bernard Chazelle a accepté de revenir sur son parcours, son expérience américaine, sa présence cette année au Collège de France, ses recherches actuelles sur les algorithmes naturels, tout en recontextualisant son expérience et ses travaux dans le temps court de ces quarante dernières années et le temps long de l'histoire des sciences.*

Valérie Schafer [2] et Benjamin Thierry [3]

Valérie Schafer : *J'aimerais, si vous le voulez bien, que nous commencions par revenir sur votre parcours scientifique et professionnel.*

Bernard Chazelle : Après les Mines de Paris, je suis parti faire un doctorat en informatique aux États-Unis. Pourquoi l'informatique ? J'aimais les mathématiques, mais je voulais aussi partir de France pour découvrir quelque chose d'autre. L'informatique était la discipline qui se prêtait le mieux à un voyage aux États-Unis, car la France était un peu en retard dans ce domaine. Dans d'autres domaines des mathématiques, il y avait moins de raisons de s'exiler. Je suis parti en 1977. J'ai fait mon service militaire en coopération là-bas et je suis resté. Je suis revenu toutefois à trois

1. ENSMENGER N., "Power to the people : Toward a social history of computing", *IEEE Annals of the History of Computing*, 2004, 26 (1), p. 95.

2. Valérie Schafer, CNRS, ISCC.

3. Benjamin Thierry, Université Paris-Sorbonne.

reprises en France, pendant une année à chaque fois, en 1985, en 1998, et maintenant. C'est la troisième année que je passe en France depuis 1977. En 1985 j'étais à l'École Normale Supérieure et à Orsay, en 1998 à Polytechnique et actuellement au Collège de France.

En partant aux États-Unis en 1977, je ne connaissais rien à l'informatique. À l'école des Mines je faisais des mathématiques appliquées. J'ai dû tout apprendre là-bas. J'ai choisi un sujet d'informatique théorique, j'ai commencé à faire des algorithmes. J'ai fait une thèse à l'université de Yale sur la côte Est avec David Dobkin. La géométrie algorithmique était un sujet tout à fait neuf. J'ai réalisé la deuxième thèse au monde dans ce domaine. J'étais au tout début de ce champ. Il y a toujours un côté très grisant à être au début de quelque chose. J'ai aussi eu la chance de faire des stages. Vous savez, aux États-Unis on fait souvent des stages dans d'autres laboratoires et je suis allé notamment dans la Silicon Valley chez Xerox. Ethernet venait juste d'être inventé, puis le concept du PC, la souris… En 1978 je jouais avec des Altos. La seule différence avec les ordinateurs de maintenant, c'était qu'ils étaient plus lents, avec moins de mémoire, mais il y avait déjà tous les concepts de base.

VS : *Vous avez fréquenté Robert Metcalfe*[4] *?*

BC : J'étais plus proche de Bob Taylor[5], une figure légendaire ! C'était le directeur du centre de Xerox. Ce n'était pas un technicien, mais il a eu le génie de mettre ensemble tous ces gens pour travailler sur des choses nouvelles. Cela a mal tourné pour Xerox, ils ont raté le coche. Ils se sont toujours vus comme une compagnie qui faisait des copieurs… La révolution technologique la plus importante du siècle s'est faite dans leurs laboratoires, mais ils ont laissé partir tout le monde ! J'y faisais des stages tous les ans. Quand tout ce monde-là a déménagé à DEC SRC (Systems Research Center), j'ai continué à faire des stages chez DEC.

C'était plus intéressant je crois à cette époque que maintenant. Les innovations actuelles palissent énormément, Facebook et les autres, face à ce qui a été imaginé alors.

La révolution n'était pas seulement technologique, elle était plutôt du côté social, par exemple avec l'invention du Web au CERN. L'idée que des gens passeraient des heures à peaufiner des pages Web… c'est un phénomène sociologique déroutant. C'est très difficile de prévoir que de tels concepts, formant un cocktail d'altruisme et de narcissisme, puissent marcher. De nos jours, le développement de Facebook

4. C'est au Xerox Parc qu'en 1973 Robert Metcalfe co-invente Ethernet, devenu un standard pour les LANs (réseaux locaux). Il quitte le Parc en 1979 pour fonder l'entreprise 3Com.

5. Après avoir été de 1966 à 1969 directeur de l'IPTO (*Information Processing Techniques Office*, à l'origine du lancement d'ARPANET) à l'ARPA (*Advanced Research Projects Agency*), Robert Taylor est au début des années 1970 à l'origine du *Computer Science Laboratory* du *Xerox Palo Alto Research Center*, puis *Founding Director* du *Digital Equipment Corporation's System Research Center* (SRC) de Palo Alto, en Californie, créé en 1984.

c'est assez prévisible, pas très innovant, alors que le Web est révolutionnaire, pas au sens technique mais au niveau social. Et c'est un sujet de grand étonnement : j'aurais pensé que les gens seraient plus soucieux de leur vie privée, que les pages Web seraient surtout des cartes de visite. Que Wikipedia soit d'une telle qualité, qu'un système comme celui-ci totalement *open* et décentralisé — ou presque —, arrive à fournir une information, alors qu'on imaginerait facilement que la tendance naturelle soit à un grand fourbi… Dans le domaine mathématique, quand on cherche une formule compliquée, on la trouve sur Wikipedia et elle est exacte… Mon étonnement, il est humain : dans un système où tout doit être monnayé, on découvre sur internet un volontariat de très haute qualité.

Quand le fondateur d'IBM a dit qu'il n'y avait un besoin que de cinq ou six ordinateurs dans le monde entier, personne ne lui a ri au nez. Un petit nombre d'ordinateurs comme on a un petit nombre de sous-marins atomiques… Et penser que c'est rentré dans le quotidien de tout le monde… De voir mes copains faire ça, c'était grisant, le transfert de cette technologie d'un petit monde d'élite à la société.

> « "*I think there is a world market for maybe five computers*", averred IBM Chairman, Thomas Watson, a gem of prescience matched only by a 1939 *New York Times* editorial : "*The problem with television is that people must sit and keep their eyes glued to the screen ; the average American family hasn't time for it.*". The great demographer Thomas Malthus owes much of his fame to his loopy prediction that exponentially increasing populations would soon outrun the food supply. As the apprentice soothsayer learns in "Crystal Gazing 101", never predict a geometric growth ! ».
>
> CHAZELLE B., "The Algorithm : Idiom of Modern Science"
> http ://www.cs.princeton.edu/ chazelle/pubs/algorithm.html

Chez Xerox les centres de recherche étaient très ouverts. On pouvait plus ou moins faire ce qu'on voulait. Il y avait des gens comme moi qui n'y faisaient que de la théorie. Personne ne nous demandait si c'était utile, si ça rapporterait de l'argent. Si on publiait, c'était très bien. Pour eux il y avait une idée très holistique de ce qui se passait. J'étais là pour travailler de façon théorique. Et je pouvais utiliser leur matériel…

J'ai fait ma thèse en trois ans, puis j'ai passé deux ans en post-doc, tout en faisant aussi mon service militaire en coopération, à Carnegie-Mellon : ça a été une très bonne idée. Il y avait un autre Français sur place, Louis Monier, qui a co-inventé AltaVista. J'ai fait des VLSI (*Very Large Scale Integration*), des circuits intégrés, mais mes recherches ont aussi toujours eu un accent théorique. Je suis revenu ensuite à la théorie de la complexité. Après Carnegie-Mellon, je suis allé trois ans à Brown

sur la côte Est (Providence). J'étais *assistant professor*, avant de gagner ma *tenure* qui donne un poste permanent [6].

En 1985, comme je vous l'ai dit, je passe un an en France à l'École Normale Supérieure et à Orsay, puis je ne reviens jamais à Brown, car Princeton m'embauche. Aux États-Unis, il n'y a pas, contrairement à la France, de cumul possible.

VS : *Princeton vous approche à quel moment ?*

BC : Ils me contactent quand je suis en France. Nous nous connaissions tous plus ou moins dans ce milieu assez étroit.

Princeton pour moi est le meilleur endroit au monde en informatique théorique, c'est un peu la Mecque de la discipline, la plaque tournante. Il y a l'université et l'Institut d'étude avancée. Dans la région il y a également d'autres gros laboratoires, New York est proche, il y a une densité de recherche élevée. Je parle du domaine de la théorie, pas d'autres disciplines de l'informatique.

VS : *Dans votre discipline le travail se fait en équipe ?*

BC : Oui, j'aime aussi parfois travailler seul, mais j'aime travailler en équipe. J'ai des étudiants en doctorat notamment.

VS : *Pourquoi avoir choisi une carrière aux États-Unis ? Evidemment ce que vous venez de m'expliquer répond déjà partiellement à la question...*

BC : L'informatique théorique veut dire deux choses différentes en Europe et aux États-Unis.

La discipline américaine de l'informatique théorique est assez peu pratiquée en France, même s'il y a quelques talents individuels très grands, notamment à l'École Normale, mais un pays comme Israël est bien plus fort en informatique théorique que la France. Il y a une autre informatique théorique où la France est très forte, mais ce n'est pas la même discipline.

La différence peut s'expliquer de plusieurs façons : en France, on regarde l'informatique par le biais des langages de programmation. La France est très forte sur l'invention de langages de programmation. C'est important en vérification. Mais on n'appelle pas cela de l'informatique théorique. Cependant il y a un autre aspect : on peut exprimer un langage informatique sous une forme algébrique. On peut aborder le problème des langages sous un aspect algébrique et combinatoire. Une école française a commencé dans les années 1950 sur ce terrain, derrière Schützenberger et d'autres chercheurs. L'informatique théorique en France c'est surtout cela, elle s'intéresse surtout au comportement et à la structure des langages.

6. Aux États-Unis ou au Canada, le système de la *tenure* offre une sécurité d'emploi permanent dans l'enseignement supérieur (par opposition à des renouvellements périodiques) et garantit une liberté d'expression des professeurs, qui les protège des pressions extérieures.

Les langages n'intéressent pas tellement les informaticiens aux États-Unis. Ils laissent davantage cela aux ingénieurs qui font de la conception. Il y a une coupure fondamentale entre les deux disciplines.

Par contre une de mes spécialités est la géométrie algorithmique. C'est une exception à la règle que je vous ai énoncée, où en France et aux États-Unis on transcende les frontières, c'est international. Je peux aller à Sophia-Antipolis et trouver toute une communauté de Français et d'Allemands avec lesquels on parle le même langage. Une autre exception est la cryptographie.

VS : *La géométrie algorithmique est votre spécialité ?*

BC : Oui, mais je passe l'essentiel de mon temps actuellement sur les algorithmes naturels… J'ai d'autres centres d'intérêts de plus en plus prenants. Ce n'est pas une distance intellectuelle, mais je n'ai plus le temps de m'y plonger autant.

VS : *Les algorithmes naturels, c'est donc ce qui vous occupe et vous préoccupe en ce moment ?*

BC : Effectivement, c'est ce qui m'occupe et me préoccupe le plus. Les gens écrivent des algorithmes comme ils conçoivent des ponts, un peu comme des ingénieurs. La différence entre des scientifiques et des ingénieurs ? Les seconds ont un problème concret, un objectif précis. La question est de construire quelque chose qui leur permette de répondre à un objectif, le but est d'accomplir cette chose. Et si quelqu'un trouve une méthode plus efficace, on va jeter la précédente. En science on ne peut pas faire cela. Le but est d'étudier les choses comme elles sont, pas comme on veut les faire.

Est-ce que les algorithmes sont comme des ponts ? Ou au contraire la nature nous propose-t-elle des algorithmes et nous n'avons pas le choix ? Oui, depuis plus d'un milliard d'années dans la vie il y a des algorithmes. Certains diront : "Quel est l'intérêt d'appeler cela des algorithmes ?". Avant on disait des systèmes dynamiques, etc. Mais, l'avantage de changer de nom, c'est qu'on change de point de vue.

Dans la physique, l'humanité a eu un succès énorme, c'est fulgurant. La physique est un monde de symétrie, de miroir, et on a un langage pour étudier ces symétries : les mathématiques. On a un langage intellectuel qui nous donne des outils parfaits pour faire de la physique. Les expériences sont indispensables, mais sans ces mathématiques extrêmement puissantes, on n'aurait pas abouti. Est-ce que ce succès peut se répéter en biologie, dans les sciences un peu moins physiques ? Pourquoi pas. Certains disent qu'après tout, tout est une molécule. D'autres, comme moi, disent que les lois de la physique sont opérantes, mais qu'elles ne servent pas à grand chose dans ce cas, que c'est regarder à une mauvaise échelle.

La différence entre la biologie et la physique, entre les cellules et le système solaire, c'est l'absence de symétrie. Toutes les symétries naturelles de la physique ont été brisées par l'histoire, ce qui donne la complexité de la biologie.

Si vous me donnez une page blanche, je peux écrire des équations qui expliqueraient 99% de ce qui nous entoure dans cette pièce, sauf vous et moi. En biologie, personne de sérieux ne croit qu'on va faire tenir sur une page des équations miraculeuses qui permettront de guérir le cancer, de tout expliquer… C'est la différence entre un haiku et *Crime et Châtiment*. La biologie c'est comme ça, il y a une complexité descriptive.

> « Le prix Nobel de physique, Eugène Wigner, s'émerveillait de "l'efficacité déraisonnable des mathématiques dans les sciences de la nature". Sans dénier l'intérêt épistémologique de ce propos, il convient de le reformuler. Wigner avait à l'esprit les sciences physiques. En guise de réponse, le mathématicien Israel Gelfand ironisa que la seule chose plus étonnante que l'efficacité des mathématiques en physique, c'est leur inefficacité en biologie. Excès de langage, soit, mais nul n'entretient l'illusion que guérir le cancer ou élucider le cortex cérébral puisse se réduire à la solution de quelques équations différentielles. La raison en est simple. L'univers physique est bâti sur des principes de symétrie et d'invariance — dont les liens s'expriment dans un théorème classique d'Emmy Noether — qui font cruellement défaut en biologie. Si l'on se réfère au schéma :
>
> Biologie = Physique + Histoire,
>
> on se heurte à la dure réalité que l'histoire est le grand briseur de symétrie. Il y a une distance considérable entre la régularité d'horloge du système solaire et les mystères du système immunitaire. »
>
> CHAZELLE B., *L'algorithmique et les sciences*, Paris, Collège de France/Fayard, coll. "Les leçons inaugurales", no. 229, 2013, pp. 76-77.

Alors a-t-on des outils conceptuels pour comprendre ce qui se passe ? Voire prédire ? Je pense qu'une des erreurs à ne pas commettre est d'être obnubilé par la physique et de penser qu'on peut appliquer les mathématiques classiques, tout réduire à des équations différentielles. On peut obtenir des résultats pour des choses simples, par exemple pour certains aspects de la morphogénèse (travaux de Turing). Mais, un processus algorithmique il faut le voir comme une pièce de théâtre, une complexité inhérente… il faut l'aborder sous l'angle d'un récit historique avec une complexité inhérente. Les mathématiques classiques ne permettent pas cela.

Il est évident qu'on ne peut pas réduire *Crime et Châtiment* à des équations mathématiques ou des statistiques sur la fréquence des mots. C'est exactement pour cette même raison que les équations mathématiques ne peuvent pas expliquer l'immunologie.

VS : *Quel est historiquement le rapport qu'entretiennent les mathématiques avec la biologie ?*

BC : Au XIX^{e} siècle, il est quasiment nul. Darwin, Mendel n'ont pas besoin de mathématiques. Au XX^{e} siècle les choses changent un peu. Il y a utilisation des probabilités et des équations différentielles pour l'étude de la dynamique de population par exemple. On peut écrire des équations pour faire quelques prédictions. Ce sont des mathématiques relativement simples. Cela a quelques succès, mais comparé à la physique du XX^{e} siècle, c'est le jour et la nuit, alors qu'au même moment les mathématiques changent tout à la physique.

Il y a donc au XX^{e} siècle des mathématiques en biologie, mais peu. Par exemple les grandes découvertes en biologie telle que la structure tridimensionnelle de l'ADN, cela demande très peu de maths.

La biologie est un sujet bien plus vaste et difficile que la physique. Je ne dis pas qu'il n'y a pas des pans de la biologie qui se prêtent aux mathématiques, mais ils sont plus proches de la chimie peut-être. Par contre dans les réseaux, le caractère algorithmique est important.

Il faut s'entendre sur le mot réseau, on pense souvent à des choses statiques, le réseau d'autoroute, le réseau téléphonique, mais il y a des réseaux qui changent tout le temps. Il faut élargir la définition d'un réseau. Prenons un monde où on commence par les nœuds du réseau, les agents, c'est-à-dire quelqu'un qui a une capacité de cognition et de communication. Cela peut être vous sur Facebook, cela peut être les oiseaux et les poissons, qui communiquent avec un nombre restreint de voisins. Les capacités cognitives employées par les agents du système ne sont pas énormes, on a un réseau, mais dynamique au sens plein. Vous regardez des bancs de poissons quand il y a des prédateurs, ce dynamisme est très important, on le voit dans les bactéries, etc. On peut modéliser des tas de choses, même très complexes, des relations d'influence, des mouvements de foule. Il y a généralement des processus cognitifs simples : ainsi en voiture, si le véhicule devant vous avance, vous avancez. La modélisation typique est de trouver des modèles d'agents qui changent tout le temps. On peut les simuler, visualiser ce qui se passe. Mais, la science c'est aller au-delà : la question est de savoir si, une fois que l'on a ces modèles, on va trouver des outils conceptuels qui nous permettent de faire des prédictions. Actuellement on essaie de calquer des équations. Il faut changer de paradigme, arrêter d'essayer de calquer des théories anciennes.

Donnons un exemple concret en dehors de la biologie : Facebook, les blogs. On a regardé comment certaines rumeurs se propagent, certaines modes se créent. Pour les épidémies en médecine, c'est intéressant aussi. Si on a seulement quelques ressources de vaccins, on va essayer de comprendre qui on va vacciner, etc. Il faut expérimenter, créer un modèle, un algorithme, qui tienne compte des comportements différents. Il faut modéliser tout cela. Mais, ce n'est pas ce qui m'intéresse directement. Pour les épidémies, la modélisation va être faite par des épidémiologistes, elle requiert des spécialistes de l'application. Une fois qu'on a cette modélisation, on la

simule, on la visualise, mais cela n'explique rien. Il faut revenir à cet algorithme et chercher à le comprendre.

> « On rapporte du grand physicien danois, Niels Bohr, cette anecdote peut-être apocryphe :
> — Professeur Bohr, je vois que vous avez un fer à cheval accroché au mur. Ne me dites pas que vous croyez à ce genre de choses !
> — Rassurez-vous, je n'y crois pas du tout, mais on m'a dit que ça marche même quand on n'y croit pas.
> Ainsi en va-t-il de la révolution algorithmique. Au-delà du scepticisme ou de l'engouement du jour pour la dernière nouveauté informatique se cache un de ces changements de paradigme chers à Kuhn. Outil conceptuel subversif, l'algorithme ouvre la possibilité d'un regard nouveau sur les sciences et les technologies qui s'étend bien au-delà de ses applications pratiques. »
>
> CHAZELLE B., *L'algorithmique et les sciences*, Paris, Collège de France/Fayard, coll. "Les leçons inaugurales", no. 229, 2013, p. 17.

VS : *À quels types de réseaux pensez-vous ? Les réseaux neuronaux ?*

BC : À mon avis il y a des réseaux plus complexes que d'autres. Dans la neurobiologie, où on trouve les réseaux les plus complexes de la nature, c'est là où je suis le moins optimiste. Alors soyons réalistes, essayons déjà de résoudre des choses plus "simples", comme les bactéries. Sur les aspects neuronaux il est en outre très difficile de faire des expériences. Mais parfois, en posant des questions simples, on ouvre des portes énormes sur d'autres problèmes. L'enjeu actuel est de construire des outils intellectuels qui nous manquent. Maintenant il faut accorder une marge de temps pour développer des outils conceptuels.

Pour comprendre le système solaire, Poincaré a inventé toute une discipline mathématique. À mon avis, il faut avoir cette disposition, ne pas être trop obnubilé par les réponses à des problèmes concrets. C'est un mal de notre temps, l'application immédiate. Il faut être un peu plus humble, modeste.

VS : *Vous travaillez en interdisciplinarité ?*

BC : Oui, absolument. La raison pour laquelle je repars aux États-Unis est que je vais travailler avec des biologistes et physiciens sur des problèmes concrets à IAS (*Institute for Advanced Study*).

C'est pour cela qu'on va avoir une redéfinition des disciplines. La nature de la discipline va complètement changer. Une expérience qui m'a beaucoup marqué, il y a quelques années, est d'avoir fait un cours pluridisciplinaire : l'idée était d'enseigner des bases scientifiques ensemble, en un seul cours s'étendant sur un an. Quand on étudie un phénomène physique, la diffusion, il y a l'aspect physique, mathématique,

chimique, etc. L'algorithme de Google est un algorithme de diffusion. On enseigne aux étudiants que le sucre qui se répand dans votre sang et l'algorithme de Google, c'est intellectuellement à peu près la même chose… c'est une perspective différente sur l'enseignement. Je ne dis pas que tout doit changer. La relativité générale par exemple n'a pas besoin de cela, mais la biologie est de plus en plus une science de l'information, et la science de l'information c'est l'informatique. Je ne dis pas que l'une va remplacer l'autre, mais dans quinze ans ces disciplines auront tellement changé… Alors qu'un département de mathématiques où ils font de la théorie des nombres, dans quinze ans les chercheurs auront fait de grands progrès, mais ils travailleront sûrement d'une manière assez semblable à aujourd'hui.

"First you prove it, then you let it sink in."
Dessin de Bernard Chazelle (sous licence CC-BY-NC-ND)

VS : *À quand remonte la notion d'algorithmes naturels ?*

BC : Les algorithmes naturels, c'est moi qui ai suggéré cette phraséologie. La formule a été utilisée avant moi mais avec un sens différent.

Il y a une démarche inverse très intéressante, mais je n'y travaille pas, c'est le biomimétisme. On regarde comment les fourmis font, afin de s'inspirer du monde naturel pour résoudre des problèmes concrets. On fait des algorithmes en piochant dans la nature. Parfois on appelle cela des algorithmes naturels. Mais moi ce n'est pas le sens que je leur donne. Je réserve ce sens aux algorithmes qu'on trouve dans la nature et qui ont évolué pendant des millions d'années.

Le moteur de recherche de Google marche bien, il a été mis au point pendant dix ans par une centaine d'ingénieurs et s'ils travaillent mal, ils sont virés… les algorithmes naturels ce ne sont pas dix ans mais plusieurs millions d'années, plusieurs milliards d'ingénieurs, et s'ils travaillent mal, ils meurent ! Ce que je veux dire par

cette boutade, c'est que quand on regarde les oiseaux, les fourmis, on a des algorithmes bien plus impressionnants que ceux que nous pouvons inventer.

Revenons aux fourmis… Celui qui fait du biomimétisme, il n'en a rien à faire des fourmis. Il s'inspire des fourmis, pour résoudre un problème, alors que moi ce sont les fourmis qui m'intéressent. Donc le but et la démarche sont différents. Je veux que cela respecte la démarche des fourmis le plus possible, je ne vais pas aider les fourmis à mieux fonctionner.

VS : *Sur votre page de Princeton on trouve aussi des articles plus personnels, engagés, cela m'a étonnée. Je pense notamment aux articles contre la torture, sur la guerre en Irak…*

BC : La signification historique de la *tenure* est très particulière. Cela vous donne le droit absolu à la liberté d'expression. L'université ne peut pas vous causer le moindre problème. Elle ne peut pas dire que cette page est professionnelle et que vous ne pouvez pas exprimer vos idées. Ce n'est pas le cas dans une entreprise, alors que dans les universités américaines, cette ligne de démarcation n'existe pas. Par contre comme c'est public, des gens peuvent ne pas être d'accord. Mais c'est vrai partout. Les courriers que je reçois dans ma profession scientifique sont toujours aimables. Quand vous vous engagez sur un terrain politique, ces règles ne sont pas toujours respectées. J'ai l'impression d'avoir un statut privilégié et une dette vis-à-vis de la société. L'idée que je ne suis payé que pour faire des algorithmes, ce n'est pas ma vision du monde intellectuel. La spécialisation est récente. Au XIX^{e} siècle les scientifiques avaient une culture énorme. Prenez les grands physiciens, Einstein par exemple, ils ne se gênaient pas pour exprimer des opinions. On a une plateforme que la plupart des gens n'ont pas. Je me sens aussi une obligation d'exprimer mes opinions.

VS : *Dans un article lié à un cours à destination de lycéens, vous étayez votre propos de nombreuses références à l'histoire. Connaître l'histoire de sa discipline, est-ce important ?*

BC : Oui, d'autant que ce n'est pas très compliqué. Cela commence véritablement dans les années 1930. Cette histoire est importante. Prenez Alan Turing. Il se pose des questions en Intelligence Artificielle et morphogénèse qui sont maintenant complètement d'actualité.

Il a posé la question de savoir à quel point le monde vivant peut être expliqué par des algorithmes. C'est lui le fondateur des algorithmes naturels ! Pour lui, l'ordinateur n'est pas qu'un calculateur. Sa vision s'est perdue ensuite. Sous Turing, c'était extrêmement large. Il a aussi compris très tôt que ce n'était pas une branche classique des mathématiques, que ce qu'il faisait était quelque chose de différent, alors qu'à l'époque certains inventaient une nouvelle mathématique, mais c'était toujours des maths. Turing a créé une autre espèce. Lui et toute une communauté. Il représente bien l'esprit universel de l'informatique… et aussi le fait qu'avant Turing on a fait

beaucoup de robots, automatisé les tâches, mais c'est le caractère universel qui a tout changé, ne pas vouloir faire un robot mieux que les autres, mais le robot universel… c'est cela à mon avis le grand saut intellectuel. L'idée de code, d'algorithmes codés de façon symbolique, une idée qu'on retrouve dans nos gènes et les protéines qui les codent, voilà un des concepts les plus révolutionnaires du XX^{e} siècle, au même titre que la relativité et la mécanique quantique.

Pour aller plus loin...

Page de Bernard Chazelle : http ://www.cs.princeton.edu/ chazelle/

Introduction aux algorithmes naturels :

CHAZELLE B., "Natural Algorithms and Influence Systems", *Research Highlights*, CACM 2012. http ://www.cs.princeton.edu/ chazelle/pubs/cacm12web.pdf

L'article de Bernard Chazelle à destination des lycéens :

CHAZELLE B., "The Algorithm : Idiom of Modern Science"
http ://www.cs.princeton.edu/ chazelle/pubs/algorithm.html

La leçon inaugurale de Bernard Chazelle au Collège de France :

CHAZELLE B., *L'algorithmique et les sciences*, Paris, Collège de France/Fayard, coll. "Les leçons inaugurales", no. 229, 2013, 104 p.

Sur Alan Turing :

HODGES *A. Alan Turing : The Enigma. The Centenary Edition*, Princeton University Press, 2012, 632 p.

Voir sur le site Interstices le film de Catherine Bernstein "Le modèle Turing"
http ://interstices.info/jcms/int_67976/le-modele-turing

Plusieurs pages du site du CNRS sont dédiées à Turing. Voir notamment :
Présentation de l'année Turing, http ://www.cnrs.fr/ins2i/spip.php ?article201
Exposition "Alan Turing, Du langage formel aux formes vivantes",
http ://www.cnrs.fr/ins2i/spip.php ?article365

Sur Xerox et le Parc :

Parc History, http ://www.parc.xerox.com/about/history/default.html

HILTZIK M. A., *Dealers of Lightning : Xerox PARC and the Dawn of the Computer Age*, Harper Business, New York, 1999, 448 p.

JOHNSON J., ROBERTS T., "The Xerox Star : a retrospective", *IEEE Computer*, vol. 22, 1989, p. 11-29.

L'informatique, outil et objet d'enseignement

Entretien avec Jean-Pierre Archambault,
président de l'EPI (*réalisé par Benjamin Thierry*)

À l'heure des débats autour de la (ré)apparition de l'informatique comme discipline scolaire en Terminale S, il nous a paru nécessaire de revenir sur les enjeux que pose cette question au travers du parcours d'un acteur important de l'informatique éducative et de l'enseignement de l'informatique.

C'est en effet sur ces deux aspects que nous avons souhaité interroger Jean-Pierre Archambault pour mieux cerner à la fois l'histoire des rapports de l'École et de l'informatique, mais également le rôle que le numérique doit aujourd'hui jouer dans la constitution d'une culture citoyenne à la mesure de la place qu'il occupe dans la société.

Jean-Pierre Archambault est agrégé de mathématiques. Formateur en informatique au début des années 1980, c'est un pionnier des TICE[1] *qui a coordonné le pôle de compétences "logiciels libres" du SCEREN*[2] *et préside l'association EPI*[3] *depuis 2007. À ce titre il a joué un rôle important dans la création en Terminale S à la rentrée 2012 de l'enseignement de spécialité optionnel "Informatique et sciences du numérique".*

Benjamin Thierry[4] et Valérie Schafer[5]

1. Technologies de l'information et de la communication pour l'enseignement.
2. Services Culture, Éditions, Ressources pour l'Éducation Nationale (http ://www2.cndp.fr/cndp_reseau/presentation/accueil.htm).
3. Association Enseignement Public & Informatique (http ://www.epi.asso.fr).
4. Benjamin Thierry, Université Paris-Sorbonne.
5. Valérie Schafer, CNRS, ISCC.

La découverte de l'informatique comme outil d'enseignement

Benjamin Thierry : *Ma première question consistera à préciser un peu votre parcours. Comment un agrégé de mathématiques se tourne-t-il vers l'informatique au début des années 1980 ?*

Jean-Pierre Archambault : 1981 a vu la relance des formations "lourdes" à l'informatique pour les enseignants du secondaire. J'ai postulé à cette formation et ainsi bénéficié d'un stage organisé à l'ÉNS Ulm. C'est dans ce cadre que j'ai découvert l'informatique et les applications de l'ordinateur à l'École.

Ce stage durait un an. Il était à plein temps. Nous étions donc complètement déchargés de cours. Autre époque dont on peut être quelque peu nostalgique... L'objectif était de former des enseignants qui allaient accompagner l'arrivée d'ordinateurs dans les lycées en assurant notamment des formations de 100 heures. Les années 1970 avaient déjà vu se mettre en place de tels stages, mais ils avaient été interrompus en 1976 avant qu'ils ne reprennent donc en 1981.

BT : *Ce stage était votre premier contact avec l'informatique ?*

JPA : Nous étions quasiment tous néophytes. Je n'avais personnellement jamais eu aucun contact avec l'informatique avant ce stage. Néanmoins, à la fin des années 1970, je voyais apparaître des calculettes programmables sur les tables de mes élèves. Cela avait éveillé en moi l'intérêt pour ces machines dont je pressentais qu'elles annonçaient des évolutions profondes dans l'enseignement.

BT : *Le soutien de la rue de Grenelle est donc total envers cette initiative...*

JPA : On doit parler plus que de soutien puisqu'il s'agissait d'une initiative ministérielle : la volonté politique était de faire émerger des compétences pour l'utilisation de nouveaux outils pédagogiques.

BT : *En quoi consistait la formation durant ce stage ?*

JPA : Les contenus variaient beaucoup d'un centre de formation à l'autre. Au centre d'Ulm, il y avait beaucoup de programmation. On parlait également d'architecture machine, mais peu de réseau (c'était l'époque du poste autonome et Internet n'était pas ce qu'il est aujourd'hui) et peu de bases de données. La formation était particulièrement équilibrée : aux fondamentaux de la science informatique était associée la présentation de pistes possibles de l'utilisation en classe des ordinateurs et des premiers logiciels programmés par des collègues dans différentes disciplines. Je me souviens en particulier d'un logiciel de lexicologie qui m'avait marqué et qui permettait de mener une analyse lexicologique du *Dom Juan* de Molière. Il faut se souvenir qu'à l'époque, n'existait pas encore d'ordinateur à interface graphique et que tout cela demandait de gros efforts aux collègues qui souhaitaient s'impliquer.

BT : *Dans le cadre de l'enseignement des mathématiques sur quels outils pouvait-on compter à cette période ?*

JPA : Essentiellement des logiciels réalisés par l'INRP et l'EPI. Chose étonnante sur laquelle nous pourrons revenir, la diffusion de ces logiciels était gratuite et l'on avait accès au code source. Du libre avant la lettre même si l'on ne se posait pas du tout la question de la licence dans le cadre de la diffusion.

BT : *Vous avez été formateur, et puis...*

JPA : J'ai assuré des formations de 1982 à 1985 puis, avec le plan "Informatique pour tous", j'ai commencé à participer au pilotage du développement de l'informatique pédagogique. J'ai organisé les stages du plan "Informatique pour tous" dans l'académie de Créteil. 6000 enseignants en ont bénéficié. On a entendu dire que ce plan avait été un échec. Donner une cinquantaine d'heures de formation en informatique à 100 000 enseignants durant l'été et l'automne 1985 : des échecs comme cela, on aurait tendance à en redemander !

Le plan "Informatique pour tous" a fondamentalement constitué un signal indiquant que quelque chose d'important était en train de se produire et qu'il fallait s'y préparer. Les établissements étaient équipés en nanoréseaux, machines Thomson et compatibles PC. Il faut se souvenir qu'à l'époque le moindre compatible PC coûtait aux alentours de 25 000 francs et que pour ce prix on pouvait équiper toute une salle avec des machines Thomson. C'est aussi l'époque du LSE.

BT : *Et du Logo…*

JPA : Oui. Ce langage a été une véritable réussite dans l'enseignement primaire. Mais il a disparu des écoles à la fin des années 1980. Du côté de l'enseignement secondaire c'est à cette époque que l'on a vu arriver les premiers logiciels de simulation pour les sciences de la vie autour par exemple d'applications sur la nutrition, mais aussi les traitements de texte et les tableurs.

Le plan "Informatique pour tous" a comporté un volet télématique. Dans l'académie de Créteil, nous avons créé un bassin télématique regroupant 20 serveurs d'établissement. Parmi les usages pédagogiques figuraient les échanges scolaires, dans le cadre des classes transplantées (classes de neige...). C'était bien entendu beaucoup plus difficile de le faire avec l'étranger : il fallait pour cela s'appuyer sur des initiatives de France Télécom.

Il y avait le réseau Transpac, des serveurs, des terminaux Minitel mais, paradoxalement, dans certains cercles de l'informatique pédagogique la télématique n'était pas perçue comme relevant vraiment de l'informatique. Pourtant, les usages de la télématique scolaire ont préfiguré ceux d'Internet qui a pu s'appuyer dès ses débuts

sur une culture de communication électronique existante. On peut dire : "De la télématique à Internet : continuité/rupture"[6].

BT : *À cette époque et dans le cadre de votre pratique dans le secondaire, quels liens avez-vous avec le milieu de la recherche en informatique et les universitaires ?*

JPA : Quasiment aucun en dehors des stages longs qui étaient dirigés par des universitaires. En 1995 débutera l'expérimentation d'Internet. Je la piloterai pour les trois académies franciliennes. Nous serons hébergés, formés et accompagnés par le Centre de calcul et de recherche de Jussieu. Une aide précieuse et une coopération fructueuse.

BT : *Et puis ?*

JPA : Je suis ensuite allé au CNDP, à la direction de l'Ingénierie éducative. Puis j'ai été chargé de mission veille technologique au CNDP-CRDP de Paris, où j'ai eu la responsabilité du dossier des logiciels libres, coordonnant le pôle de compétences logiciels libres du SCEREN.

Sur le plan associatif, je suis devenu président de l'EPI en 2007 (association à laquelle j'avais adhéré en 1981 lors du stage lourd). L'EPI prône depuis sa création la complémentarité des approches. On sait le rôle qu'elle a joué dans la création en Terminale S à la rentrée 2012 de l'enseignement de spécialité optionnel "Informatique et sciences du numérique".

L'informatique comme discipline et comme élément de la culture commune

BT : *S'il faut résumer, qu'attend-t-on de l'informatique dans l'enseignement ? Quels rôles peut-elle y jouer ?*

JPA : On peut dégager quatre statuts éducatifs de l'informatique.

Elle est outil pédagogique, l'objectif étant de développer des usages raisonnés, pertinents et efficaces. L'informatique permet de faire mieux ou autrement, ou de faire ce que l'on ne pouvait pas faire auparavant. Si beaucoup a été fait, il reste beaucoup à faire. Ainsi le ministère de l'Éducation Nationale indiquait-il récemment que 21% des enseignants utilisaient l'informatique au moins une fois par semaine. Et les 79% restants ? Nous sommes en 2012 et les débuts de l'informatique pédagogique remontent à la décennie 70.

L'informatique est outil de travail personnel et collectif des enseignants, des élèves et de la communauté scolaire dans son ensemble. Cela va de la gestion de la paye des personnels et de la préparation de la rentrée scolaire à l'utilisation d'un

6. Référence à l'ouvrage de Jean-Pierre Archambault, *De la télématique à Internet : Guide des usages pédagogiques*, CNDP, Paris, 1996.

traitement de texte pour préparer ses cours en passant par celle d'Internet pour faire un exposé. Ce statut ne se confond pas avec le précédent, la meilleure preuve étant que, si la grande majorité des enseignants utilise l'ordinateur à domicile, pour autant ils l'utilisent peu ou pas en classe.

L'informatique est facteur d'évolution des disciplines enseignées, modifiant leurs objets, méthodes et outils, leur "essence". Cela se traduit dans leur enseignement. C'est particulièrement vrai pour les enseignements techniques et professionnels où l'informatique s'est banalisée depuis plus de vingt ans déjà. Le traitement de texte a supplanté la machine à écrire, la base de données le fichier carton et le logiciel de DAO la planche à dessin. Peu ou prou, toutes les disciplines sont concernées.

BT : *On comprend très bien quelles sont les contraintes qui s'exercent sur l'enseignement technique et professionnel, mais qu'en est-il pour les disciplines d'enseignement général ?*

JPA : Bien sûr, ces disciplines subissent moins de "contraintes". Les échéances de la culture générale sont plus lointaines. Mais si je prends l'exemple des sciences, on ne peut pas aujourd'hui enseigner la physique ou les sciences de la vie et de la Terre sans parler de la simulation ou de l'expérimentation assistée par ordinateur. Pour la raison simple précédemment évoquée : ces disciplines scientifiques ont vu leur "essence" modifiée par l'informatique. Leur enseignement ne peut que tenir compte de ce qui se passe du côté de la recherche.

BT : *Est-ce à dire qu'il y a des contenus disciplinaires différents qui mettent certaines disciplines "à l'abri" de l'introduction de l'informatique ? Je pense en particulier aux disciplines littéraires ou aux sciences sociales.*

JPA : Je disais "peu ou prou". Il serait néanmoins dommage de se priver des apports de l'informatique dans ces différentes matières. Je reviens sur le logiciel de lexicologie qui m'avait marqué durant mon stage lourd. Je me souviens que le collègue qui nous l'avait présenté avait indiqué que dans l'un des actes du *Dom Juan* de Molière, de mémoire le troisième, le mot "amour" était quasiment absent alors qu'on parlait beaucoup d'amour sans jamais en employer le terme. N'est-ce pas là un excellent moyen de commencer une analyse avec ses élèves ? L'outil informatique ne jette-t-il pas une lumière nouvelle sur des textes que l'on croit connaître ?

De même, dans le primaire on connaît maintenant tout l'intérêt de l'utilisation du traitement de texte comme outil pédagogique pour l'acquisition des compétences de rédaction. Écrire c'est réécrire. Le traitement de texte permet de corriger, modifier et améliorer les textes produits par les élèves en ne provoquant pas le découragement qu'impliquent souvent ces opérations quand elles sont faites "à la main".

Cela ne constitue nullement un quelconque plaidoyer pour un "tout informatique". Le calcul mental reste incontournable. Il faut donc des séances sans calculette et sans ordinateur.

BT : *Cela va à l'encontre de certaines idées dans l'air du temps qui feraient de l'ordinateur la porte d'entrée sur un vaste espace de stockage de connaissances qu'il ne serait plus nécessaire de maîtriser puisqu'elles sont à disposition immédiatement.*

JPA : Le savoir des autres n'est pas le sien propre ; en être "informé" ne suffit pas. Il faut se l'approprier. Pour cela l'élève doit être guidé, accompagné. C'est le rôle immémorial de l'enseignant qui met en place (implicitement pour les élèves) des situations d'apprentissages fondées sur les didactiques des disciplines, dans des démarches pédagogiques s'appuyant sur l'environnement et l'expérience des élèves, qui aide à mettre en évidence le simple dans le compliqué, dans des cadres disciplinaires qu'il faut construire pour les élèves : en mettant l'élève en contact avec une multitude de savoirs, en fait, Internet renforce la mission traditionnelle de l'enseignant dans un contexte où l'élève est sollicité ("parasité" ?) par une pléthore d'informations qu'il faut transformer en connaissances maîtrisées.

BT : *Et le quatrième statut ?*

JPA : On en parle beaucoup ces derniers temps. Il s'agit de l'informatique objet d'enseignement, discipline en tant que telle, élément de la culture générale scientifique et technique de tous les élèves. Il y avait dans les années 1980 dans les filières d'enseignement général des lycées, de la seconde à la terminale, une option informatique. Présente dans un lycée sur deux et en voie de généralisation, elle a été supprimée en 1992, rétablie en 1995 puis à nouveau supprimée en 1998.

BT : *Quel a été le motif de la suppression ?*

JPA : "Discipline élitiste" ! Parmi les vraies raisons figure une crise de recrutement des professeurs de mathématiques. Or ils représentaient la moitié des professeurs de l'option. Cela dit, le problème n'est pas franco-français. On peut observer le même phénomène de balancier en Suisse qui a rétabli il y a quatre ans un enseignement en informatique après l'avoir également supprimé dans les années 1990.

BT : *Ces dernières années, avec l'EPI, vous avez joué un rôle important dans la promotion de la nécessaire (re)création de l'informatique discipline scolaire.*

JPA : Avec Jacques Baudé, président d'honneur de l'EPI, nous avons relancé en 2007 la question d'un enseignement de l'informatique dans le secondaire. Étant entendu que l'EPI a toujours prôné la complémentarité de l'informatique objet de connaissance et l'informatique outil pédagogique qui se renforcent mutuellement. Notre action a abouti à la création à la rentrée 2012 en Terminale S d'un enseignement de spécialité optionnel "Informatique et sciences du numérique".

Je rappellerai que nous l'avons mené en compagnie de Serge Abiteboul (membre de l'Académie des Sciences, Professeur au Collège de France (2011-2012)), Gérard Berry (Professeur au Collège de France, membre de l'Académie des sciences et de

l'Académie des technologies), Colin de La Higuera (Président de la Société Informatique de France (SIF)), Gilles Dowek (Directeur de recherche à INRIA, Grand Prix de philosophie de l'Académie Française), Maurice Nivat (membre de l'Académie des Sciences) et tous les membres du groupe ITIC-EPI-SIF et de l'EPI.

Le retour de l'enseignement de l'informatique en 2013

BT : *Que s'est-il passé après la suppression de l'option ?*

JPA : Après la seconde suppression de l'option informatique dans l'enseignement général, la science informatique a vécu une traversée du désert. L'idée selon laquelle on peut donner une culture générale informatique à travers son utilisation dans les autres disciplines a prévalu. Elle s'est traduite dans la mise en place du B2i dont l'expérience de ces dix dernières années a montré que c'était un échec, un échec d'ailleurs prévisible. En effet, le B2i suppose implicitement un apport de connaissances mais ne dit pas où les trouver, dans quelles disciplines, ni même ce qu'elles sont ! Cette absence de contenus scientifiques explicitement nommés est déjà à elle seule un handicap majeur et rédhibitoire. Par ailleurs, il n'est pas évident d'organiser des apprentissages progressifs sur la durée lorsque les compétences recherchées sont formulées de manière très générale (du type "maîtriser les fonctions de base" ou "effectuer une recherche simple"), éventuellement répétitives à l'identique d'un cycle d'enseignement à l'autre. Mais quand, en plus, cela doit se faire par des contributions multiples et partielles des disciplines, à partir de leurs points de vue, sans le fil conducteur de la cohérence didactique des notions informatiques, par des enseignants insuffisamment formés, on imagine aisément le caractère ardu de la tâche au plan de l'organisation concrète. Pour se faire une idée de ces difficultés, il suffit d'imaginer l'apprentissage du passé composé et du subjonctif qui serait confié à d'autres disciplines que le français (dont on décréterait en passant qu'il n'a pas de raison d'être), au gré de leurs besoins propres (de leur "bon vouloir"), pour la raison que l'enseignement s'y fait en français. Idem pour l'apprentissage des mathématiques (exit aussi !), outil pour les autres disciplines. On confierait alors l'étude des entiers relatifs au professeur d'histoire qui les traiterait lorsqu'il s'intéresse à la période "avant-après J.-C.". Et les coordonnées seraient vues lors de la présentation des notions de latitude et de longitude en géographie. D'évidence cela ne marcherait pas. Et les faits ont montré que cela ne marchait pas non plus pour l'informatique.

BT : *Une informatique ou des informatiques ?*

JPA : Il y a une science informatique avec ses grands domaines : l'algorithmique, les langages et la programmation, la théorie de l'information, et les matériels (architecture, machine, réseaux). On les retrouve dans toutes les applications de l'informatique, dans le numérique dont le cœur est l'informatique.

L'idée générale que nous défendons est que l'informatique (le « cœur » du numérique) étant omniprésente dans la société, elle est une composante de la culture générale au XXI^{e} siècle et, à ce titre, de la culture générale scolaire. Est en jeu la formation de l'homme, du travailleur et du citoyen, à savoir les missions traditionnelles de l'École.

Rappelons que la culture générale scolaire n'est pas immuable. Le latin et le grec n'y ont plus la place qu'ils y tenaient antan. Les années 1970 ont vu la création de la discipline sciences économiques et sociales ; les probabilités ont fait leur apparition en mathématiques et la géométrie descriptive a disparu. Depuis longtemps, nous savons qu'il est indispensable que tous les jeunes soient initiés aux notions fondamentales de nombre et d'opération, de vitesse et de force, d'atome et de molécule, de microbe et de virus, d'événement et de chronologie, de genre et de nombre, etc. Ces initiations se font dans un cadre disciplinaire. Aujourd'hui, le monde devenant numérique, il faut faire de même avec l'informatique qui sous-tend le numérique. Il s'agit d'un choix de société de la première importance.

BT : *Former l'homme, le travailleur et le citoyen.*

JPA : Oui. Les métiers se sont transformés. Il y a de plus en plus de biens immatériels, de biens immatériels dans les biens matériels et dans les processus de création de la richesse. L'informatique et les sciences du numérique représentent 30% de la R&D au plan mondial mais 18% seulement en Europe. Pour construire les avions on ne fait plus d'essais en vol : on utilise la simulation... L'informatisation est la forme contemporaine de l'industrialisation. S'il y a plus d'un siècle les sciences physiques sont devenues discipline scolaire c'est parce qu'elles sous-tendent les réalisations de la société industrielle.

Il faut donner aux citoyens les éléments de compréhension permettant de participer aux grands débats actuels. Une bonne culture informatique est une condition nécessaire à l'exercice de la citoyenneté. C'est d'ailleurs le rôle que jouent les enseignements de la physique et de la chimie, des sciences de la vie et de la terre, pour les questions que posent le nucléaire et la génétique. Le manque d'une culture informatique chez l'homme de la rue, mais également chez les décideurs et les élites, est apparu au grand jour lors des débats qui ont porté sur la neutralité du Net ou l'Hadopi. En effet, s'il fut question de droit d'auteur et de DRM, ce fut sur fond d'interopérabilité, de code source et de réseaux de transit. Quid des représentations mentales permettant de savoir de quoi il est question ?

Et il y a la vie de tous les jours où il faut par exemple savoir décrypter l'offre d'un fournisseur d'accès.

BT : *La question qui se pose néanmoins est la suivante : a-t-on besoin de pratiquer une discipline pour en comprendre les implications dans la société ?*

JPA : Il est rarissime que l'on résolve une équation du second degré dans le quotidien. Cela étant, en résoudre au lycée a constitué un moment dans le long chemin permettant de comprendre qu'une grandeur peut dépendre d'une autre grandeur ; contribuant à donner de la substance à cette idée. Sans cela, il est par exemple beaucoup plus problématique de comprendre les évolutions d'une courbe du chômage présentée le soir aux informations.

Rappelons que, dans les années 1950 et 1960, il était nécessaire de valider à l'université un certificat de sciences pour obtenir une licence de philosophie.

Le rôle de l'École est irremplaçable. Des recherches ont montré que les compétences des "digital natives" étaient limitées (non transférables, non verbalisées ni conceptualisées et verbalisées). Observant les enfants des favelas à Rio de Janeiro, un chercheur a constaté qu'ils savaient multiplier 50 par 3. Mais pas 3 par 50. L'École est le seul endroit où les enfants rencontrent le savoir en tant que tel et d'une manière structurée.

L'enseignement scolaire, c'est fondamentalement de la culture générale. Il y a encore un sérieux déficit en la matière. Et l'on ne peut que faire un lien avec la pénurie en personnel qualifié et le manque de compétences globales de tous dans l'entreprise [7].

BT : *Quelles sont les perspectives d'évolution dans les années qui viennent ?*

JPA : L'enseignement de spécialité optionnel "Informatique et sciences du numérique" va être étendu aux sections ES et L d'ici 2015 ; c'est un engagement de campagne du président Hollande. L'informatique va être introduite dans les CPGE [8] à la rentrée 2013. Mais il reste l'école primaire, le collège (en relation à définir avec le cours de technologie). Et aussi le lycée dans la mesure où l'enseignement de l'informatique doit être un enseignement pour tous.

Il y a des défis à surmonter. Notamment le nombre des enseignants qu'il va falloir trouver pour assurer ces horaires actuels et à venir. Pour l'EPI, l'étape suivante est la mise en place d'un CAPES [9] et d'une agrégation d'informatique.

Bibliographie indicative.

Baron Georges-Louis, *L'informatique, discipline scolaire ?*, Presses Universitaires de France, Paris, 1987, 230 pages.

7. Louis Beck, *Contre l'illettrisme numérique en entreprise* (disponible à l'adresse http ://www.epi.asso.fr/revue/articles/a1301e.htm).

8. Classes Préparatoires aux Grandes Écoles.

9. Certificat d'Aptitude au Professorat de l'Enseignement du Second degré.

Dimet Bernard, *Informatique : introduction dans l'enseignement obligatoire (1980-1997)*, Editions L'Harmattan, Paris, 2003, 318 pages.

Harrari Michelle, *Informatique et enseignement élémentaire 1975-1996. Contribution à l'étude des enjeux et des acteurs*, Georges-Louis Baron, Université Paris V, 2000, 348 pages.

Le programme de la spécialité "Informatique et sciences du numérique" de Terminale S :
Dowek Gilles, Archambault Jean-Pierre & Alii, *Informatique et sciences du numérique - Spécialité ISN en terminale S, avec des exercices corrigés et des idées de projets*, Eyrolles, Paris, 2012, 301 pages.

Quelques points de repère dans une histoire de 40 ans :
Baudé Jacques, *L'association Enseignement Public et Informatique (EPI) de février 1971 à février 2011*.
`http://www.epi.asso.fr/revue/histo/h10oi_jb1.htm`,
`http://www.epi.asso.fr/revue/histo/h10oi_jb2.htm`,
`http://www.epi.asso.fr/revue/histo/h10oi_jb3.htm`.

Enseigner l'informatique aux jeunes enfants

Maurice Nivat [1]

Comme Monsieur Jourdain faisait de la prose sans le savoir, tout le monde programme sans en avoir vraiment conscience. Comme Monsieur Jourdain faisait de la prose dès qu'il ouvrait la bouche pour parler, on effectue un programme ou l'on écrit un programme dès que l'on cherche à atteindre un but au moyen d'une suite d'opérations élémentaires, ces opérations élémentaires (qui peuvent être tout ce qu'on voudra, la définition des opérations élémentaires est très arbitraire) étant échelonnées dans le temps ou simultanées, effectuées par une ou plusieurs personnes accompagnées de machines ou armés d'outils.

Il est rare que nous agissions sans but, notre action globale est la résultante d'une suite d'actions instantanées dirigées, organisées, exécutées dans un certain but. Réveillés à l'instant t, un jour de semaine ordinaire, nous allons à l'école, au lycée, au collège, à la fac, à l'atelier, au bureau, à l'usine, en général comme des automates : la suite des actions à effectuer est programmée à l'avance. Il faut se lever, s'habiller, prendre un petit-déjeuner, sortir de chez soi puis aller là où nous devons aller à pied, en vélo ou en voiture, en suivant un itinéraire connu pour arriver à l'heure h. Pour ce faire nous suivons un programme auquel, la plupart du temps, nous ne réfléchissons guère, car nous l'avons dans la tête, nous le suivons, nous l'effectuons *ne varietur*, jour après jour. Mais il est clair que si un incident se produit, si la voiture est en panne, si les transports en commun sont en grève, si une route est barrée, si nous nous réveillons plus tard que d'habitude, nous devons faire autre chose et la première des choses à faire est d'imaginer une autre façon d'atteindre notre lieu de travail dans le temps imparti. Et là, nous programmons, nous sommes obligés de programmer.

1. Professeur honoraire à l'université Paris Diderot et membre correspondant de l'Académie des Sciences.

Stanislas Dehaene, dans sa leçon inaugurale au Collège de France, ne craint pas d'écrire que le cerveau humain est une « machine à programmer » : cet éminent psychologue montre comment, quand nous apprenons à lire, une partie de notre cerveau se transforme pour recevoir le programme « câblé » qui nous permet de lire, c'est-à-dire de transformer en suite de sons émis par notre bouche une suite de symboles écrits sur une feuille de papier ou apparaissant sur un écran.

Nous pouvons résumer l'universalité de la notion de programme en peu de mots : pour faire on suit ou exécute un programme, pour faire faire on conçoit et on écrit un programme. Savoir faire c'est avoir dans la tête des programmes qui précisément permettent de faire.

Les hommes programment donc depuis qu'ils agissent autrement qu'en suivant leur instinct animal : nos lointains ancêtres programmaient une chasse au mammouth, l'érection d'un menhir, comme la fabrication des lames de silex. Des programmes très anciens étaient sûrement déjà très sophistiqués et certain défient même notre science moderne : on ne sait pas comment les hommes préhistoriques mettaient debout des menhirs pouvant peser cent tonnes, on ne sait pas comment les premiers égyptiens, grecs, chinois, romains, incas et autres construisaient avec une précision millimétrique avec des outils très rudimentaires et des appareils de mesure qui l'étaient encore plus. La somme des programmes utilisés depuis les temps les plus reculés est considérable et la plupart ne doivent rien à l'informatique, évidemment tous ceux qui ont précédé l'apparition de l'informatique (que nous faisons naître réellement en 1960 avec l'apparition des premiers langages de programmation de haut niveau, comme on les appelait alors, que sont FORTAN, Algol et Lisp) et encore aujourd'hui la plupart des programmes conçus, mis au point, utilisés, dans les divers secteurs de l'activité humaine, le sont sans intervention directe d'informaticiens.

Mais l'informatique, en soixante ans d'existence, a transformé nos vies, nos façons de travailler, sans doute aussi nos façons de penser et toutes nos façons d'agir. Ceci est dû au mélange de techniques informatiques qui (alliées à beaucoup d'autres) produisent quantités de circuits électroniques et qui permettent de faire effectuer à ces circuits avec une rapidité stupéfiante des opérations fort complexes et de science informatique, celle-ci étant avant tout une réflexion sur les programmes et leur efficacité ainsi que sur les langages permettant de décrire ces programmes. Comme toute science, la science informatique est un savoir concernant les programmes, née de la réflexion et du travail de nombreux chercheurs ainsi que de très nombreuses expériences et observations, qui incorpore, simplifie, explique, permet de comprendre les savoir-faire techniques accumulés au cours des âges, savoir-faire destinés à disparaître quand un nouveau procédé vient en remplacer un autre. La métallurgie, la science des matériaux, qui s'est constituée dans la deuxième moitié du dix-neuvième siècle est un exemple de science qui est née tard par rapport à des techniques qui ont

commencé à se développer dès l'âge du bronze quelques cinq mille ans avant notre ère et c'est plus à elle que je songerais à comparer l'informatique, science des programmes, née encore plus tard par rapport aux premiers programmes conçus et mis en œuvre au néolithique. Sauf que la métallurgie a comme champ d'action l'extraction et la purification des métaux et alliages, qui est une activité importante de l'humanité, mais une parmi des milliers, alors que l'informatique, en tant que science des programmes, a comme champ d'action potentiel l'ensemble des activités humaines qui donnent toutes naissance à des programmes.

La cause est aujourd'hui entendue : il faut enseigner de l'informatique à tous les niveaux de notre système éducatif, un récent rapport de l'Académie des sciences le démontre et cette conclusion est unanimement soutenue par le Conseil national du numérique. On vient aussi d'apprendre que la Grande-Bretagne a décidé de généraliser l'enseignement de l'informatique depuis l'école primaire.

Dans les lignes qui suivent nous tentons de préciser en quoi peut consister cet enseignement surtout pour les plus jeunes à l'école et au collège.

Nous pensons que le principal but de l'enseignement est d'apprendre à parler et à écrire dans sa langue et (un peu, en fait fort peu) dans le langage particulier des mathématiques. Les enfants qui réussissent à l'école et qui réussiront plus tard sont ceux qui parviennent à s'exprimer bien, clairement, correctement, voire avec élégance dans leur langue. On ne saurait surestimer l'importance de la maîtrise de la langue dans tout apprentissage.

Il se trouve que les langues vernaculaires sont mal adaptées à la description de suites d'opérations, comme on s'en convainc aisément en essayant de décrire précisément la suite d'opérations à effectuer pour réaliser des opérations familières et assez simples. Essayez de décrire précisément ce que vous faites pour remettre en ordre un tas d'objets qui ont été dérangés, essayez de décrire la stratégie que vous employez pour jouer à un jeu quelconque et vous serez vite convaincus.

Les manuels d'entretien, les modes d'emploi, les recettes diverses dont nous sommes abreuvés laissent souvent perplexes par leur imprécision, les ambiguïtés qu'ils recèlent aussi bien dans la désignation des actions que dans leur ordonnancement. Même le langage mathématique, qui pourtant se veut rigoureux, n'est pas exempt de cruelles imprécisions : de nombreux exercices proposés aux élèves des collèges et lycées consistent à « simplifier » une expression arithmétique alors que l'on ne sait pas, qu'il n'a nulle part été défini, ce qui est « simple » !

Cette inaptitude qui semble commune à toutes les langues a donné naissance à quantités de langages techniques ou notations spécialisées dont on est obligé de convenir qu'ils sont très peu compréhensibles du profane, de qui n'est pas du métier.

Les machines informatiques s'accommodent mal des imprécisions et des ambiguïtés et c'est ce pourquoi des langages de programmation sont nécessaires : toute

machine a son langage, très analytique puisqu'il décrit tous les déplacements du contenu d'une case de mémoire dans une autre, toutes les deux nommément désignées par leur adresse physique. La programmation en langage machine était cependant très fastidieuse et c'est, comme nous l'avons déjà dit, l'apparition des langages de haut niveau qui a permis de programmer facilement et, en même temps, de constituer une bibliothèque de programmes susceptibles, pour peu que l'on dispose du compilateur voulu, d'être exécutés par n'importe quelle machine. Toutes sortes de gens se sont mis à programmer à partir de 1960 et il est fallacieux de croire que c'était pour calculer, en donnant au mot calcul le sens qu'il a pris dans notre langue : le calcul scientifique n'occupe qu'une faible fraction des ordinateurs en service dans le monde, et les ordinateurs ne contiennent qu'une faible fraction des puces électroniques en service dans le monde. La plupart des ordinateurs servent à archiver, ranger, trier, classer, rechercher d'innombrables informations ou à contrôler des systèmes complexes, la plupart des puces sont incorporées à des machines ou à des systèmes dont elles assurent certaines fonctions à côté d'autres organes physiques mécaniques ou d'acteurs humains.

La pratique de la programmation à grande échelle et le développement de la théorie de la programmation font que c'est à peu près tout qui aujourd'hui se programme, tous les processus, industriels, administratifs, médicaux, architecturaux, dans toutes leurs composantes. C'est cette programmation qui permet de construire des machines de plus en plus à la fois sophistiquées et performantes, c'est cette programmation qui permet d'innover bien souvent en substituant un organe électronique à un organe mécanique, en confiant une opération qui était dévolue à un acteur humain à un organe électronique, quand ce n'est pas toute l'activité d'un être humain qui se trouve effectuée par un robot plus fiable et plus rapide : l'automobile et les économies d'énergie dans le bâtiment comme ailleurs sont deux domaines dans lesquels l'innovation passe pour l'essentiel par l'informatisation, c'est-à-dire le truffage par des puces et des logiciels (et nous n'avons pas encore tout vu ; d'ici dix ans, les voitures déjà dotées de nouvelles fonctionnalités pour automatiquement manœuvrer et éviter les obstacles se conduiront pratiquement — ou presque — toutes seules).

Un des grands pionniers de l'informatique, Donald Knuth, parlait dans la préface de l'ouvrage en trois volumes « *The Art of Programming* », paru dans les années quatre-vingt et qui a eu une influence considérable sur toute une génération d'informaticiens, du plaisir de programmer, plaisir finalement très voisin, pensons-nous, du plaisir d'écrire un beau texte en bon français. Nous pensons que ce n'est rien à côté du plaisir que procure la programmation réussie d'un gros système, car on ne peut programmer sans comprendre complètement comment le système fonctionne ou devrait fonctionner.

L'activité de programmation non seulement ne décroit pas mais elle s'amplifie, à vrai dire, elle a changé de nature : il y a toujours pas mal de gens occupés à écrire

des programmes (baptisés logiciels, quand ils sont suffisamment documentés et testés pour être diffusés ou vendus) réalisant de nouvelles opérations et ceux-ci s'accumulent, il y a toujours la possibilité d'écrire un logiciel innovant qui fait mieux que ses prédécesseurs, ces logiciels se comptent par centaines de milliers mais la programmation est de plus en plus une programmation globale de systèmes dans laquelle les opérations réalisées par les logiciels existants servent d'opérations élémentaires. Le problème à résoudre n'est plus de programmer une machine mais, face à un système, ou un processus, de déterminer quand et comment on peut utiliser des machines pour effectuer des parties des opérations nécessaires et d'assurer la bonne intégration des parties automatisées avec les autres qui restent à la charge d'opérateurs humains.

Il faut apprendre aux enfants à programmer

À nos yeux peu importe dans quel langage, il y en a désormais des quantités et même de nombreux assez simples qui ont été spécialement conçus pour les enfants, comme le langage LOGO de Seymour Papert, on peut même programmer, dans une certaine mesure, en français.

L'amusant est que l'on demande déjà aux enfants de programmer et d'exécuter certains programmes dès leur entrée à l'école primaire et l'on peut difficilement faire autrement :

– la notion de nombre se construit à partir de la mise en correspondance biunivoque d'un ensemble d'objets et d'une section commençante de la suite des entiers naturels que les enfants doivent apprendre par cœur et cette mise en correspondance résulte d'un programme qui a l'air simple mais qui occupe les élèves du CP pendant trois ou quatre mois : prendre un objet de l'ensemble E et le baptiser un, l'enlever de E, et prendre un objet du nouvel ensemble E, le baptiser deux, et ainsi de suite, jusqu'à ce que E soit vide.

– l'orthographe est constituée de choses qu'il faut savoir, apprendre par cœur, qui ne peuvent s'inventer (par exemple le vent qui souffle, s'écrit *vent*, et non *van*, ou *vend*) et de règles qu'il faut apprendre à appliquer ce qui suppose une analyse de la phrase pour découvrir à quoi se rapporte un adjectif ou quels sont le genre et le nombre de certains compléments.

Nous pensons que tous les petits programmes qui apparaissent ainsi dans l'enseignement d'aujourd'hui doivent être identifiés, explicités, et débarrassés de toutes les ambiguïtés ou incertitudes qui en rendent aujourd'hui la compréhension et la mise en œuvre difficiles.

Si l'on y réfléchit bien, c'est ainsi que l'informatique s'est imposée comme un outil de gestion, aussi bien d'entreprises ou de chantiers, que d'administrations, d'une

grande efficacité : tous les programmes plus ou moins implicites que devaient exécuter le personnel ont été rendus explicites, souvent simplifiés, et toujours débarrassés des imprécisions pouvant donner lieu à des interprétations diverses.

Appliquons à l'école la méthode qui a été appliquée dans la vraie vie : informatiser c'est d'abord expliciter complètement ce que l'on veut faire, c'est décrire précisément toutes les situations qui peuvent se présenter et dire quelle conduite à tenir face à chacune d'elles.

Il faut apprendre aux enfants à ranger, à classer, à ordonner

S'il est une opération que tout le monde est amené à effectuer, c'est ranger, c'est disposer ses affaires ou ses idées ou ses connaissances dans un certain ordre qui permet de les retrouver quand on en a besoin. Les enfants apprennent à ranger bien avant leur entrée à l'école, en général de leurs parents. Curieusement l'école, elle n'apprend pas explicitement à ranger, les professeurs peuvent donner des conseils, peuvent aussi s'insurger contre le désordre d'un enfant, mais « ranger » n'est pas une matière à enseignement. Ranger, comme programmer, est une activité très ancienne, il y avait des bibliothécaires qui dans l'ancienne Égypte rangeaient les papyrus, mais il a fallu attendre le dix-neuvième siècle pour que le rangement devienne un sujet de recherche et donne naissance à des outils techniques destinés à le rendre efficace et rapide : c'est Hollerith qui a inventé la carte perforée, et ainsi donné naissance à l'industrie mécanographique pour exploiter les données fournies par un recensement de la population des États-Unis d'Amérique.

Nous venons d'employer le mot de données, nous aurions pu employer celui d'informations. L'informatique s'appelle ainsi parce qu'elle traite l'information, et les ordinateurs parce qu'ils sont faits pour ordonner celle-ci.

Nous n'allons pas tenter de définir précisément ce qu'est l'information, c'est parfaitement inutile. Les seules choses sûres sont que nous avons autant besoin d'information pour vivre que d'oxygène et qu'elle coule à plein bord et que chacun d'entre nous en reçoit une énorme quantité tous les jours, la presse écrite ou parlée en diffuse en continu, le net nous en apporte à domicile beaucoup plus que nous n'en pouvons absorber. Que l'information nous soit indispensable pour vivre est évident : si en nous réveillant nous ne savons pas où nous sommes, si nous ne sommes pas capables de nous situer dans l'espace, ou d'identifier l'espace dans lequel nous nous trouvons, nous ne pouvons évidemment rien faire.

Que nous n'ayons que faire de la plus grande partie de l'information qui nous arrive, que nous percevons avec nos yeux et nos oreilles est aussi manifeste. Heureusement, nous avons une grande faculté d'oubli, la plupart des choses que nous voyons ou entendons disparaissent presque instantanément.

Pour agir, pour vivre, nous avons quand même besoin de beaucoup d'informations qu'en général nous portons avec nous dans notre mémoire. La mémoire humaine

est quelque chose de stupéfiant, elle peut contenir énormément de choses, certaines qui demeurent très longtemps, d'autres qui ne s'inscrivent que le temps où elles sont utiles : mais elle n'est pas toujours fiable et nous notons sur des carnets ou des calepins les informations qui nous paraissent indispensables.

L'informatique a donné naissance à ce qu'on appelle les bases de données pour stocker de l'information en quantité pratiquement illimitée de façon structurée pour permettre la recherche dans ces bases que l'on interroge en posant des questions. Le net a familiarisé tout le monde avec les moteurs de recherche qui donnent les adresses de tous les articles qui sont sur le net dans lesquels apparaissent certains mots, le net agissant alors comme une gigantesque base de données.

Il paraît un peu aberrant que les informations que l'on demande aux élèves d'apprendre, et Dieu sait si elles sont nombreuses, ne soient pas consignées dans des bases de données, alors que partout ailleurs qu'à l'école, dans la vraie vie, les informations, de quelque nature qu'elles soient, le sont ! Car ainsi, non seulement elles sont stockées, mais elles sont structurées, classées, hiérarchisées, reliées entre elles par des relations.

Les enfants dès qu'ils savent lire et écrire devraient constituer leurs propres bases de données contenant, dans un ordre auquel il convient évidemment de réfléchir longuement, l'essentiel de ce qu'ils sont supposés apprendre et savoir dans toutes les matières.

Là, nous devons insister : la programmation est l'explicitation des suites d'opérations à effectuer pour faire quelque chose et nous pensons que le premier but de l'enseignement de la programmation est de faire réfléchir les enfants à la notion d'action, d'obtention d'un but, ou d'un résultat par une suite d'actions qui y mène. Les bases de données sont l'occasion de collecter, de classer des informations et de réfléchir ainsi aux liens qui existent entre elles, donc à la structure de la connaissance. Comme la programmation devrait être enseignée essentiellement à partir de programmes qui apparaissent déjà dans les programmes scolaires, sous forme implicite, les bases de données devraient être construites et utilisées pour structurer les connaissances qu'on leur demande déjà d'acquérir.

Dans les deux cas l'informatique apparaît à la fois comme un outil pratique et comme un outil conceptuel, une espèce de méthode générale pour décrire, préciser, simplifier, effectuer des actions, expliciter des relations. Toutes les matières enseignées devraient idéalement servir de support à une activité et une réflexion informatique, ce qui permettrait de mettre en parfaite lumière le caractère universel de l'informatique, de ses idées, de ses concepts, de ses méthodes, de ses machines.

Conclusion

Le retard que nous avons pris en France, en matière d'enseignement de l'informatique, est dû au fait que longtemps on n'a voulu voir dans l'informatique qu'une

discipline ancillaire, au service des autres. Et beaucoup de gens se sont mis à faire de l'informatique sans en avoir vraiment appris, avec bonheur et parfois grand succès. Le temps où l'on pouvait croire que l'informatique n'était qu'une technique pourvoyeuse d'outils bien commodes est je pense tout à fait révolu.

Par ses concepts l'informatique est au cœur de nombreuses recherches en particulier toutes celles qui portent sur le fonctionnement du cerveau humain, de la mémoire et finalement sur ce que sont vraiment la connaissance et le savoir.

Par ses méthodes de programmation et de structuration des données elle irrigue tous les domaines de la connaissance et tous les secteurs de l'activité humaine.

Il n'est en effet que temps de donner désormais à nos enfants, en commençant par les plus jeunes, la meilleure formation possible à l'informatique.

J'enseigne la programmation au lycée...

David Roche [1]

David Roche est enseignant de sciences physiques au Lycée Guillaume Fichet de Bonneville en Haute-Savoie. Avant de s'impliquer cette année dans l'enseignement d'ISN en classe de Terminale, il a expérimenté depuis quelques années une initiation à la programmation avec ses élèves en classes de Seconde et Première. Il nous relate ici la mise en place de ces enseignements et leur évolution au fil du temps.

Genèse du projet

« La plupart de nos élèves utilisent un ordinateur quotidiennement. Les personnes de ma génération, nées dans les années 70, ont eu de la chance ! Certes les micro-ordinateurs étaient beaucoup moins répandus dans les foyers, mais vu leurs faibles capacités, ils ne pouvaient servir qu'à une seule chose : apprendre la programmation.

Aujourd'hui, les adolescents utilisent ce formidable outil principalement pour jouer ou pour parcourir les réseaux sociaux. Pourquoi ne pas leur faire découvrir cette activité, bien plus enrichissante intellectuellement, qu'est la programmation ? »

Telle fut notre réflexion de base.

L'enseignement de l'informatique au lycée en 2009

À cette époque, mai 2009, la spécialité ISN de Terminale S, Informatique et Sciences du Numérique, n'existait pas encore et l'option informatique avait disparu depuis longtemps. L'enseignement de l'informatique était totalement absent des programmes du lycée.

L'utilisation des « outils numériques » était déjà à la mode. Il suffisait de demander un TBI (Tableau Blanc Interactif) pour en obtenir un. Cette « révolution » de l'enseignement par les outils numériques devait être la solution miracle à tous nos problèmes ! Quelques années plus tard, je trouve les résultats bien maigres, mais c'est un autre débat.

Certes, il existait un enseignement de détermination en classe de seconde dénommé MPI : Mesures Physiques et Informatique. Cet enseignement de trois heures par semaine proposait aux élèves d'aborder quelques notions périphériques de la science informatique : conversion analogique-numérique, porte logique, capteurs... mais il restait très coloré sciences physiques et ne proposait aux élèves aucune initiation à la programmation.

1. Enseignant au Lycée Guillaume Fichet, Bonneville, Haute-Savoie. david.roche@ac-grenoble.fr

Depuis 2007 ou 2008, l'enseignement MPI était en perte de vitesse dans notre lycée, de moins en moins d'élèves candidataient.

Le contenu, relativement difficile d'approche, ne convenait plus à nos élèves. La proviseure de l'époque nous a demandé si nous avions des idées pour revitaliser l'enseignement MPI. La réponse fut affirmative.

Début d'expérience en seconde

Décision est prise, avec l'accord de notre inspection, de proposer aux élèves suivant l'enseignement MPI, une initiation à la programmation. Nous étions alors à l'extrême limite de ce que nous autorisait le programme de MPI.

Les bases étant posées, restait une question fondamentale : comment faire ? Physicien de formation, je suis un autodidacte en programmation.

Après avoir envisagé diverses solutions (utilisation de python...), décision fût prise d'utiliser le logiciel développé par l'Université Carnegie Mellon : Alice.

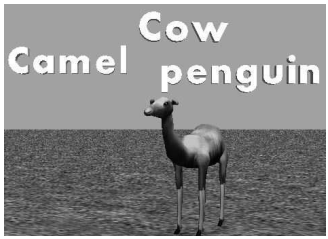
Alice est un logiciel gratuit qui permet d'aborder toutes les structures de base (boucle, condition, fonction avec passage de paramètre...), et même la programmation orientée objet, en permettant aux élèves de créer des scènes 3D interactives peuplées de personnages plus étranges les uns que les autres.

Alice est un environnement interactif de création de programmes. Le programmeur déplace des briques graphiques qui correspondent à des instructions ou des constructions du langage de programmation. L'exécution du programme au sein de l'environnement permet de faire facilement le lien entre les instructions et le comportement des objets dans l'animation des scènes créées.

Suppression de MPI, création de PSN

En 2010, dans le cadre de la réforme de la seconde générale, le ministère décide de supprimer l'enseignement MPI. Le nouvel enseignement d'exploration MPS, Méthodes et pratiques scientifiques, une heure et demie par semaine, est mis en place pour remplacer l'enseignement de MPI. MPS ne nous permettant pas de poursuivre notre expérience avec Alice, nous avons dû créer de toutes pièces, avec l'aide de l'inspection, un nouvel enseignement que nous avons nommé PSN, Pratique scientifique et numérique.

Exemple d'exercice « Alice » proposé aux élèves



Vous allez créer un "jeu" pédagogique destiné aux enfants de l'école primaire : l'enfant verra apparaître sur l'écran un animal (un chameau, une vache ou un pingouin) ; au-dessus de l'animal, l'enfant aura 3 choix possibles (texte en 3D) : "Camel", "Cow" ou "Penguin". Si l'enfant clique sur le bon mot (celui qui correspond à l'animal affiché), on verra apparaître "Bravo" et un nouvel animal s'affichera à la place du précédent. Dans le cas contraire on verra apparaître "Faux essaye encore une fois".
Attention, le choix de l'animal à afficher (vache, chameau ou pingouin) devra être aléatoire.

Pour les connaisseurs des arcanes de l'Éducation nationale, nous avons bénéficié d'un « article 34 » qui nous accorde, sous conditions, une certaine liberté pédagogique pendant trois heures par semaine.

Cet enseignement, spécifique à notre lycée, propose trois types d'activités aux élèves :

– initiation à la programmation avec Alice (reprise de ce que nous faisions en MPI) ;

– construction de A à Z d'un altimètre électronique (sans utilisation d'une quelconque mallette pédagogique, en utilisant exclusivement des composants du commerce : capteur de pression, plaquette d'essai, ampli-op...) ;

– physique et physiciens du XX^{e} siècle : initiation à la mécanique quantique, à la relativité et à la physique des particules (avec une visite du CERN à Genève).

Cet enseignement accueille une vingtaine d'élèves. Nous réalisons une « sélection » sur dossier, car la demande est souvent supérieure au nombre de places disponibles, cette sélection est principalement basée sur le sérieux des élèves et pas forcément sur leur niveau. Les filles et garçons sont à peu près à part égale.

Pour ce qui est de l'initiation à la programmation, les élèves ont chaque semaine une ou deux activités à réaliser. Les activités sont composées d'une partie « théorique », par exemple : qu'est-ce qu'une boucle ?. Cette partie s'appuie sur de nombreux exemples utilisant ou non le logiciel Alice. Elle est suivie de nombreux exercices à résoudre en utilisant Alice. Les élèves travaillent en autonomie sur les activités, l'enseignant est uniquement là pour les aider en répondant à leurs questions.

Nous nous sommes rapidement rendus compte qu'en cas de monologue de l'enseignant sur les différents thèmes à aborder, l'attention des élèves n'excède pas 15

minutes : l'apprentissage de la programmation au lycée ne se prête pas aux cours magistraux.

Cap sur la première S

À la fin de l'année scolaire 2010, fin de la première année d'initiation à la programmation avec Alice, plusieurs élèves ayant suivi l'enseignement MPI nous ont fait part de leur désir de poursuivre leur découverte de la programmation. Décision est alors prise de créer un club informatique, aucune structure officielle ne pouvant accueillir cet enseignement. À mon grand regret, à cause de contraintes d'emploi du temps, seuls les élèves de première S ont pu participer aux activités du club. Je suis personnellement convaincu que la programmation est une activité qui ne doit pas être réservée qu'aux scientifiques.

La réforme de la première, mise en place en septembre 2011, va nous permettre de poursuivre cet enseignement de la programmation en première S dans un cadre plus officiel.

Avec cette réforme arrive l'Accompagnement personnalisé, AP. En étant une fois de plus « limite » avec les textes réglementaires régissant cet AP, mais toujours en concertation et en accord avec notre hiérarchie, nous proposons aux élèves de poursuivre (ou de commencer, les débutants étant les bienvenus) leur découverte de la programmation dans le cadre de l'AP. Les heures d'AP sont intégrées dans l'emploi du temps des élèves, alors que l'heure du club informatique ne l'était pas. Cela peut sembler un détail, mais avec ce changement l'effectif a, en moyenne, doublé.

L'AP fonctionne par quinzaine. Les élèves doivent choisir deux disciplines scientifiques (parmi maths, sciences physiques, SVT et informatique). Pour une quinzaine donnée, un élève n'a pas la possibilité de choisir deux fois la même matière. Au maximum les élèves ont donc la possibilité (mais pas l'obligation !) de suivre une heure d'informatique tous les quinze jours.

Alice en seconde, mais que faire en première ?

Il ne me paraissait pas opportun de poursuivre avec Alice en première S. Il n'était pas question non plus de proposer aux élèves une extension du cours de mathématiques (depuis quelques années, l'algorithmique est au programme de mathématiques en seconde, en première S et en terminale S). Au-delà de l'aspect scientifique et technique, la programmation est, selon moi, une activité permettant à nos élèves d'exprimer leur potentiel créatif. Ce potentiel créatif a très peu l'occasion de s'exprimer dans les enseignements classiques. Cette vision des choses a toujours guidé mes choix. C'est d'ailleurs une des raisons qui me pousse à penser que la programmation n'est pas une activité à réserver aux seuls élèves des filières scientifiques ; pourquoi un élève de section littéraire serait-il incapable de développer un site web complet

(HTML, CSS, JavaScript, langage côté serveur) ou une application web « single page » ? En quoi les notions mises en jeu seraient hors de sa portée ?

N'ayant aucune expérience en matière d'enseignement de la programmation (si j'excepte l'utilisation d'Alice en seconde), j'ai dû tester beaucoup de choses... et donc subir quelques « échecs ». Voici un bref résumé narrant mes pérégrinations.

En 2010, les smartphones commençant à se retrouver de plus en plus souvent dans les poches de nos élèves, je décide de leur proposer une initiation à Java suivie d'une initiation au développement de logiciels pour la plateforme Android. Mais si les élèves les plus motivés accrochent immédiatement, l'essentiel de leurs camarades est rebuté par la difficulté et abandonne rapidement. Je qualifierai donc cette expérience *dev Android* de demi-échec.

Après divers essais plus ou moins fructueux (GTW de Google, Unity 3D), la découverte du couple HTML5 + JavaScript va enfin me permettre de trouver un outil adapté à mes objectifs.

Sans vouloir trop entrer dans des détails techniques, la balise HTML5 canvas, entre autres, a un intérêt pédagogique qui me paraît évident. Elle permet notamment aux élèves de développer de petits jeux vidéo en utilisant par exemple le framework JavaScript *EaselJS*. (J'ai écrit une documentation sur l'utilisation de ce framework.)

À ce sujet, les collègues des autres disciplines sont parfois quelque peu ironiques et s'imaginent que les activités « jouer à des jeux vidéo » et « programmer des jeux vidéo » sont similaires. Il y a sans aucun doute un énorme déficit de communication à résorber de toute urgence.

La création de jeux vidéo n'est évidemment pas une obligation, certains élèves choisissent de créer un site internet (utilisation de HTML5, CSS3 et JavaScript).

Étant donné l'organisation de l'accompagnement personnalisé décrite ci-dessus (cours à la carte avec au maximum une heure tous les quinze jours), il est difficile d'organiser une progression. J'ai donc choisi de fournir aux élèves une documentation abordant les différents aspects (HTML5, CSS3, JavaScript, EaselJS...) et d'utiliser l'heure de quinzaine pour répondre à leurs questions, en partant du principe qu'ils ont travaillé chez eux entre deux séances.

En fin de compte, il s'avère que les élèves travaillent uniquement pendant les séances en classe. La progression est donc extrêmement lente, une heure par semaine me semble un minimum en-dessous duquel il ne faudrait pas descendre, mais en ces temps de disette budgétaire, il ne faut pas rêver.

Malgré cela je poursuis tout de même l'expérience, car cet enseignement en première permet d'assurer une continuité entre la seconde, PSN, et la terminale S, ISN.

PSN à la rentrée 2013

Nous prévoyons une modification du contenu de l'enseignement PSN pour septembre 2013, toujours en accord avec l'inspection ; n'allez pas imaginer que nous avons le « droit » de faire n'importe quoi !

Premièrement, terminé Alice, je me sens prêt à aborder le JavaScript avec les élèves de seconde (je suis en train de travailler sur des activités spécialement conçues pour eux, un site internet interactif style *codecademy.com* pourrait aussi voir le jour). Les réactions et l'aisance des élèves de première à s'approprier les notions de base m'ont convaincu que leurs camarades de seconde (voire même de collège...) sont tout à fait capables d'aborder directement la programmation (en utilisant un « vrai » langage, sans passer par l'intermédiaire d'un logiciel comme Alice) à condition de s'adapter à leur rythme d'apprentissage (passer plus de temps et surtout détailler des notions qui pourraient paraître évidentes mais qui ne le sont pas pour des élèves débarquant à peine du collège et totalement novices en la matière).

Une fois les bases acquises, les élèves pourraient travailler avec la balise canvas comme les élèves de première S.

Autre possibilité, la création de *web app* en utilisant les très à la mode *AngularJS* côté client et *nodejs* côté serveur. Le travail sur la notion client-serveur nous permettrait d'aborder des concepts liés à la notion de réseau dès la seconde, toujours amener les aspects « théoriques » en s'appuyant sur du « pratique ».

L'idée serait, le plus rapidement possible, d'amener les élèves à s'investir dans un projet : mon année d'ISN m'a définitivement convaincu de la pertinence de la *pédagogie de projet*.

En lieu et place de l'altimètre, nous aimerions initier les élèves à la programmation d'un microcontrôleur. Notre choix s'est naturellement porté sur l'Arduino Uno. Ici encore la pédagogie de projet sera privilégiée.

Les deux aspects évoqués ci-dessus pourront même fusionner en un seul projet puisqu'aujourd'hui il est possible de programmer l'Arduino en JavaScript.

En première, les élèves pourront poursuivre leur initiation et participer au projet « Une start-up au lycée ».

Projet « une start-up au lycée »

La création d'une start-up dans le domaine des « nouvelles technologies » demande bien évidemment des personnes ayant des « talents » de codeur, mais pas seulement. Il faut aussi des connaissances en marketing (étude de marché, etc.), en droit, en management d'équipe, en design...

Des élèves n'ayant pas de connaissances particulières en programmation pourraient donc aussi participer au projet. Au lycée G. Fichet nous avons une filière technologique STMG (sciences et technologies du management et de la gestion),

avec les spécialités Gestion et finance, Ressources humaines et communication, et Marketing (mercatique). La participation d'élèves de cette filière serait, je pense, un véritable plus. Nous avons également dans le lycée des bacs pro commerce qui pourraient aussi être motivés par un tel projet. Cela serait un bon moyen de faire travailler ensemble des élèves qui n'ont pas forcément l'habitude de se côtoyer.

Mais travailler sur quoi ? On peut imaginer qu'un élève ait une idée de service de soutien scolaire « par des lycéens pour des lycéens ». Un site internet avec un système de visioconférence et un tchat pourrait être envisagé. Viendrait ensuite le problème du « modèle économique » à mettre en place. Les « professionnels » du marketing auraient aussi un rôle certain à jouer. Tout cela pouvant être, je suis sûr, d'excellents travaux pratiques.

Évidemment, ceci n'est qu'un exemple, les idées devront venir des élèves eux-mêmes.

Des intervenants extérieurs pourraient apporter leur expertise et leur expérience aux élèves : dirigeants d'entreprises, « pépinière » de start-up, etc.

Je tiens à préciser qu'évidemment l'aspect mercantile sera purement virtuel.

Ce projet est à l'état embryonnaire, je recherche des personnes compétentes dans ces domaines pour m'aider dans sa mise en place. Si vous avez des idées sur ce sujet, n'hésitez pas à me contacter.

Quelques précisions pour terminer

Je n'ai jamais eu pour objectif de former des développeurs professionnels, j'en serais bien incapable, mais juste de proposer aux élèves une activité que je trouve intellectuellement intéressante. Pour ce faire, j'ai choisi d'utiliser des technologies modernes et « à la mode » afin de rendre cette approche la plus « fun » possible pour les élèves. Certains pourront sans aucun doute me reprocher un manque de classicisme, mais j'insiste encore une fois, mon objectif n'est pas de former des programmeurs, et encore moins des informaticiens.

De la même façon, je ne prétends pas faire de la science informatique avec mes élèves de seconde et de première. Je réserve cela, pour l'instant, aux élèves suivant la spécialité ISN en terminale.

Je pense que l'initiation au développement peut être une porte d'entrée pour amener les élèves à s'intéresser à la science informatique et donc susciter des vocations. En effet, les plus motivés et les plus intéressés des élèves auront, sans doute, l'envie d'aller plus loin et de comprendre ce qui se passe « sous le capot » : mais qu'est ce qu'un langage ? Ce langage est-il directement compris par le microprocesseur ? Qu'est-ce qu'un microprocesseur ? Pourquoi certains algorithmes sont plus « efficaces » que d'autres pour réaliser les mêmes opérations ? Pour répondre à toutes ces interrogations les élèves devront s'intéresser à la science informatique.

Pour conclure, je trouve qu'il est rassurant que l'Éducation nationale que l'on présente souvent comme un mastodonte inerte puisse permettre à ce genre d'expérience de se développer. Je tiens tout de même à préciser que rien n'aurait été possible sans le très fort soutien de l'Inspection de Sciences physiques et des différentes équipes de direction qui se sont succédées au lycée G. Fichet depuis 2009.

Quelques liens

- Site officiel Alice : www.alice.org
- Tous mes documents (sous Creative Commons) : www.infoaulycee.byethost5.com/ressources.html
- Mon site sur le « développement web » : www.webisn.byethost5.com/

Journée des doctorants, Congrès de la SIF 2013

Vincent Autefage, Claudia Marinica, Olivier Baudon

La dernière journée du Congrès de la SIF, qui s'est déroulé du 6 au 8 Février 2013 à Nice Sophia-Antipolis, était consacrée aux doctorants. Différents intervenants sont venus parler et échanger sur des sujets variés tels que les publications, la valorisation des résultats ou encore l'après-thèse. Ce compte-rendu relate les points centraux des différents exposés et discussions qui ont eu lieu lors de cette journée.

Publications et valorisation des résultats

Création d'entreprise

André Labat (incubateur Paca-Est) a expliqué les mécanismes qui permettent à un chercheur de s'investir dans une entreprise privée tout en conservant son poste de fonctionnaire. Depuis la loi Claude Allègre (1999), un chercheur travaillant dans le secteur public a cette possibilité. Il peut être actionnaire d'une société à hauteur maximum de 49% des parts, mais ne peut pas y occuper de poste de direction. Un chercheur ayant des projets d'entrepreneuriat peut faire appel à un incubateur pour l'aider dans ses démarches. Un incubateur est une structure publique ayant pour but d'apporter une aide logistique ainsi que des formations aux chercheurs dans une démarche de création d'entreprise pour une durée maximale de 24 mois. Les critères d'incubation sont principalement les suivants :

- disposer d'une idée innovante ayant un lien avec la recherche ;
- être présent physiquement sur la région d'incubation ;
- ne pas avoir créé de société depuis au moins 6 mois.

Propriété intellectuelle

Eric Catapano (INPI [1]) a présenté les grands axes de lois en matière de propriété intellectuelle scientifique qui s'appliquent notamment au CNRS. Il a également abordé la problématique des brevets qui sont de plus en plus reconnus dans la recherche publique. Il a indiqué qu'un brevet devait être déposé avant toute publication relative pour être valable. L'utilisation d'un cahier de laboratoire, gardant la trace du cheminement ayant abouti à l'innovation, est un bon moyen de prouver la propriété des résultats obtenus.

Crédit d'impôt recherche

Nadine Marchandé, chargée de mission au MESR [2], a abordé les mécanismes du crédit d'impôt recherche (CIR). Ce dispositif a été mis en place en 1983 dans le but d'inciter les entreprises à faire de la recherche en permettant une défiscalisation partielle de leurs impôts. Une entreprise voulant profiter de ce dispositif doit rédiger un dossier mettant notamment en avant des verrous ou des incertitudes scientifiques. Elle doit également justifier du temps effectif de travail de son personnel ainsi que de la qualification recherche de celui-ci. Les dossiers sont examinés par des experts académiques. Du point de vue du recrutement, une entreprise obtient une défiscalisation à hauteur de 50% du salaire de ses chercheurs, ingénieurs ou techniciens sur un projet relevant du CIR. Ce taux passe à 200% dans le cas d'un jeune docteur.

Évaluation de la recherche

Jean-Pierre Merlet, directeur de recherche INRIA, a abordé les problèmes relatifs à l'évaluation de la recherche. Il a notamment pointé les points faibles des indicateurs suivants :

- le nombre de publications d'un auteur n'est pas forcément représentatif dans la mesure où il n'y a pas de preuve qu'il en soit l'auteur véritable ;
- la position d'un auteur dans ses publications n'est valable que si l'ordre est significatif ;
- le nombre de citations d'un auteur n'est pas nécessairement représentatif de la qualité de son travail ;
- les moteurs de citations tels que Thomson [3] ou Scopus [4] couvrent entre 2 et 20% des revues mondiales ;
- le facteur d'impact (*impact factor*) n'est pas exempt de failles ;
- le *h-index* tolère le mécanisme de citations circulaires.

1. Institut National de la Propriété Industrielle.
2. Ministère de l'Enseignement Supérieur et de la Recherche.
3. `http://thomsonreuters.com/web-of-science/`
4. `http://www.info.sciverse.com/scopus`

Il a également insisté sur l'importance du choix des publications mises en avant lors des recrutements. Il conseille ainsi d'en sélectionner 3 à 5 qui répondent aux critères de l'offre d'emploi.

Archives ouvertes

Estelle Nivault, documentaliste INRIA, a expliqué la démarche des archives ouvertes. Le but de ces plates-formes est de mettre les résultats de recherche à disposition du plus grand nombre sans restriction d'accès ; HAL [5] (publications) et TEL [6] (thèses) en sont des exemples bien qu'il existe à ce jour plus de 3000 archives ouvertes. Depuis 2012, la Commission européenne recommande que toute recherche financée sur fonds publics soit diffusée librement dans un délai de 6 à 12 mois.

Conseils de publication

Jean-Claude Bermond, directeur de recherche émérite CNRS, a présenté une collection de conseils pour réussir ses publications tant sur le plan de la rédaction que sur le plan de la soumission. Il a notamment insisté sur l'utilisation d'un correcteur orthographique, la non multi-soumission d'un article et enfin de tenir fortement compte de l'avis des *reviewers* de ses articles.

Présentation de la CERNA

Max Dauchet, professeur émérite, membre de la Commission nationale d'éthique, a présenté les principaux axes de la Commission de réflexion sur l'Éthique de la Recherche en sciences et technologies du Numérique (CERNA [7]) d'Allistene. Son objectif est de mener une réflexion sur les briques scientifiques en amont de toute application. Ses réflexions actuelles s'orientent autour de l'éthique de la recherche en robotique, ainsi que sur l'accès aux données à des fins de recherche en sciences et technologies du numérique.

Conférence invitée sur la vie en milieu académique

Serge Abiteboul, directeur de recherche INRIA, membre de l'Académie des sciences et du Conseil national du numérique, a présenté sa vision sur la vie en milieu académique et le bonheur d'être enseignant-chercheur. Il a également abordé les clés pour faire vivre son sujet de thèse, son conseil final étant d'*être fainéant mais de travailler beaucoup*. Ses transparents sont disponibles sur son site web [8].

5. http://hal.archives-ouvertes.fr
6. http://tel.archives-ouvertes.fr
7. https://www.allistene.fr/cerna
8. http://abiteboul.com/pub/13.PhD.SIF.ppt

L'après-thèse

Cette session a principalement porté sur les procédures de recrutement pour les jeunes docteurs.

Recherche publique

INRIA

Pascal Guitton, directeur de la recherche à INRIA, a expliqué le fonctionnement de l'institut et notamment la notion d'équipe-projet. Il a brièvement abordé la question de la rémunération (niveaux CR [9] et DR [10]) en présentant les grilles de salaires moyens (en k€ par année) :

Grade	Min	Max
CR2	25	31
CR1	26	45
DR2	35	53
DR1	45	64
DR0	64	74

Il a également énoncé quelques conseils de recrutement en précisant que la quantité de publications n'était pas le critère central :

- faire la meilleure thèse possible ;
- réaliser une mobilité (géographique et/ou thématique) ;
- avoir effectué des transferts et de la médiation scientifique ;
- avoir un projet ambitieux mais crédible.

INRIA propose environ 70 post-docs actuellement.

CNRS

Philippe Baptiste, directeur de l'INS2I [11], a présenté la tradition d'excellence académique du CNRS en insistant sur son caractère inter-disciplinaire. Il a abordé la grande liberté dans la carrière et dans les sujets de recherche des chercheurs et des ingénieurs. Il a précisé qu'il y avait un grand besoin d'ingénieurs de recherche en informatique dans les laboratoires d'autres dominantes scientifiques. Il a également indiqué que le concours de recrutement des chercheurs était particulièrement difficile (30 à 50 candidats par poste de type CR). Il a terminé en indiquant qu'une expérience à l'étranger était absolument indispensable. Le CNRS a par ailleurs pour tradition de recruter beaucoup d'étrangers.

9. Chargé de recherche.
10. Directeur de recherche.
11. Institut des sciences de l'information et de leurs interactions.

CNU-27

Annick Montanvert, présidente du CNU-27, a présenté le CNU [12] et notamment la section 27 (Informatique) qui est la plus grande en terme de personnels : 3350 enseignants-chercheurs sur 49000 pour l'ensemble des sections CNU. Elle a précisé que 120 à 150 postes par an relevaient du CNU-27. Cette année, 812 demandes de qualification MCF [13] ont été déposées sur GALAXIE [14], 651 ont été traitées et 448 acceptées. Les demandes non traitées concernent des dossiers incomplets ou irrecevables. Elle a également rappelé les différentes étapes de la qualification :

(1) inscription sur ANTARES [15] (généralement en septembre/octobre) ;

(2) envoi du dossier (CV, descriptif d'activités, liste des publications, rapports de pré-soutenance, etc.) *avant* la date limite (généralement mi-décembre).

Recherche privée

SAP

Jean-Christophe Pazzaglia, directeur de SAP Labs France, est venu parler de la recherche privée. Il a précisé que le chercheur privé doit, en opposition au chercheur public, privilégier l'utile pour l'entreprise, les résultats étant à fournir par trimestre. Les entreprises prennent peu en compte les publications et s'intéressent d'avantage aux brevets. Il a notamment indiqué que les CVs étaient, en moyenne, évalués en première lecture par les DRH en 1 minute. Il faut donc mettre en avant les informations les plus importantes et privilégier les CVs courts. En ce qui concerne le salaire, un jeune docteur gagne l'équivalent du niveau DR2 INRIA mais évolue de manière moins linéaire.

12. Conseil national des universités.
13. Maître de conférences.
14. Portail du Ministère de l'enseignement supérieur et de la recherche.
15. Module du portail GALAXIE, dédié à la procédure de qualification.

Antoine Joux, Prix Gödel 2013

Jacques Stern [1]

Antoine Joux est Professeur associé à l'Université de Versailles Saint-Quentin-en-Yvelines et chercheur au sein de l'équipe cryptographie du laboratoire PRISM (UVSQ/CNRS). Le 3 juin 2013, il a reçu avec deux autres scientifiques américains le prix Gödel, qui récompense des travaux remarquables d'informatique théorique, lors de l'ACM Symposium on the Theory of Computing, à Palo-Alto (Californie).

Le prix Gödel 2013 a été décerné à Antoine Joux, Dan Boneh et Matt Franklin. C'est à la fois une juste reconnaissance de la profondeur des travaux d'Antoine Joux et de ses deux collègues, une véritable fierté pour la communauté des informaticiens français et, pour moi, la source d'une très grande joie. Antoine est en effet le premier de mes élèves qui a fait une thèse en cryptographie, au sein du Laboratoire d'informatique de l'École normale que nous avons rejoint pour ainsi dire ensemble. Je me souviens des heures passées en discussions autour de concepts mathématiques et d'implantations informatiques : Antoine avait déjà cette capacité étonnante de penser l'algorithmique de manière native, alors que, pour ceux venant d'une génération antérieure, il avait fallu un long apprentissage passant par l'étude des travaux des fondateurs. Les noms des fondateurs les plus emblématiques, Gödel et Turing, ont d'ailleurs été donnés aux deux grands prix internationaux d'informatique et ces prix sont allés tous deux cette année à des cryptologues, marquant ainsi la vitalité de cette discipline.

1. Professeur à l'École normale supérieure.

Paradoxe de ce prix Gödel : Antoine, Dan et Matt n'ont absolument pas travaillé ensemble mais ils ont, de part et d'autre de l'Atlantique, fait faire à la cryptologie deux pas de géant, qui ont permis, une fois mis bout à bout, de résoudre une conjecture ancienne et d'ouvrir un nouveau chapitre de la discipline. La conjecture avait été énoncée par Adi Shamir en 1984 lors du congrès Eurocrypt organisé à la Sorbonne, à quelques pas des lieux où Antoine a préparé sa thèse. Simple coïncidence bien sûr ! C'était huit ans après l'article de Whit Diffie et Marty Hellman *New Directions in Cryptology* [2], imaginant la cryptologie asymétrique et six ans après les travaux de Ron Rivest, Adi Shamir et Len Adleman, en donnant le premier exemple, le célèbre système RSA [8]. Comme on l'enseigne aujourd'hui dès le lycée, cette cryptologie permet de chiffrer un message à l'aide d'une clé publique accessible à tous ; le destinataire légitime seul peut reconstituer le message clair à l'aide d'une autre clé, appelée clé privée, mathématiquement reliée à la première, mais gardée quant à elle secrète. Dans le système RSA, la clé privée est choisie en premier et la clé publique s'en déduit. Il est donc impossible pour un utilisateur de choisir cette dernière a priori, par exemple en prenant son adresse mail ou son numéro de sécurité sociale. La conjecture de Shamir portait précisément sur l'existence de systèmes cryptographiques asymétriques dotés de cette propriété. C'est cette conjecture que les travaux de Joux, Boneh et Franklin ont résolue par l'affirmative et la branche de la cryptologie qu'ils ont ouverte a naturellement pris le nom de cryptologie fondée sur l'identité.

Pour comprendre les contributions des uns et des autres, il faut rappeler que Diffie et Hellman n'étaient pas parvenus à proposer un système asymétrique et qu'il avait fallu attendre l'invention de RSA pendant deux ans. Toutefois, ils avaient pu réaliser une fonctionnalité très proche : l'échange public de clé. Un tel échange se joue entre deux participants, chacun annonçant une donnée publique liée mathématiquement à des quantités gardées secrètes. Au terme de l'échange, les deux joueurs ont en commun une clé secrète commune, calculée à partir de leurs secrets respectifs et des données publiques. Il faut rappeler aussi que les recherches actives pour trouver d'autres exemples que le système RSA avaient conduit à l'utilisation des courbes elliptiques, comme proposé par Neil Koblitz [6] et Victor Miller [7] en 1985. Il faut rappeler enfin que certaines de ces courbes avaient été reconnues comme présentant des faiblesses, celles pour lesquelles on pouvait définir une opération dotée de propriétés algébriques remarquables, introduite dans les travaux d'André Weil dans les années 1940 et qui a depuis reçu le nom de couplage de Weil.

La découverte d'Antoine Joux procède d'un étonnant changement de perspective : utiliser le couplage non pour monter une attaque mais pour profiter de ses propriétés algébriques afin d'obtenir de nouvelles fonctionnalités. Par un choix extrêmement fin des paramètres, il a montré qu'il y avait bien un espace à l'abri des attaques connues, pour développer ces fonctionnalités et il en a donné un premier exemple en 2000 : l'échange triparti à la Diffie-Hellman [4]. Dans cet échange ce sont non plus

deux mais trois joueurs qui construisent une clé secrète commune après avoir diffusé chacun une unique annonce.

Le second pas vers la solution de la conjecture de Shamir résulte là encore, d'un surprenant changement de perspective opéré cette fois par Dan Boneh et Matt Franklin [1]. En 1984, Taher El Gamal avait montré comment rendre l'échange de clé de Diffie-Hellman dissymétrique pour en faire un cryptosystème à clé publique [3] ; un des joueurs – celui disposant de la clé secrète – restait stable, tandis que ceux qui lui adressaient des messages créaient des quantités secrètes éphémères. C'était un peu faire jouer Kasparov aux échecs contre le reste du monde ! Pour passer de l'échange triparti à un cryptosystème fondé sur l'identité, il faut ajouter un arbitre. Cet arbitre reste immuable ; Kasparov et les autres grands maîtres en reçoivent une clé dérivée de leur identité et ils jouent alors avec le reste du monde. Cependant, chaque partie, prise isolément, se joue à trois suivant les règles proposées par Antoine Joux. Ceux qui trouvent – sans doute à juste titre - l'analogie peu éclairante sont invités à lire les articles d'Antoine Joux *A One-Round Protocol for Tripartite Diffie-Hellman* [5] et de Boneh-Franklin *Identity-Based Encryption from the Weil Pairing* [1]. Ils y reconnaîtront le talent respectif des auteurs et ils verront la filiation entre les deux textes. Cette filiation est d'ailleurs avérée : en montant à la tribune du congrès Crypto durant l'été 2001, Dan a dit en souriant : je vais utiliser une hypothèse algorithmique nouvelle mais, d'une part, elle m'est utile et, d'autre part, un chercheur l'a utilisée avant moi. Ce chercheur c'était Antoine naturellement. Quant à l'hypothèse algorithmique, elle affirme qu'il est difficile de calculer la valeur d'un couplage de Weil, même en disposant d'autres valeurs du couplage prises sur des arguments liés par des relations algébriques simples. Comme indiqué plus haut, l'utilisation de cette hypothèse a véritablement ouvert un nouveau domaine de recherche.

Le prix Gödel est attribué pour une contribution en particulier et non pour un ensemble de travaux. Toutefois, je ne peux terminer sans mentionner que le registre d'Antoine n'est pas limité au couplage de Weil : il a notamment montré comment calculer explicitement des collisions pour certaines fonctions de hachage et il détient plusieurs records sur le calcul de logarithmes discrets. Dans les deux cas il a su combiner avec élégance idées mathématiques et calculs massifs. Pour conclure, je livre au lecteur une observation faite sur le site *Mathematics Genealogy Project* [2] : en notant DT la fonction qui associe à un chercheur son directeur de thèse, on a DT^5(Joux) = DT(Weil). Simple coïncidence encore ! Ou peut-être plus : bien que leur science soit plus récente, les membres de la communauté des informaticiens sont, au même titre que les collègues d'autres disciplines, les héritiers d'une tradition scientifique d'excellence. Le prix décerné à Antoine Joux fait honneur à cette tradition.

Références

2. http://genealogy.math.ndsu.nodak.edu/

[1] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. Proc. CRYPTO '2001, *Lecture Notes in Comput. Sci.* **2139**, 213-–229 (2001).

[2] W. Diffie and M. Hellman. New Directions in Cryptology. *IEEE Trans. Inform. Theory* **22**, 644–654 (1976).

[3] T. El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. Proc. CRYPTO '1984, *Lecture Notes in Comput. Sci.* **196**, 10–18 (1985).

[4] A. Joux. A One Round Protocol for Tripartite Diffie-Hellman. Proc. ANTS-IV, *Lecture Notes in Comput. Sci.* **1838**, 385–394 (2000).

[5] A. Joux. A One Round Protocol for Tripartite Diffie-Hellman. *J. Cryptology* **17**, 263–276 (2004).

[6] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation* **48**, 203–209 (1987).

[7] V. Miller. Uses of elliptic curves in cryptography. Proc. CRYPTO '1985, *Lecture Notes in Comput. Sci.* **218**, 417–426 (1986).

[8] R. Rivest, A. Shamir and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM* **21**, no. 2, 120–126 (1978).

Bilan du prix de thèse Gilles Kahn 2012

Nicole Bidoit

Le prix de thèse Gilles Kahn 2012, décerné par la SIF et patronné par l'Académie des Sciences, a été attribué à :

MATHIEU FEUILLET
pour sa thèse "Allocation de bande passante dans les grands réseaux stochastiques", soutenue à l'École Polytechnique, et préparée dans l'équipe-projet RAP (Inria Paris-Rocquencourt), sous la direction de Philippe Robert et Thomas Bonald.

Les deux deuxièmes prix sont décernés à (par ordre alphabétique) :

CAMILLE COUPRIE
pour sa thèse intitulée "Optimisation variationnelle discrète et applications en vision par ordinateur", soutenue à l'Université Paris Est, et préparée au Laboratoire d'Informatique Gaspard Monge (UMR 8049 - CNRS - UPEMLV - ESIEE - ENPC), sous la direction de Hugues Talbot, Laurent Najman et Leo Gardy,

MATHILDE NOUAL
pour sa thèse "Mise à jour de réseaux d'automates" soutenue à l'École Normale Supérieure de Lyon, et préparée au Laboratoire de l'Informatique du Parallélisme (UMR 5668 CNRS - ENS Lyon - UCB Lyon 1 - Inria), sous la direction de Sylvain Sené et Eric Rémila.

Les thèses des lauréats sont accessibles sur le site de la SIF [1].

36 candidats ont concouru pour le Prix de thèse Gilles Kahn 2012. Chaque dossier a été évalué par un expert extérieur et deux membres du jury auquel participent, pour trois ans, les lauréats du Prix des sessions antérieures.

La liste des 36 candidats est la suivante : Pierre Allain, Barbara André, Baptiste Caramiaux, Emmanuel Caruyer, Camille Couprie, Hadrien Courtecuisse, Cezara Dragoi, Swan Dubois, Alejandro Diaz-Caro, Mathieu Feuillet, Bruno Figliuzzi, Thomas Fuhr, Camille Guinaudeau, Mioara Joldes, Bilal Kanso, Roula Karam, Sébastien Kubicki, Claire Sondès Larafa, Bo Li, Muhammad Muzzamil Luqman, Odalric-Ambrym Maillard, Julien Marzat, Maxime Meilland, Marc Mezzarobba, Mathieu Morey, Perrin Nicolas, Mathilde Noual, Anne-Cécile Orgerie, Loïc Paulevé, Nicolas Philippe, Benoit Pradelle, Aristeidis Sotiras, B Srivathsan, Christophe Thovex, Yanjun Wang, John Whitbeck.

Merci à tous pour la qualité de vos travaux !

1. http://www.societe-informatique-de-france.fr/prix-these/historique.html

Allocation de bande passante dans les grands réseaux stochastiques

Mathieu Feuillet [1]

Mathieu Feuillet a soutenu sa thèse en juillet 2012 à Inria Paris-Rocquencourt, sous la direction de Philippe Robert (Inria) et Thomas Bonald (Telecom ParisTech). Il est actuellement chargé de mission à l'agence nationale de la sécurité des systèmes d'information (ANSSI).

L'objectif de cette thèse était d'étudier les performances d'algorithmes utilisés dans les réseaux à l'aide de modèles probabilistes. Cette approche permet de comprendre le fonctionnement des réseaux, de construire des méthodes pour dimensionner des systèmes et peut apporter une contribution à la conception d'algorithmes. L'utilisation des probabilités pour modéliser les réseaux est naturelle du fait de la nature intrinsèquement aléatoire du trafic, du caractère stochastique de certains algorithmes et des pannes et erreurs qui affectent les réseaux. Les premiers travaux en ce sens sont ceux d'Erlang [1] en 1909. Pour cette thèse, trois problèmes différents ont été abordés mais tous font intervenir un phénomène appelé *stochastic averaging* et des méthodes de renormalisation ont été abondamment utilisées dont certaines ont été développées pendant ce doctorat.

Le premier problème abordé lors de cette thèse a consisté à étudier le contrôle de congestion dans Internet. Il est communément admis dans la communauté réseau que la croissance de Internet n'aurait pas été possible sans l'utilisation du protocole

1. http://mathieu.feuillet.pro

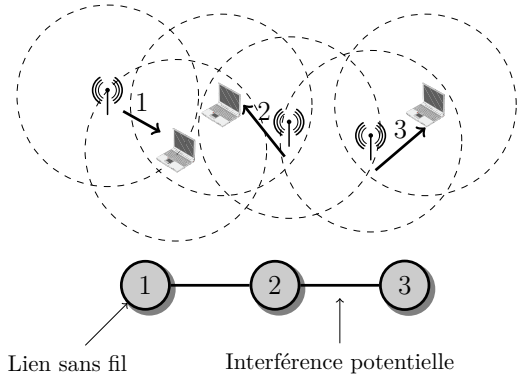


FIGURE 1. Un exemple de graphe d'interférence

TCP et notamment de sa fenêtre de congestion. Ce mécanisme permet de répartir les ressources disponibles entre les utilisateurs d'un réseau de manière complètement distribuée. En 2000, Massoulié et Roberts [3] ont proposé une nouvelle modélisation des réseaux qui a permis de mesurer les performances d'algorithmes tels que TCP à l'aide de modèles probabilistes et de prouver s'ils utilisent les ressources de manière efficace ou non. En 2009, nous avons utilisé ce type de modèle pour comprendre ce que serait Internet sans contrôle de congestion. Nous avons prouvé qu'il existe des cas où un effondrement des performances peut se produire entrainant un gaspillage des ressources disponibles. Néanmoins, nous avons également conjecturé qu'il existe une grande classe de réseaux où les ressources sont malgré tout utilisées de manière quasi-efficace, et ce d'autant plus que les débits maximaux des utilisateurs sont petits par rapport aux débits des équipements dans le cœur de réseau. Le début de cette thèse a été consacré à l'étude de cette conjecture sur une classe particulière de réseaux.

Le second problème était l'étude des performances de l'algorithme qui répartit la bande passante entre les utilisateurs dans le wifi : le CSMA/CA. Un modèle utilisant les graphes d'interférence (voir Figure 1) a été proposé en 1992 par Tassiulas et Ephremides [5]. Ils ont également prouvé l'existence d'un protocole qui garantit un usage efficace des ressources disponibles. Cependant, cet algorithme exige la présence d'une autorité centrale et revient à résoudre un problème NP-complet, ce qui le rend inutilisable en pratique. Plus récemment, différents progrès ont été réalisés, notamment par Shah et Shin [4] et par Jiang et Walrand [2] en 2008 qui permettaient de supputer l'existence d'un algorithme simple, complètement distribué et efficace. Dans cette thèse, nous avons prouvé que CSMA/CA n'est pas efficace et nous avons proposé une modification simple de cet algorithme qui le rend efficace.

Enfin, le troisième problème a consisté à étudier une table de hachage distribuée avec pertes pour estimer la vitesse de disparition des fichiers qu'elle contient. Il s'agit d'un domaine qui avait fait l'objet de peu d'études théoriques jusqu'à présent. Nous avons donc proposé un premier modèle très simple qui permet néanmoins de caractériser la vitesse de disparation des fichiers en fonction de paramètres comme le nombre de copies, la durée de vie d'une copie et la capacité de duplication des fichiers du système. Nous avons également pu caractériser le nombre maximum de fichiers qu'un tel système peut contenir au-delà duquel une fraction significative des fichiers est perdue dès la mise en route du système.

Références

[1] A.K. Erlang. The Theory of Probabilities and Telephone Conversations. *Nyt Tidsskrift for Matematik B* **20**, 33 (1909).

[2] L. Jiang and J. Walrand. A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks. In : *Allerton Conference on Communication, Control, and Computing* (2008).

[3] L. Massoulié and J. W. Roberts. Bandwidth Sharing and Admission Control for Elastic Traffic. *Telecommunications Systems* **15**, 185–201 (2000).

[4] D. Shah and J. Shin. Randomized scheduling algorithm for queueing networks. *The Annals of Applied Probability* **22**, no. 1, 128–171 (2012).

[5] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control* **37**, 1936–1948 (1992).

Optimisation variationnelle discrète et applications en vision par ordinateur

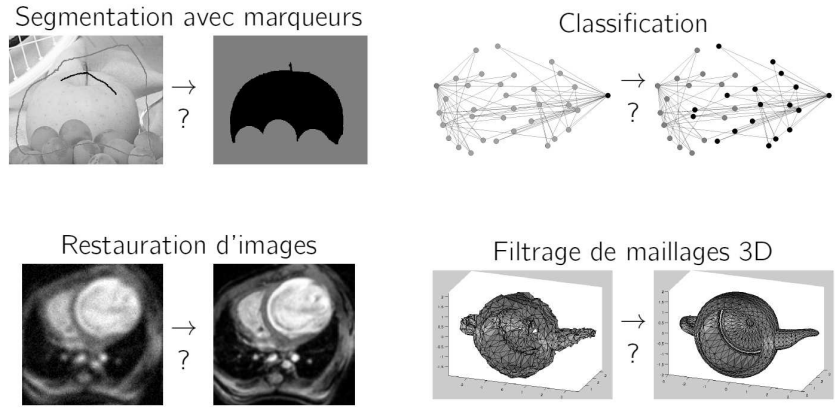
Camille Couprie [1]

Camille Couprie a soutenu sa thèse en octobre 2011 à l'université Paris-Est, thèse réalisée au sein de l'équipe A3SI du Laboratoire d'Informatique Gaspard Monge (LIGM), sous la direction de Hugues Talbot, Laurent Najman et Leo Grady. Elle a ensuite effectué un séjour post-doctoral au sein du Courant Institute of Mathematical Science (New York University) avec le Professeur Yann LeCun. Elle est actuellement Ingénieure de Recherche à IFP Energies Nouvelles.

Les besoins en traitement et analyse d'images sont présents dans de nombreux domaines (imagerie médicale, étude de matériaux, multi-média, etc.). Ces besoins incluent notamment la segmentation, c'est-à-dire l'extraction d'objets d'intérêt dans l'image, pour des applications de quantification, visualisation, etc. Pour traiter des images, une approche courante est de les assimiler à des graphes pondérés, les sommets du graphe étant localisés sur les pixels, et des arêtes reliant les pixels voisins.

Les méthodes basées sur les graphes pondérés permettent d'exprimer les informations nécessaires à la segmentation interactive comme des marqueurs, le gradient de l'image, ou la couleur des objets à segmenter. De ce fait, ces méthodes sont très générales et permettent l'optimisation de divers problèmes hormis la segmentation. On peut citer parmi les plus usitées les coupes minimales,

1. http://www.esiee.fr/~coupriec



le flot maximum continu, le marcheur aléatoire, et les plus courts chemins. Ces méthodes ont toutefois certains désavantages : les coupes minimales ont tendance à produire des contours suivant la grille carrée, ont une complexité super-linéaire, et sont limitées à la segmentation binaire dans le cadre classique.

La première partie de ma thèse a été consacrée à la proposition d'une nouvelle solution au problème de blocs produit par la méthode des coupes minimales pour la segmentation. Les coupes minimales peuvent être calculées au moyen de leur problème dual, consistant à calculer un flot maximum discret dans un graphe. L'introduction d'une nouvelle formulation du problème de flot maximum, basée sur le problème continu, nous permet d'éviter l'effet de blocs présent dans le cas des coupes minimales classiques. La méthode de point intérieur employée permet d'optimiser le problème plus rapidement que les méthodes existantes, et la convergence est garantie.

La formulation proposée a été ensuite efficacement étendue à la restauration d'image. Grâce à une approche duale contrainte et à l'emploi d'algorithmes proximaux parallèles, la méthode variationnelle proposée permet de restaurer des images, ainsi que des données arbitraires modélisées par des graphes, et préserve un meilleur contraste qu'avec la méthode de variation totale classique.

La seconde partie de ce doctorat a été consacrée à l'établissement de liens entre différentes méthodes de segmentation : les coupes minimales, le flot maximum continu, le marcheur aléatoire, et les plus courts chemins avec un algorithme de segmentation par ligne de partage des eaux (LPE).

La *Ligne de Partage des Eaux*, un algorithme de segmentation d'image classique dans des graphes, consiste à "inonder" l'image pour en extraire les contours dans les zones ou les différents bassins d'eau se rejoignent. Cette technique est très utilisée

en imagerie du fait de sa complexité linéaire et de sa capacité à segmenter les images en un nombre arbitraire de régions.

Ces liens ont inspiré un nouvel algorithme de segmentation multi-labels rapide produisant une ligne de partage des eaux unique, moins sensible aux fuites que la LPE classique. Nous avons nommé cet algorithme “LPE puissance”. L’expression de la LPE sous forme d’un problème d’optimisation a ouvert la voie à de nombreuses applications possibles au-delà de la segmentation d’images, en filtrage pour l’optimisation d’un problème non convexe, en stéréo-vision, et en reconstruction rapide de surfaces lisses délimitant des objets à partir de nuages de points bruités.

En post-doctorat à New York University dans le laboratoire d’apprentissage automatique du professeur Yann LeCun, nous avons travaillé à l’apprentissage des poids de nos graphes pour extraire *automatiquement* les contours des objets d’intérêt dans les images et *reconnaître* les catégories sémantiques de ces objets.

À présent ingénieure de recherche en traitement des images et du signal à l’IFP Energies nouvelles, mes recherches trouvent leurs applications en analyse de graphes pour la bio-informatique, en recalage de nuages de points pour des données chimiques, et en restauration et traitement de données sismiques.

Mise à jour de réseaux d'automates

Mathilde Noual

Mathilde Noual a soutenu sa thèse en juin 2012 à l'École Normale Supérieure de Lyon, thèse préparée au Laboratoire de l'Informatique du Parallélisme (LIP), sous la direction de Sylvain Sené et Eric Rémila. Elle effectue actuellement un stage post-doctoral au sein du Laboratoire d'Informatique, Réseaux et Systèmes (I3S) de Sophia-Antipolis.

Les réseaux d'automates booléens (RAB) sont des objets hérités des réseaux de neurones de la cybernétique [1] qui ont depuis été considérés sous des angles plus ou moins théoriques [2, 3, 6] et appliqués [4, 5]. Ils peuvent être vus à la fois comme des modèles de calcul (composés d'unités de calcul, boîtes noires a priori sans restriction, fonctionnant en réseau), et comme des modèles de systèmes d'interaction, spécialement de réseaux de régulation biologiques (composés d'entités différentes s'incitant les unes les autres à changer d'état). Leur simplicité inhérente permet de mettre en avant deux notions élémentaires : celle de *possibilité de changement*, ou *instabilité locale*, et celle de *travail en réseau*. Dans un RAB, mettre à jour l'état d'un automate capable de changement, c'est le faire obéir à ses influences actuelles et transformer sa possibilité de changement en un changement effectif. Choisir un mode de mise à jour revient donc à organiser dans le temps les événements locaux possibles.

Durant ma thèse j'ai cherché à relier des propriétés comportementales globales des RAB à des propriétés de leur structure d'interaction et de leur mode de mise à

jour. Une de ces questions récurrentes principales a été celle de la différence fondamentale existant entre les effets du *synchronisme* et de l'*asynchronisme*.

Forcer une mise à jour totalement asynchrone consiste à imposer que chaque automate soit mis au courant dès que le moindre changement atomique a lieu dans le réseau (et donc à interdire tout chevauchement d'événements durables). Le synchronisme total (mode parallèle) quant à lui, exploite systématiquement et immédiatement tout potentiel de changement. Quand la structure du RAB est un cycle orienté, le synchronisme maintient constante la quantité de changements possibles de sorte qu'il faut au moins un peu d'asynchronisme pour stabiliser le système localement *et* globalement. Pourtant, dans certains cas, les rôles peuvent être inversés : l'asynchronisme peut maintenir des instabilités asymptotiquement que seul un peu de synchronisme peut espérer stabiliser définitivement en interdisant à plusieurs automates d'avoir le temps de se communiquer leurs mouvements. Les conditions nécessaires à cette situation suggèrent toutefois que majoritairement, ajouter du synchronisme est ineffectif (cela ne fait que court-circuiter ou dévier des trajectoires asynchrones). Les "effets prédominants" du synchronisme et de l'asynchronisme diffèrent donc sensiblement selon les RAB et leur structure.

Sous l'hypothèse d'interactions *monotones*, une caractérisation et une comparaison combinatoires des comportements asymptotiques en parallèle des cycles isolés et tangents a permis de formuler et partiellement prouver la conjecture suivante : "*Intersecter un cycle à un autre réduirait dramatiquement la quantité d'instabilités locales et, par là, réduirait d'un facteur exponentiel le nombre d'attracteurs au profit des attracteurs* localement *plus stables*". Une interaction entre cycles consisterait donc essentiellement en une forte gêne mutuelle. Et même sous le mode de mise à jour le plus enclin à entretenir les instabilités locales le long des cycles, il semblerait que les intersections de cycles "forcent" l'asynchronisme. Ceci expliquerait le peu d'effet du synchronisme dans la plupart des cas. Dans les autres cas, différents résultats suggèrent qu'intrinsèquement, la sensibilité au synchronisme est liée à une circulation d'informations contradictoires entre deux automates (généralisation de la notion d'interaction *non-monotone*).

Références

[1] W. S. McCulloch and W. H. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5**, 115–133 (1943).

[2] Ch. Choffrut. Automata networks. *Lecture Notes in Computer Science* **316** (1988).

[3] E. Goles. Comportement dynamique de réseaux d'automates. Thèse de l'université scientifique et médicale de Grenoble (1985).

[4] S. A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology* **22**, 437–467 (1969).

[5] R. Thomas. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology* **42**, 563–585 (1973).

[6] F. Robert. *Discrete iterations : a metric study*. Springer Verlag (1986).

Prix de thèse Gilles Kahn 2013

Nathalie Bertrand

PRIX DE THÈSE GILLES KAHN 2013

patronné par l'Académie des Sciences et décerné par la SIF
http://www.societe-informatique-de-france.fr/prix-these/

Date limite de candidature : **15 septembre 2013**.

Le prix de la Société informatique de France (SIF, anciennement SPECIF) a été créé en 1998 pour récompenser chaque année une excellente thèse en Informatique. Gilles Kahn, qui a présidé les trois premiers jurys du prix, était convaincu de l'intérêt de promouvoir les jeunes talents les plus prometteurs de notre discipline. En son honneur, le prix a pris depuis 2007 le nom de Prix de thèse Gilles Kahn et est patronné par l'Académie des Sciences qui rend ainsi hommage à un de ses membres éminents.

La SiF souhaite par ce prix promouvoir toutes les facettes de l'informatique, des travaux fondamentaux aux travaux appliqués ayant donné lieu à transfert industriel, de ceux réalisés dans les grands centres à ceux réalisés dans des centres plus modestes. L'objectif de ce prix est de dynamiser et de motiver de jeunes chercheurs en les récompensant, et de faire connaître à l'ensemble de la communauté informatique d'excellents travaux de recherche. Un jury d'universitaires et de chercheurs, présidé par Michel Riveill, sélectionnera parmi les thèses soutenues au cours de l'année universitaire celle qui recevra ce prix. En outre, le jury pourra également distinguer, s'il le souhaite, au plus deux accessits.

La remise officielle du prix se fera début 2014 au cours d'une cérémonie associant la SIF et l'Académie des Sciences. À cette occasion, le récipiendaire se verra

remettre un chèque de 1500 euros et chacun des autres lauréats éventuels un chèque de 500 euros. Tous seront également invités à présenter leurs travaux à l'ensemble de la communauté scientifique. Sous réserve de remplir les conditions de candidature, les lauréats au prix de thèse Gilles Kahn seront considérés comme candidats à la nomination par Inria pour le prix Cor Baayen de l'ERCIM.

Les critères pris en compte par le jury pour sélectionner les lauréats sont notamment l'originalité des résultats, l'originalité du domaine et des méthodes utilisées, l'importance et l'impact des résultats obtenus, et bien évidemment la qualité de la rédaction puisqu'il s'agit de récompenser non seulement un travail mais surtout une thèse.

En 2012, sous la présidence de Nicole Bidoit, le jury était constitué de Xavier Allamigeon (prix 2010), Béatrice Bérard, Hugues Berry, Nathalie Bertrand, André-Luc Beylot, Nicole Bidoit, André Chailloux (prix 2011), Christine Collet, Hubert Comon, Cyril Gavoille, Mathieu Giraud, Jacques-Olivier Lachaud, Pierre Marquis, Christian Michel, Pascale Minet, Amedeo Napoli, Philippe Palanque, Jean Ponce, Michel Riveill, Isabelle Simplot-Ryl et Juan-Manuel Torres Moreno. Comme c'est l'usage, le jury sera partiellement renouvelé pour le prix 2013, le jury étant totalement renouvelé tous les trois ans.

Calendrier

- 15 juillet 2013 : ouverture de l'interface web de soumission
- 15 septembre 2013 : date limite de dépôt des candidatures
- 2 décembre 2013 : notification des résultats
- début 2014 : remise officielle du prix lors de l'Assemblée Générale de la SIF

Interface web de candidature

`http://prix-sif.inria.fr/`

Dossier de candidature

Peut candidater tout étudiant ayant soutenu son doctorat d'informatique dans une école ou université française entre le 1er septembre 2012 et le 31 août 2013. Toute candidature devra être explicitement soutenue par le directeur de thèse ou un des co-directeurs. Il n'est pas permis à un même encadrant de soutenir deux candidats.

Tous les documents doivent être déposés, sous forme de fichiers PDF exclusivement, par le biais de l'interface web. En cas de problèmes à utiliser l'interface, ou pour toute autre question concernant le prix, les candidats sont invités à contacter par courrier électronique la secrétaire du prix, Nathalie Bertrand (nathalie.bertrand@inria.fr).

Chaque dossier doit notamment comprendre :

- la thèse (en PDF),
- les rapports de pré-soutenance des rapporteurs (scannés, en PDF),
- le rapport de soutenance (scanné, en PDF),

– une lettre appuyant la candidature au prix de thèse, directement envoyée par le(s) directeur(s) de thèse,

– des rapports complémentaires que le candidat jugerait utile de fournir au jury, envoyés par les personnes concernées.

Le formulaire en ligne demande également de saisir certaines informations : un résumé de 2 pages de la thèse, un CV d'une page maximum ainsi qu'une liste de publications.

Un problème à résoudre de tête, et un autre pour faire rougir vos microprocesseurs

Jean-Paul Delahaye [1]

La rubrique récréation informatique aura une double fonction. D'une part, on y posera de petites énigmes algorithmiques ou liées aux aspects combinatoires de l'informatique dont la solution sera donnée dans le numéro suivant du bulletin... pour ceux qui ne les auront pas résolues. D'autre part on y soumettra des problèmes impliquant un vrai travail de programmation et conduisant sans doute parfois à de longs calculs... pour vos machines. Le but sera de résoudre un problème dont la solution n'est pas connue (voir un exemple plus bas), ou de battre un record. Les contributions des lecteurs seront cruciales pour cette seconde catégorie de problèmes.

Trouvez l'algo

La petite énigme que je soumets est due à John Conway, le fameux mathématicien inventeur du Jeu de la vie et de bien d'autres merveilles amusantes et sérieuses. Il paraît que cette énigme a le pouvoir de clouer sur leur chaise ceux qui cherchent à la résoudre, et qui y restent un long moment sans bouger.

Une liste de noms — longue mais finie — est lue à voix haute, certains noms sont répétés plusieurs fois. Vous écoutez cette liste, et votre mémoire assez limitée ne vous permet que de mémoriser un seul nom à la fois (que vous pouvez comparer à celui qui vient ensuite) et un entier qui peut vous servir de compteur (vous ne pouvez que l'augmenter ou le diminuer d'une unité à la fois). La liste lue comporte

1. Université des Sciences et Technologies de Lille, Laboratoire d'Informatique Fondamentale de Lille, UMR 8022 CNRS, Bât M3-ext, 59655 Villeneuve d'Ascq Cedex. E-mail : delahaye@lifl.fr.

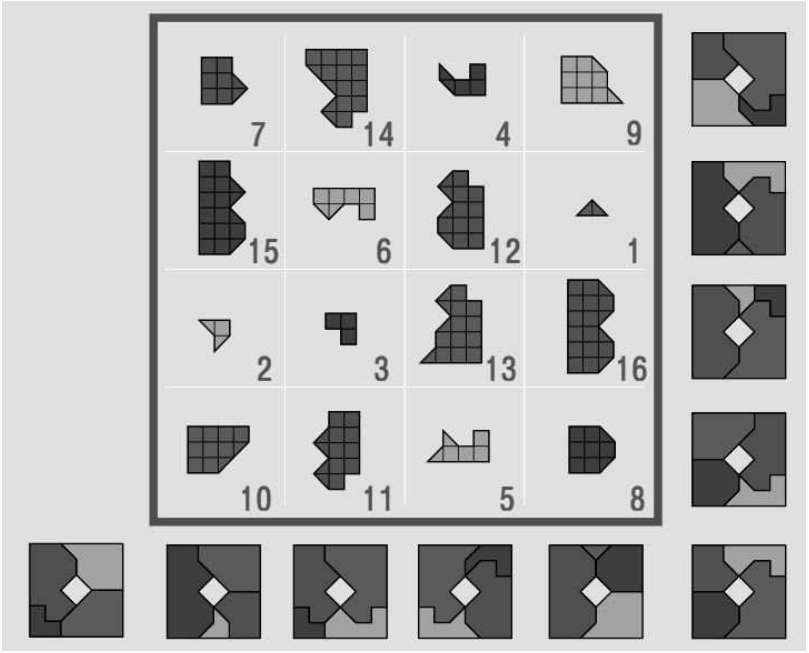
un nom majoritaire (il apparaît dans plus de la moitié des cas). Comment devez-vous procéder pour que lorsque la lecture se terminera vous sachiez quel est ce nom majoritaire ?

Et maintenant beaucoup plus difficile.

Carré géomagique

Lee Sallows a inventé une nouvelle catégorie d'objets combinatoires dont on n'a sans doute pas fini d'entendre parler : les carrés géomagiques[2].

Il s'agit de tableaux de formes géométriques toutes différentes qui — comme pour les carrés magiques — donnent toujours la même chose quand on additionne les éléments d'une ligne, d'une colonne ou d'une des deux diagonales. Le mot "additionner" voulant dire ici qu'on juxtapose les formes (sans recouvrement et en les retournant si nécessaire). Un exemple de carré géomagique d'ordre 4 fera comprendre l'idée.

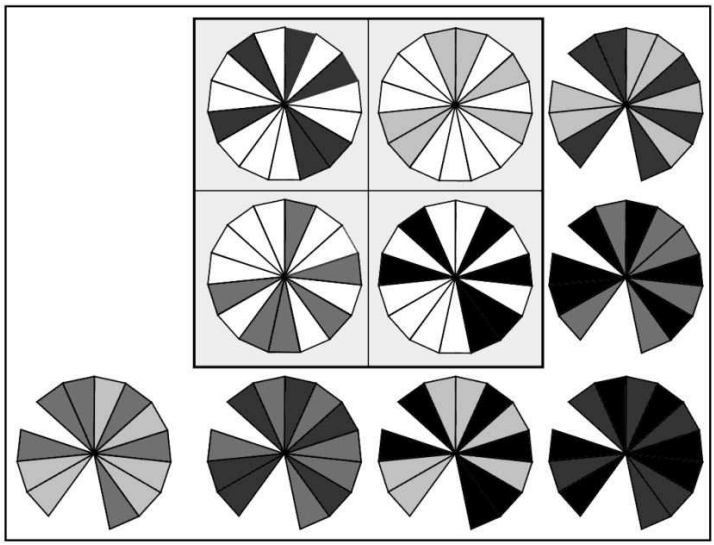


2. Voir : http ://www.geomagicsquares.com/ ou Lee Sallows, Geometric Magic Squares : A Challenging New Twist Using Colored Shapes Instead of Numbers, Dover Publications, 2013.

Il y a 16 formes différentes disposées dans une grille 4×4. Les quatre formes de la première ligne s'assemblent et donnent une forme assez simple (ici un carré troué par un carré plus petit tourné de 45° par rapport à l'horizontale). De même les quatre formes de la seconde ligne donne le même carré troué. Et c'est le cas encore pour les deux autres lignes, les 4 colonnes, et les deux diagonales. On remarque que lorsque les formes utilisées ont des aires différentes — c'est le cas ici — le tableau des aires est un carré magique. La condition que les aires soient différentes n'est pas dans la définition des carrés géomagiques qui exige seulement que les formes de base dans la grille soient deux à deux distinctes.

Contrairement aux carrés magiques qui n'existent qu'à partir de l'ordre 3 (comme on le démontre sans mal), il est possible de concevoir des carrés géomagiques d'ordre 2 : quatre figures différentes qui se recomposent en une même figure de 6 façons (deux lignes, deux colonnes, deux diagonales).

Il se trouve que les seules solutions trouvées sont des variantes évidentes de celle-ci :



Cette solution est assez insatisfaisante car les 4 figures de base sont "décomposées" (chacune est formée de plusieurs morceaux reliés par un seul point central). Il en va de même pour la figure "somme" qui est en deux morceaux.

Le problème — ouvert aujourd'hui — est :

existe-t-il quatre figures non décomposées constituant un carré géomagique d'ordre 2, ou au moins telles que la figure "somme" ne soit pas décomposée ?

Le problème est assez récent, il est donc possible qu'il existe une solution simple. À vos programmes ! Il se peut cependant qu'il n'y ait pas de solutions, mais alors on aimerait en avoir la démonstration.

Institut Henri Poincaré,
11 rue Pierre et Marie Curie,
75231 Paris Cedex 05