

Vers une formation des enseignants chargés de la spécialité *Informatique et sciences du numérique* en terminale S

Dans un précédent document *Proposition de programme de formation pour les enseignants chargés de la spécialité Informatique et sciences du numérique en terminale S*, nous avons réfléchi, à la demande du cabinet du Ministre de l'Éducation Nationale, à un programme de formation pour les enseignants chargés de la spécialité *Informatique et sciences du numérique* en terminale S à la rentrée 2012. Ce document partait de l'hypothèse d'un volume horaire pour cette formation de 432 heures (une journée de six heures par semaine, pendant deux années de trente-six semaines). Nous avons découpé ces 432 heures en douze cours de 36 heures chacun.

L'informatique étant une science structurée autour de quatre concepts, ceux d'information, de langage, d'algorithme et de machine et une règle importante dans l'organisation de tout enseignement d'informatique, qu'il soit destiné aux lycéens ou à leurs professeurs, étant de veiller à un équilibre entre ces quatre notions, nous avons organisé ces douze cours en quatre groupes de trois cours, chacun centré sur l'une de ces notions. Nous avons ensuite réparti ces douze cours par niveau en deux groupes : six cours pour la première année et six cours pour la seconde.

Information

- Représentation numérique de l'information (36h, année I)
- Structuration et contrôle de l'information (36h, année I)
- Bases de données et systèmes d'information (36h, année II)

Langage

- Langage de programmation (36h, année I)
- Automate et grammaire (36h, année II)
- Compilation et vérification (36h, année II)

Algorithme

- Algorithmes classiques (36h, année I)
- Calculabilité et complexité (36h, année II)
- Algorithmes à grande échelle (36h, année II)

Machine

- Architecture de machine (36h, année I)
- Réseaux (36h, année I)
- Systèmes d'exploitation (36h, année II)

Si ce programme reste l'objectif à atteindre, diverses voies sont possibles, et nous réfléchissons, dans ce document, sous l'hypothèse que, dans une phase transitoire, cet enseignement pourrait être réduit de

moitié, à 216 heures. Nous proposons donc de garder les six cours de l'année I pour cette formation initiale, les six cours de l'année II faisant alors l'objet de modules complémentaires de 36 heures, organisés au-delà des deux premières années de formation des enseignants. Pour les enseignants qui ont déjà acquis ailleurs les connaissances de l'année I, il est souhaitable, à l'inverse, de garder les six cours de l'année II pour cette formation.

Les six cours de l'année I abordant les notions les plus élémentaires de l'informatique, nous avons associé à chacun d'eux quelques éléments de réflexion pédagogique sur l'enseignement de ces notions.

Chaque cours est typiquement constitué de douze cours magistraux d'une heure et demie chacun et de douze séances de travaux dirigés ou de travaux pratiques d'une heure et demie également. Certains cours demandent une séance de travaux dirigés, mais la plupart demandent plutôt une séance de travaux pratiques, devant un ordinateur. Ces séances de travaux pratiques permettent aux enseignants en formation à la fois de mettre en œuvre les notions vues en cours et de consolider leur savoir-faire. À partir d'un certain moment dans la formation, ces séances de travaux pratiques s'enchaînent naturellement en mini-projets. Chaque cours se conclut par un examen qui peut prendre plusieurs formes : sur table, oral, validation de projet, ...

Il est important de veiller à mettre en valeur les nombreuses articulations entre les différents concepts théoriques et pratiques présentés dans les différents cours. Ces articulations se placent à la fois au niveau conceptuel (la programmation fait appel aux concepts de langage et d'algorithme, la compilation aux concepts de langage et de machine, ...) et au niveau technique (ces concepts étant utilisés conjointement dans la conception des applications modernes et des objets numériques qui nous entourent). Une application suffisamment riche (un système de modélisation / simulation d'un processus physique, biologique ou démographique, un jeu, un smartphone, ...) peut servir de fil rouge à l'ensemble de la formation, chaque cours détaillant la manière dont les notions introduites dans ce cours sont utilisées dans la conception de cette application.

Certaines parties du cours pourront aussi faire appel à des conférences vidéo ou des webinars (séminaires sur le web).

1 Les cours

1.1 Représentation numérique de l'information (36h)

Codage numérique du texte (toutes langues), de l'image, du son, de la vidéo, de la géométrie, des forces, ... Exemples d'opérations numériques : inversion vidéo d'une image, augmentation du contraste.

Interfaces, capteurs et actionneurs, orientés hommes (clavier, souris, écran) ou physique (CCDs, accéléromètres, ...). Valeurs numériques et événements abstraits associés.

Densité d'information. Compression générique et spécifique. Compression maximale, complexité de Kolmogorov. Code préfixe, méthode de Huffman. Compression d'une image par arbre quaternaire. Codage cryptographique.

Avec cette notion d'information, une question pédagogique essentielle est celle du lien entre les

concepts scientifiques du cours et le monde numérique environnant. Pour illustrer un algorithme de compression d'images, par exemple, il est important de choisir une image réelle, et non un exemple artificiel conçu pour ce cours, ou mieux laisser les élèves apporter leur propre image – une image qu'ils ont trouvée sur le web ou une photo qu'ils ont prise avec leur appareil. Il est important de montrer également comment ces algorithmes de compression, de chiffrement, ... peuvent être intégrés dans l'appareil photo ou dans des outils externes, par exemple de courrier électronique ou de chat.

1.2 Structuration et contrôle de l'information (36h)

Structures de données de base : types numériques, enregistrements, ... Vision abstraite et indépendance de l'implémentation par l'introduction d'API (Application Programming Interface). Persistance de données : linéarisation et sauvegarde en fichiers.

Structures universelles du web : les langages HTML et XML, les URL et DNS, les pages web statiques et dynamiques (PHP-MySQL).

Protection de données. Où les informations sont-elles stockées ? Contrôle des accès et pare-feux. Licences. Propriété intellectuelle. Respect de la vie privée. Droit à l'oubli.

Aborder la question du web en classe au lycée mène à aborder des questions éthiques et juridiques, qui sont l'occasion d'un travail interdisciplinaire. Toutefois, il est important de susciter une réflexion de la part des enseignants, sur la place à donner à ces questions, et sur les méthodes pédagogiques à employer, par exemple, sur la place des exposés ou des travaux personnels encadrés dans la transmission de ces connaissances.

1.3 Langage de programmation (36h)

Le noyau impératif (affectation, séquence, test, boucle, déclaration). La notion d'état et de transformation d'état. Fonction. Récursivité. Allocation. Type de données dynamique. Partage, copie. Objet, module.

Méthodes et outils de développement (méthodes de debugging, éléments de génie logiciel).

Un programme décrit un algorithme, manipulant de l'information, exprimé dans un langage, afin d'être exécuté par une machine. De ce fait, la programmation est la clé de voûte d'un enseignement élémentaire de l'informatique. Toutefois, c'est un ensemble de savoirs et de savoir-faire qui s'enseigne d'une manière très différente de beaucoup d'autres, puisque les travaux pratiques jouent un rôle essentiel dans l'acquisition des compétences associées. Il est donc nécessaire de donner une place importante à la réflexion pédagogique sur cette activité. Une partie de cette réflexion pédagogique concerne l'organisation de ces travaux pratiques : comment mener une séance ? Avec quel fil rouge ? Que distribuer aux élèves ? Jusqu'à quel point faut-il les laisser chercher ? Comment partager son temps entre les différents élèves lors d'une séance de travaux pratiques ? Comment favoriser l'aide des élèves par les élèves ? À quel moment de l'année faire commencer un projet informatique par les élèves ? Quel langage choisir ? Il est également nécessaire de susciter une réflexion sur la place des enseignements magistraux et sur leur articulation avec les travaux pratiques : comment montrer en

cours magistral des exemples de l'activité de programmation, sans être redondant avec une séance de travaux pratiques ? Des questions pédagogiques plus en amont doivent également être abordées, par exemple : comment verbaliser ce qui se passe quand on exécute un programme ? Comment transmettre l'idée que la rigueur syntaxique et sémantique exigée dans un programme est une condition nécessaire pour que ce programme fonctionne ?

1.4 Algorithmes classiques (36h)

Tri, recherche en table. Parcours d'arbres en profondeur et en largeur d'abord. Arbres de recherche. Méthode min/max. Parcours de graphe en profondeur et en largeur d'abord. Algorithme RSA. Notion de protocole cryptographique. Exemples d'algorithmes géométriques (enveloppe convexe, algorithme de Bresenham, ...).

Un enseignement de l'informatique au lycée ne demande pas l'apprentissage successif de nombreux algorithmes classiques. En revanche, il est important de sensibiliser les élèves à la nature algorithmique ou non d'une définition. Si on imagine une suite infinie, comme la suite des décimales du nombre π , il y a un algorithme simple qui permet de décider s'il y a un 7 dans les dix premiers éléments de la suite ou non, puisqu'il suffit d'examiner ces dix éléments l'un après l'autre. En revanche, il n'y a pas d'algorithme qui permette de décider s'il y a un 7 dans la suite en entier ou non, puisque cela demande, dans certains cas, d'examiner l'intégralité de la suite. De même, il est important de faire comprendre que certains problèmes qui nous paraissent simples sont difficiles à résoudre algorithmiquement. Par exemple, nous savons tous à peu près reconnaître si une image contient un cercle ou non. En revanche, construire un algorithme qui décide de cela est très difficile. Un autre exemple, est celui de savoir si une phrase est grammaticale en français ou non, comme en témoigne la difficulté à écrire des programmes de correction orthographiques. D'un point de vue pédagogique, le but de cette réflexion doit être d'amener les élèves à s'émerveiller à nouveau de phénomènes qui leur paraissent banals, tels la reconnaissance d'un cercle dans une image, de la grammaticalité d'une phrase.

1.5 Architecture de machine (36h)

Circuits Booléens : calcul d'une fonction booléenne à l'aide de portes logiques. Circuits associés aux automates d'états finis.

Composants synchrones : processeurs, DSPs, mémoires, gestionnaires mémoires, interfaces (USB, GSM, ...). Circuits programmables (FPGAs). Composants analogiques : convertisseurs analogique / digital, radios, capteurs. Machines de Moore et de Mealy.

Structure d'un processeur : modèle de Von Neumann (mémoire, unité arithmétique et logique, contrôleur, entrée/sorties). Langage machine.

Composition de composants dans les Systèmes sur puces : mémoires distribuées, bus, files d'attente multi-horloges. Streaming pour les signaux audio et vidéo.

Sur le plan pédagogique, une difficulté est apparue récemment par la complexification des machines. L'image que nous pouvons transmettre d'un ordinateur – avec son microprocesseur, sa mémoire et ses

périphériques – est devenue une abstraction parmi d'autres, qui ne correspond plus à la réalité d'un ordinateur. Cependant, cette abstraction, comme d'autres, est utile pour la verbalisation de ce qu'il se passe quand on exécute un programme. Une autre difficulté est apparue par la diversification des machines. À côté des ordinateurs sont apparus d'autres types de machines : d'une part les réseaux, d'autre part des petits objets munis d'une véritable puissance de calcul, tels les téléphones, les appareils photos, ... Un objectif est ici de faire comprendre aux élèves l'unité qu'il y a derrière cette diversité : toutes ces machines sont des systèmes matériels, donc régis par les lois de la physique, qui traitent de l'information de manière réellement uniforme. L'histoire de ces machines doit être replacée dans celle des machines en général – qui comprennent les machines à vapeur ou les postes de radio – tout en insistant sur la rupture en termes de complexité et d'universalité qu'introduisent les ordinateurs dans cette histoire.

1.6 Réseaux (36h)

Notion de paquet et de protocole. Transmission d'information point à point, détection et correction d'erreurs de transmission. Réseaux local, réseau global, détection et correction d'erreurs réseau. Routage, nommage, TCP / IP. ADSL. Décomposition en couches. Équipement (hub, switch, routeur, ...). Qualité de service. Outils de développement distribué.

Une difficulté pédagogique dans l'enseignement de la notion de réseau est que les travaux pratiques deviennent plus difficiles à organiser : d'une part, ils demandent d'utiliser plusieurs machines en réseau. D'autre part, écrire des programmes distribués – qui exploitent un réseau – est plus difficile. C'est l'occasion d'utiliser d'autres méthodes pédagogiques, qui utilisent des outils de plus haut niveau, comme des plateformes de simulation, qui donnent une place plus importante à l'observation qu'à la construction ou qui exploitent les possibilités du cours magistral.

Questions pédagogiques transverses : comment équilibrer son enseignement entre les quatre concepts de langage, d'algorithme, de machine et d'information ? Dans quel ordre traiter les différentes parties du programme ? Comment intégrer la perspective de l'épreuve du bac et de ses modalités ? Quelles parties du cours faire en direct, quelles parties faire à l'aide de supports externes (vidéos, applications Web, ...) ?

2. Questions à aborder dans la suite de la formation

2.1 Bases de données et systèmes d'information (36h)

Algèbre relationnelle, calcul relationnel, théorème d'équivalence. SQL, requêtes et mises-à-jour. Structure d'accès, table de hachage et arbre B. Optimisation de requêtes. Conception de schéma, UML, dépendances fonctionnelles et multivaluées. Concurrence et distribution.

Architecture fonctionnelle des systèmes d'information (décomposition descendante, modularité, service, processus). Modélisation et architecture des données (UML, distribution, transaction). Fiabilité du système d'information (disponibilité, durée de reprise, ordres de grandeur, analyse d'ensemble). Performance (débit, latence, files d'attente).

2.2 Automate et grammaire (36h)

Automate d'états finis, application (interface homme-machine, contrôle discret, protocole, ...). Expression régulière. Grammaire. Langue naturelle et langage formel (classification de Chomsky). Principes d'un analyseur syntaxique simple. Algorithmique du texte (recherche d'une sous-chaîne, appartenance à un dictionnaire, algorithmes sur le génome).

2.3 Compilation et vérification (36h)

Sémantique opérationnelle à petits et grands pas. Réalisation d'un interpréteur pour un langage simple. Machine abstraite, transformation de l'interpréteur en compilateur. Principes d'optimisation.

Test, vérification, preuve.

2.4 Calculabilité et complexité (36h)

Complexité en temps et en espace. Complexité en moyenne et dans le pire cas. Complexité des algorithmes de tri et borne inférieure. Classes de complexité P, EXPTIME, PSPACE. Définition de la notion de fonction calculable. Réécriture, machines de Turing, Turing-complétude. Calculabilité de l'interpréteur. Indécidabilité du problème de l'arrêt. Machine de Turing non déterministe. La classe NP et ses centaines d'algorithmes pratiques. Le problème $P \neq NP$.

2.5 Algorithmes à grande échelle (36h)

Moteur de recherches. Diffusion pair à pair. Cloud computing. Évaluations probabilistes de complexité.

2.6 Systèmes d'exploitation (36h)

Différents types de systèmes d'exploitation (interactif, batch, embarqué dédié, temps réel). Principes de gestion de ressources (abstraction, sécurité, virtualisation). Interface programmes/noyau (appel système, abstraction de périphérique). Interfaces noyau/matériel (interruptions, projection d'entrées-sorties en mémoire). Composants du noyau (gestion mémoire, ordonnancement, système de fichiers, pilotes de périphériques). Exclusion mutuelle, algorithme de Peterson. Interface homme-machine. Système multi-fenêtre. Éléments de gestion des multiprocesseurs.

EPI et groupe ITIC de l'ASTI (mai 2010)

- Jean-Pierre Archambault, Président de l'EPI
- Gérard Berry, Membre de l'Académie des Sciences, Professeur au Collège de France
- Gilles Dowek, Professeur à École Polytechnique
- Maurice Nivat, Membre correspondant de l'Académie des Sciences