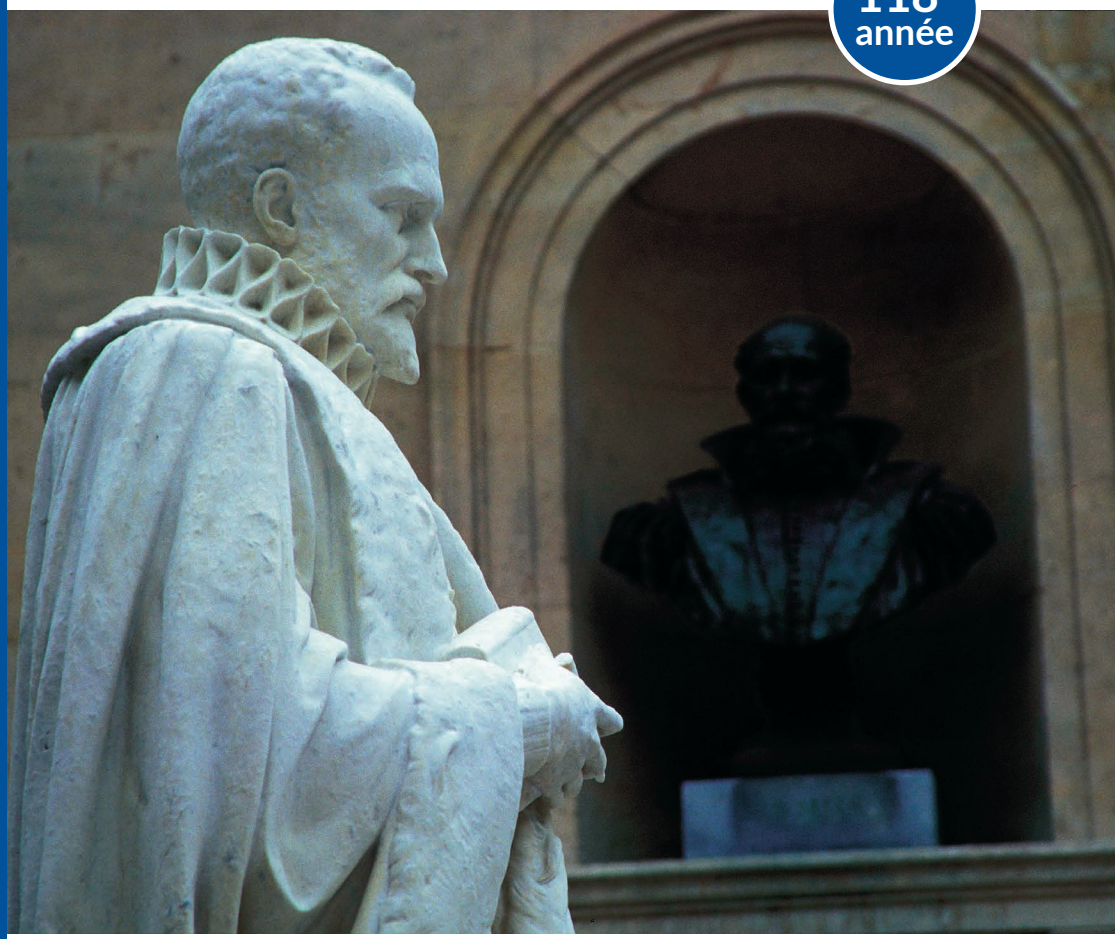
ANNUAIRE du **COLLÈGE DE FRANCE** 2017 - 2018

Résumé des cours et travaux

118e année



COLLÈGE DE FRANCE
1530

ALGORITHMES, MACHINES ET LANGAGES

Gérard BERRY

Membre de l'Institut (Académie des sciences)
et de l'Académie des technologies, professeur au Collège de France

Mots-clés : algorithmes, Esterel, temps réel, circuits digitaux, vérification formelle

La série de cours et séminaires « Esterel de A à Z » est disponible, en audio et/ou en vidéo, sur le site internet du Collège de France (https://www.college-de-france.fr/site/gerard-berry/course-2017-2018.htm) ainsi que le colloque « Imagerie médicale et apprentissage : vers une intelligence artificielle ? » (https://www.college-de-france.fr/site/gerard-berry/symposium-2017-2018.htm).

ENSEIGNEMENT

COURS ET SÉMINAIRES – ESTEREL DE A À Z

Introduction

Ce cours a présenté en détail le langage de programmation synchrone Esterel, déjà décrit de façon succincte dans deux séances en 2012-2013. Esterel est destiné aux systèmes réactifs, c'est-à-dire aux systèmes informatisés capables de réagir au temps et aux événements provenant de l'extérieur en étant soumis à des contraintes de temps de réponse. Ces systèmes se trouvent en particulier dans l'informatique embarquée destinée à conduire et surveiller les objets informatisés (avions, voitures, etc.), dans les circuits électroniques digitaux, et dans d'autres dispositifs comme les interfaces homme-machine et les protocoles de communication des réseaux. Esterel permet de les programmer de façon sûre, en donnant de plus accès à des systèmes de vérification automatique d'absence de bugs. Le cours a expliqué en détail ses points essentiels : choix syntaxiques, style de programmation, chaînes de sémantiques mathématiques conduisant de la spécification formelle à l'implémentation, traduction en circuits digitaux ou en code C exécutable, et vérification formelle des programmes. La spécification rigoureuse du langage et ses implémentations directement fondées sur sa sémantique formelle ont joué un rôle crucial dans les développements académiques

et industriels du langage et de son outillage. Industrialisé par Esterel Technologies, Esterel a reçu des applications industrielles dans des domaines variés : avionique, interfaces homme-machine, synthèse et vérification de circuits électroniques, etc. Pour le langage SCADE 6 de la société Esterel technologies, Esterel a été unifié avec Lustre, autre langage synchrone développé à Grenoble. Il sert aussi de base à une extension de JavaScript nommée HipHop, qui fait actuellement l'objet de mes recherches en collaboration avec Manuel Serrano à l'Inria Sophia-Antipolis, ainsi qu'à d'autres langages à travers le monde.

Cours 1 – Principes, idées et styles pour la programmation réactive

Ce premier cours a d'abord présenté les raisons qui ont conduit à la création et à l'approfondissement des langages parallèles synchrones, dédiés à la programmation des systèmes dits « réactifs », dont la fonction est de maintenir une interaction constante et temporellement précise avec leur environnement, Il a ensuite étudié l'évolution de ces langages, et en particulier la progression de leurs styles de programmation, depuis les styles initiaux d'Esterel et Lustre inventés au milieu des années 1980 jusqu'au langage industriel SCADE 6 d'Esterel technologies. Ce dernier est devenu dans les années 2000 leader pour les systèmes embarqués certifiés en avionique et dans d'autres domaines où la sûreté de l'informatique est critique. Les avancées ont été dues soit à de nouveaux résultats scientifiques, soit au besoin de résoudre des problèmes industriels difficiles. Le cours a mis en valeur les nombreuses influences externes qui ont enrichi les langages synchrones, en particulier par l'introduction de la programmation graphique ou mixte textuelle/graphique, tout en les gardant très différents des langages parallèles asynchrones classiques. Il a étudié l'opposition entre l'approche synchrone, qui réconcilie mathématiquement et pratiquement parallélisme et déterminisme comportemental, et d'autres approches qui cherchent à marier la programmation réactive avec la programmation séquentielle classique ou avec la programmation parallèle asynchrone non déterministe. Les inconvénients de ces approches deviennent rapidement très gênants pour toute application sérieuse. Le cours a finalement montré comment on peut assembler à haut niveau des composants synchrones à l'aide de communications asynchrones, construisant ainsi des systèmes GALS (*globally asynchronous locally synchronous*) bien adaptés par exemple à la robotique, aux systèmes électroniques sur puces et à la programmation du Web.

Cours 2 – Sémantiques, causalité et constructivité des langages synchrones

Ce deuxième cours a été consacré aux mathématiques des langages synchrones en général et d'Esterel en particulier, centrées autour de la notion de causalité de la transmission de l'information dans les programmes. L'étude fine de la causalité dans le cadre synchrone conduit à une chaîne de sémantiques formelles de plus en plus fines ; cette chaîne se termine par la sémantique constructive, fondée sur des idées de la logique éponyme et aussi justifiée par des considérations physiques sur la propagation des fronts électriques dans les circuits, déjà présentées dans le cours de 2013-2014. Ces constructions sémantiques ont toujours sous-tendu les développements pratiques des différentes versions d'Esterel, leurs assises théoriques ayant permis de les garder toujours compatibles entre elles. Elles ont servi de juges

de paix pour toutes les décisions importantes. Cependant, d'autres auteurs ont développé des sémantiques plus restreintes pour rendre le style d'Esterel plus compatible avec les langages classiques, avec en premier le langage Reactive C de Frédéric Boussinot qui intègre de façon simple et élégante certaines des notions d'Esterel dans C et a inspiré plusieurs autres langages, dont ReactiveML présenté en séminaire le même jour. Des extensions sémantiques ont aussi été développées plus récemment, comme celles utilisées par le langage Scl qui a été présenté dans le séminaire 4 du 14 mars 2018.

Séminaire 2 – ReactiveML, programmation réactive d'applications non temps-réel

Louis Mandel (chercheur à l'IBM Thomas J. Watson Laboratory, États-Unis)

Ce séminaire a présenté le langage ReactiveML, qui intègre les idées de ReactiveC dans le langage fonctionnel OCaml. ReactiveML ne vise pas les applications temps-réel, mais des applications logicielles où la souplesse de la programmation fonctionnelle permet d'introduire élégamment beaucoup plus de dynamicité et d'évolutivité dans le modèle réactif synchrone. Ce point a été illustré par de nombreux exemples hors de portée des langages synchrones classiques.

Cours 3 – Traduction d'Esterel en circuits

Ce troisième cours a présenté la traduction des programmes Esterel en circuits booléens optimisés. Véritable révolution conceptuelle par rapport aux approches précédentes fondées sur la théorie des automates, elle est née lors d'une collaboration avec l'équipe de Jean Vuillemin à Digital Equipment qui cherchait à mieux spécifier les parties contrôles des circuits électroniques reconfigurables par logiciel (FPGAs). Elle a immédiatement joué un rôle majeur en faisant complètement disparaître les explosions exponentielles de la taille du code généré par les premières implémentations à base de génération d'automates finis, qui limitaient fortement leur usage. Elle a ainsi permis aux industriels d'utiliser Esterel pour des applications commerciales en circuits électroniques ou en logiciel embarqué, ce qui a provoqué la création de la société Esterel technologies. Le cours s'est limité aux programmes sans boucles, en présentant le principe général de la traduction, celle de chaque instruction, puis la connexion avec la sémantique constructive. Le cas des boucles, plus délicat, a été traité dans le cours suivant. Le cours s'est terminé par un rappel des méthodes d'optimisation booléennes présentées en 2013-2014, qui permettent d'optimiser les circuits en atteignant des niveaux de performance souvent hors de portée des méthodes humaines classiques.

Séminaire 3 – Le langage Signal et ses applications

Albert Benveniste et Thierry Gautier (directeurs de recherche à l'Inria Rennes Bretagne-Atlantique)

Conçu et développé à la même période que les autres langages synchrones, Signal, avec le recul, apparaît comme étant à part dans le paysage. Il est dédié aux systèmes dits « ouverts » et se focalise sur la synchronisation et les horloges. La synchronisation d'un programme est synthétisée et non pas vérifiée. L'exposé a

expliqué les principes de cette approche et, en premier, le calcul d'horloges et de causalité qui permet de dégager une notion d'interface temporelle utile pour la compilation modulaire. Au-delà de la présentation du langage, l'objectif a été de montrer l'intérêt de ces principes. L'exposé a expliqué comment exploiter les avancées récentes des techniques de preuve pour augmenter la puissance du calcul d'horloges. Et, surtout, il a expliqué comment une extension du calcul de causalité permet de traiter des programmes incluant des contraintes numériques, extension utile lorsque l'on s'intéresse à la modélisation de systèmes physiques.

Cours 4 – Boucles et réincarnation en Esterel

Cette séance a été dédiée à la production de circuits et codes logiciels efficaces à partir de programmes Esterel. Elle a d'abord traité le cas des boucles dans la traduction en circuits, qui pose le problème particulièrement délicat de la *réincarnation* des instructions, c'est-à-dire de la possibilité d'exécuter plusieurs fois successivement mais simultanément une même instruction au cours d'une même réaction. Cela ne pose pas de difficultés pour un interpréteur, mais, pour une compilation efficace, il faut recopier un nombre de fois suffisant une partie de ces instructions. Nous avons montré comment passer graduellement d'une méthode triviale mais exponentielle à des méthodes quadratiques puis quasi linéaires, adoptées dans les compilateurs industriels, en rappelant pourquoi la sémantique constructive conduit à des circuits corrects même s'ils comportent des cycles combinatoires (voir le cours du 26 mars 2014). Le cours a ensuite brièvement présenté une méthode radicalement différente de compilation plus efficace pour les cibles logicielles, initiée par Stephen Edwards à l'université de Columbia. Une fois la réincarnation traitée, et en tirant parti d'un encodage subtil des états et du fait que les tests logiciels n'évaluent qu'une seule branche alors que la propagation booléenne parcourt toujours les deux dans les circuits, cette méthode conduit à des performances meilleures que celles obtenues par la traduction de circuit, pouvant même devenir sublinéaire en la taille du programme. Une variante en a été implémentée dans le compilateur industriel Esterel v7 d'Esterel technologies dont j'ai dirigé le développement.

Séminaire 4 – *Sequential Constructiveness, SCL and ScCharts: incorporating synchrony in conventional languages*

Reinhard von Hanxleden (professeur à l'université de Kiel, Allemagne)

Engagé depuis longtemps avec son équipe dans l'enseignement et les applications d'Esterel, Reinhard von Hanxleden a récemment simplifié et étendu les principes d'Esterel en généralisant la notion de séquentialité constructive à un traitement plus général de variables partagées synchrones dans les langages classiques. Cette extension permet des changements multiples de leurs valeurs dans un même instant, chose interdite aux signaux d'Esterel, ainsi qu'un meilleur traitement des initialisations. Cette nouvelle vue sémantique n'est pas pertinente pour les circuits électroniques mais elle est naturelle pour les applications logicielles. Elle a conduit à la création du nouveau langage SCL fondé sur C et accompagné du langage graphique SCCharts ainsi que du langage SCEst qui étend Esterel avec la nouvelle sémantique. Tous ces points ont été illustrés par de nombreux exemples.

Cours 5 – *Clock-gating*, multi-horloges, implémentation et optimisation

Ce cours a d'abord étudié l'insertion de programmes Esterel (ou synchrones en général) dans des environnements d'exécution arbitraires en détaillant les différentes façons de recevoir ou d'émettre des événements par une machine d'exécution reliant le monde localement synchrone d'Esterel et le monde extérieur asynchrone. Il a détaillé la primitive « exec » d'appel et de gestion d'actions asynchrones, introduite initialement pour le contrôle de robots et déjà brièvement présentée lors du premier cours. Le cours a ensuite étudié les différentes façons de démontrer automatiquement des propriétés des programmes Esterel, en s'appuyant sur les techniques présentées en 2014-2016 pour la vérification de programmes : calculs directs sur automates explicites, méthodes implicites de vérification à l'aide de solveurs SAT/SMT, etc. Des exemples industriels ont illustré la complémentarité entre les mises au point par simulation et par preuve automatique, mettant en valeur la puissance de cette dernière approche.

Séminaire 5 – La vérification formelle d'un compilateur Lustre

Timothy Bourke (chercheur à l'Inria Paris)

Le séminaire a porté sur la formalisation et la preuve en Coq d'un générateur de code impératif pour un noyau du langage Lustre. Une telle vérification formelle pourrait avantageusement compléter les méthodes officielles de certification actuellement employées pour SCADE 6, qui intègre Lustre et une version restreinte d'Esterel. Même si le cœur du compilateur n'est qu'un petit morceau du système total, il est essentiel car tout bug engendré par ce cœur peut avoir des conséquences imprévisibles et potentiellement graves. L'exposé a présenté la spécification et la vérification d'un générateur de code simple qui gère les traits principaux de Lustre : l'échantillonnage, les nœuds et les délais. La chaîne de génération de code intègre le compilateur C formellement vérifié CompCert pour produire le code assembleur final. Les garanties offertes par CompCert permettent d'étendre le théorème de correction de la traduction de Lustre à toute la chaîne de compilation des programmes Lustre vers l'assembleur.

Cours 6 – HipHop.js, exec et vérification formelle

Ce sixième et dernier cours a brièvement décrit le langage HipHop.js, nouvelle version Javascript du langage HipHop déjà présenté avec Manuel Serrano (Inria Sophia-Antipolis) dans le cours et le séminaire du 28 mai 2013. HipHop.js, qui reprend en syntaxe JavaScript les constructions et le parallélisme synchrone d'Esterel, rend facile l'intégration de la programmation réactive synchrone à la programmation traditionnelle du Web en Hop.js. Par la possibilité de modifier au vol le code source HipHop.js à l'aide de fonctions Hop.js, il ajoute aussi un niveau supplémentaire de dynamicité aux programmes synchrones, ce qui est indispensable car le Web est en lui-même un environnement particulièrement dynamique. Mais le Web nécessite aussi un mélange fin des approches synchrones et asynchrones. Ceci est réalisé par l'intégration dans Hop qui comprend bien les événements du Web et permet aussi de lancer des réactions HipHop.js quand un composant d'interface homme-machine est activé ou quand la valeur d'une variable interne change. Enfin l'ajout de la primitive « exec » d'appel d'actions asynchrones introduite à la fin des

années 1980 dans le langage Esterel v5 permet de lancer et de contrôler finement des actions externes, comme le téléchargement depuis un site ou le mouvement d'un robot. Le premier compilateur prototype de HipHop.js vers Hop.js a été développé par Colin Vidal (thésard Inria), et son évolution fait partie de mes recherches.

Séminaire 6 – Vérification formelle en Coq de la chaîne des sémantiques pour la compilation d'Esterel

Lionel Rieg (post-doctorant à l'université de Yale)

L'exposé a d'abord présenté des preuves en Coq des théorèmes de correspondance reliant les différentes sémantiques de plus en plus précises d'Esterel, telles qu'énoncées dans le livre numérique *The Constructive Semantics of Pure Esterel* puis affinées dans la thèse d'Olivier Tardieu en 2004 : sémantique logique, sémantique constructive, sémantique par transitions entre états marqués dans les termes. Il a ensuite introduit une nouvelle sémantique opérationnelle réalisant la sémantique constructive par propagation dans le programme de jetons colorés, intuitivement proche de la traduction en circuits mais plus facile à traiter. La preuve de correction et de déterminisme de cette nouvelle sémantique est actuellement limitée aux programmes sans boucle : elle devra être étendue aux programmes généraux en incorporant un des traitements de la réincarnation présentés dans le cours 4 ci-dessus.

Colloque – Imagerie médicale et apprentissage : vers une intelligence artificielle ?

Ce colloque a été dédié aux algorithmes en médecine. Ses deux centres d'intérêt principaux ont été l'analyse automatique des images médicales et la collecte et l'analyse de grandes données, désormais permises par l'informatisation du suivi des patients. Les exposés ont rassemblé des chercheurs et industriels en imagerie et en analyse de données, des médecins discutant les apports médicaux de ces nouvelles méthodes, des acteurs des méthodes de collecte et d'exploitation de grandes données médicales, et des juristes concernés par les questions difficiles posées par l'ensemble de ces sujets. Un accent particulier a été porté sur les techniques d'apprentissage profond développées dans le domaine de l'intelligence artificielle. Le colloque s'est terminé par une analyse des transformations qui vont être apportées au métier de radiologue, suivie d'une discussion globale.

Enseignement à l'extérieur

Cours – La photographie numérique, un parfait exemple de la puissance de l'informatique

Inria Lille

L'appareil photographique numérique est un excellent exemple de l'évolution actuelle des systèmes cyber-physiques, c'est-à-dire des systèmes couplant intimement mécanique, physique, électronique et logiciel. C'est aussi un exemple merveilleux et accessible à tous de la puissance des méthodes de l'informatique par rapport à celles de la physique et de la mécanique seules. Le cours a présenté la

panoplie des nombreux algorithmes embarqués dans les appareils photographiques modernes ou les logiciels de post-production, de l'aide à la l'exposition et à la mise au point à la production de l'image finale par correction des distorsions optiques et fusion d'images multiples. Il a analysé en particulier les problèmes de correction automatique des défauts des images et les progrès de l'ergonomie des appareils photos modernes. L'exposé a insisté sur l'importance de la liaison entre physique et algorithmique : même si les algorithmes prennent de plus en plus le pas sur les procédés physiques, en particulier pour les appareils photographiques des téléphones, les capteurs et les systèmes de stabilisation « anti-bougé » des objectifs ou capteurs continuent à faire des progrès importants. De leur côté, les objectifs, bien sûr conçus algorithmiquement, s'allègent et maigrissent en taille.

Séminaire – Interaction homme-machine

Stéphane Huot (directeur de recherche à l'Inria Lille)

L'interaction homme-machine (IHM) a toujours été au cœur de certaines des visions qui ont contribué à forger l'informatique moderne. Mais, trop souvent considérée comme secondaire, elle reste souvent un point faible des systèmes informatisés actuels. Cette situation s'est récemment améliorée avec notamment l'avènement des dispositifs tactiles des smartphones et tablettes, pour lesquels la simplicité d'utilisation est devenue déterminante. Mais celle-ci a aussi conduit à appauvrir les possibilités d'usage avancé offertes par les technologies actuelles. Le difficile équilibre entre simplicité d'usage et puissance de l'outil est le défi majeur de l'IHM. Il requiert une approche centrée sur l'utilisateur, donc la compréhension des phénomènes sensorimoteurs et psychomoteurs, cognitifs, sociaux et technologiques mis en œuvre. Le séminaire a présenté les objectifs, méthodes et pratiques de la recherche en IHM, par l'étude des apports des pionniers du domaine et celle de ses visions modernes. Il s'est aussi appuyé sur des exemples et contre-exemples de qualité des IHM.

Recherche

Le cours 2016-2017 n'ayant pas eu lieu, je me suis concentré cette année-là sur deux points : l'écriture d'un livre destiné à un large public, et la poursuite du travail sur le langage HipHop, version d'Esterel destinée au Web. En 2017-2018, j'ai poursuivi la recherche sur HipHop et abordé deux nouveaux sujets : l'apprentissage automatique et les protocoles médicaux. En 2016-2018, j'ai aussi donné 53 conférences, dans des congrès scientifiques ou médicaux aussi bien que dans des organisations ou associations destinées au grand public. C'est pour moi une activité essentielle tant la connaissance de l'informatique moderne reste encore bien trop faible dans notre pays.

Sur le sujet HipHop, conduit avec Manuel Serrano de l'Inria Sophia-Antipolis, j'ai co-encadré la thèse de Colin Vidal, qu'il a soutenue à l'Inria le 6 juin 2018. Il y a proposé une nouvelle version textuelle de HipHop, qui remplace avantageusement la version XML précédente. Mais son principal apport a été l'écriture d'un compilateur prototype de HipHop vers JavaScript, que nous améliorons actuellement pour passer à l'échelle de nouvelles applications d'orchestration de services Web ou de contrôle d'objets connectés. Ce travail est fait dans le cadre du projet ANR Cisc.

Mon livre *L'Hyperpuissance de l'informatique* est paru en octobre 2017 chez Odile Jacob. Son objectif n'était pas d'initier le lecteur aux techniques de l'informatique et en particulier à la programmation des ordinateurs (au « code », comme on semble dire maintenant), mais de présenter la pensée informatique et la façon dont elle change le monde rapidement et profondément. Cet ouvrage est donc conçu dans le même esprit que ma première leçon inaugurale de janvier 2008, « Pourquoi et comment le monde devient numérique », mais avec neuf ans d'expérience supplémentaire. Après une présentation générale des concepts et façons de faire de l'informatique, six sujets sont discutés en profondeur, dans des domaines d'applications variés : l'évolution des télécommunications ; Internet avec ses moteurs de recherche et ses objets connectés ; les images, sons et cartes numériques avec un accent particulier sur la photographie numérique qui illustre parfaitement la conjonction moderne physique-informatique ; l'informatisation de la médecine, en particulier à travers l'imagerie médicale ; enfin, l'entrée en force de l'informatique dans les sciences. J'étudie ensuite de façon détaillée un problème-clé : comment rendre l'informatique plus sûre en conjurant ses deux dangers majeurs, à savoir les bugs et les trous de sécurité. Je termine par une vision personnelle de l'avenir de l'informatique, que j'amplifierai en 2018-2019 dans mon cours intitulé « Ou va l'informatique ? ». Pour les passionnés du sujet, des annexes analysent des points plus techniques sur les algorithmes, le parallélisme, les langages de programmation et la vérification formelle de programme ou de théorèmes mathématiques.

En septembre 2017, j'ai recruté un nouvel ATER, Alan Aboudib, après avoir fait partie de son jury de thèse à Brest. Son sujet est l'apprentissage automatique non supervisé ou semi-supervisé à base de neurones pouvant s'exciter mais aussi s'inhiber les uns les autres. Alan Aboudib n'a pas encore obtenu de résultats tangibles pour l'instant, mais son action se poursuit.

Enfin, lors d'une invitation à Northwestern University Chicago en mai 2018, j'ai entamé un nouveau sujet prometteur avec des professeurs et médecins de cette université et avec Manuel Serrano de l'Inria : l'utilisation d'Esterel et HipHop pour la rédaction précise et la traçabilité des protocoles médicaux, dont la conduite évolue avec le temps en fonction de diverses mesures. Ces protocoles complexes sont généralement spécifiés sous forme de listes de règles en langage naturel, ce qui les rend très difficiles à écrire, à lire et à suivre. Cela provoque de nombreuses erreurs de compréhension et d'exécution pouvant être graves (plusieurs dizaines de milliers par an dans les hôpitaux aux États-Unis). Nous pensons qu'Esterel ou HipHop sont idéaux pour écrire les protocoles médicaux de façon précise, les rendant beaucoup plus clairs, précis, et analysables, tout en améliorant leur traçabilité dynamique pour les différents acteurs concernés.

PUBLICATIONS

BERRY G., *L'Hyperpuissance de l'informatique : algorithmes, données, machines, réseaux,* Paris, Odile Jacob, 2017.

BERRY G., *The Informatics of Time and Events. Inaugural lecture delivered on Thursday 28 March 2013*, trad. L. LIBBRECHT, Paris, Collège de France, coll. « Leçons inaugurales », 2016, DOI : 10.4000/books.cdf.4120, http://books.openedition.org/cdf/4120.

BERRY G., « Vers une approche algorithmique des sciences du vivant », *in* T. GAUDIN, D. LACROIX, M.-C. MAUREL et J.-C. POMEROL (dir.), *Sciences de la vie, sciences de l'information. Actes du colloque qui s'est déroulé du 17 au 24 septembre 2016 au Centre*

culturel international de Cerisy, ISTE Éditions, coll. « Systèmes d'information, web et société », 2017, p. 163175.

BERRY G., « Un géant de création et de simplicité », *in* J.-M. LÉVY-LEBLOND (dir.), *Lettres à Alan Turing*, Vincennes, Éditions Thierry Marchaisse, 2016, p. 47-62.

BERRY G., *La numérisation du monde : Pourquoi ? Comment ? Quelles perspectives pour les mondes agricoles et alimentaires ?*, Angers, École supérieure d'agriculture, coll. « Les leçons inaugurales de l'ESA », 2017.

BERRY G. et DELAHAYE J.-P., « Jouer ou ne pas jouer au Loto, telle est la stratégie », *Interstices*, INRIA, 2017, https://hal.inria.fr/hal-01533685.

BERRY G., Préface de LAZARD E. et MOUNIER-KUHN P. *Histoire illustrée de l'informatique*, Les Ullis, EDP Sciences, 2016.

VILLANI C., BERRY G., BROUÉ M. *et al.*, *Comprendre les mathématiques. Les textes fondamentaux : Euclide, Descartes, Euler, Cauchy, Galois, Riemann, Turing, Nash, Grothendieck...,* Paris, Le Point Références, 2017.