ADRESSE PERSONNELLE : 23 RUE FÉLIX FAURE
54000 NANCY - FRANCE
TÉL. DOMICILE : 03 72 14 08 64

ADRESSE PROFESSIONNELLE : LORIA - BP239
615, RUE DU JARDIN BOTANIQUE
54506 VILLERS-LES-NANCY - FRANCE
TÉL. : 03 54 95 84 61

MEL BONFANTE@LORIA.FR

CORPS : MAÎTRE DE CONFÉRENCES
SECTION CNU : 27
COMPOSANTE : UNIVERSITÉ DE LORRAINE, MINES DE NANCY
LABORATOIRE DE RATTACHEMENT : LORIA - UMR 7503

# GUILLAUME BONFANTE

Né le 07 janvier 1971,
de nationalité française
vie maritale, 2 enfants

## FORMATION

École des Mines de Nancy. diplômé en 1995

DEA d'informatique de l'INPL. 1995

Doctorat d'informatique de l'INPL, *Constructions d'ordres et analyse de la complexité*. 2000.
*direction* : Adam Cichon, François Lamarche.
Jury : Peter Dybjer (rapporteur), Jean-Pierre Jouannaud (rapporteur, président), Jacques Jaray, Jean-Yves Marion

Habilitation à diriger des recherches de l'Université de Lorraine, *Complexité implicite des calculs : interprétations de programmes. 2011*

Direction : Jean-Yves Marion, Professeur, Université de Lorraine,

Président : Claude Kircher, Directeur de Recherches, INRIA

Rapporteurs :
- Roberto Amadio, Professeur, Université Paris Diderot (7),
- Etienne Grandjean, Professeur, Université de Caen
- Simona Ronchi della Rocha, Professeur, Université de Turin

Examinateurs :

- Daniel Leivant, Professeur, Université d'Indiana,
- Patrick Baillot, Chargé de recherches, ENS Lyon,
- Philippe de Groote, Directeur de recherches, INRIA.

## FONCTIONS ASSURÉES

Allocataire de recherches, MESR. 1996-1999

Moniteur à l'Université Nancy 2. 1997-1999

ATER à l'Université Henri Poincaré, Nancy 1. 1999-2000

ATER à l'Institut National Polytechnique de Lorraine. 2000-2001

Maître de conférences à l'Institut National Polytechnique de Lorraine. 2001-2006 et depuis 2008 (***PES 2011-2015***)

Chargé de recherches à INRIA. 2006-2008

## DOMAINES DE RECHERCHE

Virologie informatique

Complexité implicite des calculs

Traitement automatique de la langue

## ACTIVITÉS D'ENSEIGNEMENT

*Responsabilité de cours, actuellement*, c.à.d. depuis 2008 :

- **Programmation Java** (2ème semestre) (60 h/an). Il s'agit d'un cours d'apprentissage de la programmation pour les débutants (première année (1A=L3) École des Mines). Contenu : structures de données, notions d'algorithmique, aspects scientifiques de la discipline.
- **Informatique théorique** (30 h/an). Cours pour étudiants en 2A=M1 à l'École des Mines, spécialisés en informatique. Contenu : langages, automates, compilation, calculabilité, complexité, logique.
- **Modélisation du logiciel** (30 h/an). Cours pour étudiants en 2A=M1 à l'École des Mines, spécialisés en informatique. Contenu : cycle de vie du logiciel, planification du développement, outils de modélisation (UML).
- **Techniques d'animation et jeux video** (40 h/an). Cours d'ouverture pour tous les étudiants de 2A=M1 de l'École. Contenu : problématique du jeu vidéo, notion de rendu, modèle du monde, techniques de programmation (OpenCL, OgreSDK).
- **Sûreté et sécurité des systèmes informatiques** (30 h/an). Il s'agit d'un cours pour les étudiants en 3A=M2 à l'École des Mines, spécialisés en informatique. Contenu : systèmes formels (i.e. TLA+, COQ), méthodes de protection vis à vis du piratage, virologie informatique.

*Autres cours dispensés à l'École des Mines* (avant 2008) :

- **Logique** (Première année, 20h/an pendant 3 ans)
- **Algorithmique** (Deuxième année, 10h/an pendant 3 ans)
- **Bureautique** (Première année, 10 h/an pendant 4 ans)

- **Bases de données** (Deuxième année, 40h, une année)
- **Visual Basic/Excel** (Première année, FIMGP, 30h, une année)

*Cours dispensés* en dehors de l'Ecole des Mines :

**Programmation C**, L1-IUT Charlemagne, Université de Nancy 2, 1999

**Mathématiques pour l'informatique**, L3, Université Henri Poincaré (UHP), 2000
**Programmation Eiffel**, Première année, ESIAL, UHP, 2000

**Programmation CAML**, L3, UHP, 1999

**Utilisation d'Unix**, DEA imagerie numérique d'Epinal, UHP, 1999

**Algorithmique**, M2, Metz 2012

**Virologie informatique**, M2, Nancy 2013

*Projets scientifiques et techniques* à l'École des Mines :
- 1ère année, de quatre à six étudiants par an. Sur toute l'année, avec une semaine bloquée. Sujets : robot LEGO, musique électronique, synthèse d'images.
- 2ème année, 1 groupe de 2 étudiants par an. Sur toute l'année, un jour par semaine. Exemples de sujets : calcul d'interprétations polynomiales, machine virtuelle certifiée pour la programmation embarquée, programmation des expressions régulières en JAVA, construction de virus pour x86, recherche de vulnérabilité.

## ECOLE D'ÉTÉ

Prospects in Theoretical Physics, *Institute for Advanced Studies*, Princeton, juillet 2012. Voir la page web de l'évenement:

http://www.sns.ias.edu/pitp2/index.html

Cours sur la virologie informatique. J'ai établi une collaboration avec Benjamin Greenbaum (de l'IAS) suite à l'école d'été.

## DESCRIPTION DES COURS (ÉCOLE DES MINES)

**Programmation Java (semestre 2)**
Ce module  a pour but :
- de donner aux étudiants une autonomie en programmation ;
- d'élargir le spectre de l'informatique à d'autres aspects que la programmation : bases de données, réseaux, etc ;
- de montrer que l'informatique est (aussi) une discipline scientifique.

Ce cours fait suite à un premier cours de programmation donné au premier semestre. Il est à destination de l'ensemble de la promotion, un cinquième rentrant au département d'informatique. Le troisième point mentionné ci-dessus est un aspect que j'ai tout particulièrement développé. Force est de constater que l'informatique apparaît aux étudiants qui rentrent à l'École comme un simple outil technique un peu rébarbatif. Nous montrons qu'il y a également une dimension scientifique :  dans ce cours, nous l'illustrons en décrivant des problèmes ouverts de l'informatique dans de nombreux thèmes : informatique théorique, traitement de la langue, réseaux de neurones, réseau. Par exemple, pour l'informatique théorique, nous parlons des propriétés (comme la terminaison ou la complexité) de certains programmes, et des raisonnements afférents. Mention est faite des travaux de Cook (à propos de NP), d'Immermann (complexité descriptive), de Leivant (sur le Tiering).

Les supports de cours (transparents de cours et TD) sont en ligne sur la page de l'Ecole.

**Modélisation du logiciel (2A informatique)**
Ce module a pour but :
- de faire prendre conscience de l'importance de la modélisation pour la gestion du projet logiciel et la qualité du logiciel, de définir la notion de spécification, de cahier des charges ;
- de présenter le cycle de vie du logiciel, du processus de développement ;
- de présenter des langages pour la modélisation comme UML, et les outils afférents.

L'originalité de ce cours est son processus d'évaluation. Les étudiants travaillent par groupe de quatre. Nous proposons aux étudiants un projet de taille trop ambitieuse pour être réalisé. Par exemple : un système de gestion de salle, un portail d'achat en ligne automatisé, un réseau social, etc. Les étudiants doivent définir et réaliser deux cas d'utilisation. Ils sont notés sur l'adéquation entre ce qu'ils se sont engagés à faire, et le résultat.  Au sein de chaque groupe, chaque étudiant a un rôle défini, et il doit rendre compte individuellement de son apport dans le projet. Parce qu'un tel projet n'est pas faisable dans les temps imposés, les étudiants doivent prendre conscience de la difficulté des problèmes, et développer leur aptitude à se coordonner.

Le cours est réalisé au tableau.

**Fondement de l'informatique/sémantique (2A informatique)**
Ce module a pour but de présenter les outils et les notions fondamentales de l'informatique comme :
- les automates,
- les grammaires/langages,
- la sémantique (opérationnelle).
- la calculabilité,
- la complexité,

Les aspects théoriques de ce cours sont illustrés au cours d'un projet de compilation/interprétation de programmes. Le projet change tous les ans. De nombreux outils ont été employés comme java (expressions régulières), javacc, sablecc, antlr, etc. Nous avons compilé vers des langages comme postscript, x86, AVR

Le cours est réalisé au tableau.

**Techniques de l'animation et du jeu video (2A, non spécialistes)**
Ce module a pour but :
- de présenter les problèmes techniques de l'animation et du jeu video,
- d'aborder les aspects sociologiques du jeu vidéo,
- de présenter les outils de développement spécifiques au jeu vidéo.

Dans ce cours, nous faisons intervenir un(des) homme(s) du métier du jeu vidéo. Il aborde un certain nombre de problèmes qui apparaissent dans cette industrie. De ce fait, il s'agit d'un cours assez technique (une aisance en programmation est requise pour participer à ce cours) où nous programmons par exemple les cartes graphiques (OpenCL). En outre, nous utilisons le SDK Ogre, et DirectX, plus généralement la programmation graphique 3D sous Windows. L'aspect de modélisation est abordé au travers du logiciel Blender et/ou Sketchup où nous introduisons différents concepts : scène, forme, transformation géométrique, armature, mouvement.

Les transparents de cours sont accessibles en ligne

**Sécurité et sûreté (3A informatique)**
Ce module a pour but :
- de présenter les aspects de sûreté et de sécurité des systèmes,
- de faire prendre conscience de la profonde vulnérabilité des systèmes,
- de décrire des techniques de protection

Ce cours aborde les enjeux de la sécurité informatique et traite des concepts de base, puis aborde les problèmes de risques et la sûreté dans les systèmes d'information, et les moyens d'assurer la qualité de ces systèmes.

Pour la partie concernant la sûreté dont j'ai la charge, après un tableau général des méthodes employées (méthodes formelles à base de preuve, interprétations abstraites, model-checking), je présente en détail l'outils TLA+.

Le cours est réalisé au tableau

**Projets 1A**
Ce module a pour but :
- d'aborder les généralités de la gestion de projet, ses composants et son analyse ;
- d'initiation à la gestion de projet avec application réelle ;
- de définir les risques ainsi que la conduite à mener pour une réussite du projet.

L'aspect ludique et créatif plaît beaucoup aux étudiants. D'où les sujets : robot LEGO, musique électronique, synthèse d'images. C'est pour moi la possibilité de découvrir des étudiants doués, créatifs, susceptible de produire un travail scientifique plus conséquent.

**Projets 2A**
Ce module a pour but :
- de permettre aux étudiants d'envisager la complexité des systèmes d'information au travers d'un problème technique et scientifique de taille conséquente ;
- d'élargir les compétences techniques des étudiants ;
- de s'entraîner à la conduite de projet de longue durée.

Ce type de projet, un jour par semaine sur une année, permet un travail de plus grande envergure. À titre d'illustration, nous avons encadré la programmation du logiciel CROCUS, un logiciel de calcul d'interprétation polynomiale pour des programmes fonctionnels du premier ordre. Actuellement, j'encadre un groupe chargé de la recherche de vulnérabilité dans des binaires.

## DIX ANS À L'ÉCOLE DES MINES

Après avoir exercé au département d'informatique de l'IUT comme moniteur, et à celui de l'Université Henri Poincaré comme ATER, j'ai eu un poste de maître de conférences à l'Ecole des Mines de Nancy.

C'est un contexte un peu particulier. Premièrement, c'est une école d'ingénieurs généraliste. Certains étudiants s'orienteront vers les matériaux, d'autres vers le secteur de l'énergie ou la géologie. Enseigner l'informatique dans ces conditions est un exercice difficile : chaque spécialité emploie ses propres outils: mathematica, matlab, udec, fortran, etc. Les enseignements de première années donnent un socle de base de programmation, censé couvrir tous ces types de besoins.

Deuxièmement, c'est un milieu de tradition. Les cours ont un format précis, uniforme pour toutes les disciplines. Les emplois du temps, les modes d'évaluation sont contraints. Cela donne peu de degrés de liberté dans la forme.

Troisièmement, les étudiants n'ont pas de problème pour trouver un emploi. Cela leur donne un rapport au savoir différent de celui que j'ai pu observer dans les

filières standards de l'université. Les étudiants ont encore plus tendance à choisir ce qui les intéresse, mais alors, ils le font avec sérieux.

### PREMIÈRE ANNÉE

Pour les aspects pédagogiques, la première année est importante : c'est celle où nous sommes en «compétition» avec les autres spécialités. J'ai mis en oeuvre trois réformes : le développement de l'aspect scientifique de la discipline en cours, l'évaluation par des oraux, et les TDs en anglais. L'évaluation est réalisée en grande partie par un contrôle continu, la présence en TD étant obligatoire. Un contrôle terminal confirme la note donnée par les assistants de TD. Pour la très grande majorité des cas, les deux notes concordent. Cependant, le contrôle à l'oral nous a permis d'affiner la notation pour les cas problématiques, bien plus que le contrôle écrit que nous faisions avant.

Les jeunes élèves-ingénieurs devront faire un stage à l'étranger pour une durée minimale de six semaines. J'ai proposé qu'ils puissent bénéficier de TD en anglais. Cela fonctionne sur une base de volontariat, et nous constatons tous les ans plus de demandes que d'offres.

### DEUXIÈME ANNÉE

Le contenu du cours d'informatique théorique a été revu de fond en comble. Il touchait des sujets comme le lambda-calcul, la sémantique dénotationnelle, à mon avis sans pertinence compte tenu du manque de culture des étudiants. Nous illustrons maintenant la sémantique opérationnelle au travers de la compilation. Nous produisons du code pour de nombreuses architectures : postscript, x86 (via nasm), ou pour la puce AVR.

J'ai mis en route un cours sur les techniques du jeu vidéo, un cours qui a un grand succès. Je l'ai introduit parce qu'il 1) est pertinent pour l'industrie du jeu, 2) il aborde de nombreuses questions d'informatique : réseau, graphisme, intelligence artificielle, complexité des calculs, etc, 3) il permet de mettre en oeuvre l'algorithmique vue en première année : algorithmes A*, programmation dynamique, etc, 4) il permet de s'attaquer à la programmation à bas niveau (sur la carte graphique) au travers des shaders. En 14 séances de 3h, nous avons le temps de travailler en profondeur.

### EN TROISIÈME ANNÉE

J'ai construit le cours de sûreté et sécurité (en collaboration avec Jean-Yves Marion). Il n'y avait pas de tels cours, ce qui m'est apparu comme un manque flagrant pour une école d'ingénieurs. La notion de sûreté est une notion qui passe plutôt bien auprès des étudiants. Ce n'est pas le cas de la notion de sécurité. Nous faisons intervenir des personnes du monde industriel (mais aussi académique) pour faire prendre conscience des problèmes de sécurité. Dans ce cadre, le rôle de l'enseignant est de mettre tout en contexte et de faire le point entre les éléments anecdotiques présentés par les extérieurs et ceux qui sont plus fondamentaux.

### UNE PROPOSITION DE MASTER D'ECOLE (2012)

J'ai proposé (et rédigé la proposition) cette année le montage d'un Master d'Ecole (dit Duby) sur la sécurité des systèmes informatiques en partenariat avec deux autres écoles d'ingénieurs de Nancy : Télecom-Nancy et l'ENSEM. Le dossier contient : 1) le positionnement de la filière au sein de l'université, 2) sa justification dans le contexte économique, 3) un contenu pédagogique, 4) un budget (de 150k€/an), 5) les aspects administratifs et l'accord entre les trois écoles. Le dossier est en cours de traitement au ministère.

## ACTIVITÉS DE RECHERCHE

J'ai travaillé au sein des équipes suivantes :

LORIA - Équipe EUREKA, responsable Pierre LESCANNE. 1996-1999
Thèmes de recherche : lambda-calcul, réécriture, graphes et combinatoire.
Dans cette équipe, j'ai acquis des compétences autour de la réécriture de termes, de la terminaison et de la complexité. Les travaux réalisés concernent le calcul automatique de la complexité des programmes du premier ordre, plus particulièrement, le calcul par *interprétations polynomiales.*

LORIA - Équipe CALLIGRAMME, responsable Philippe DE GROOTE. 1999-2007
Thèmes de recherche : logique linéaire, traitement de la langue naturelle.
Dans cette équipe, j'ai acquis des compétences en traitement automatique des langues. Mes contributions ont porté essentiellement sur la *désambiguïsation lexicale*, et sur l'analyse grammaticale des grammaires d'interaction.

LORIA - Équipe CARTE, responsable Jean-Yves MARION. 2007-
Thèmes de recherche : virologie informatique, complexité implicite des calculs.
Dans cette équipe, j'ai participé à la création d'une activité de recherche en sécurité informatique, et plus particulièrement en virologie informatique. Mes contributions portent essentiellement sur la reconnaissance de programmes malveillants par signature de graphe, et l'étude théorique de l'*auto-référence* et de l'*auto-modification* de programmes.

### EN BREF

Mes contributions
- en virologie informatique :
    - Liens entre théorèmes de récursion et la virologie
    - Conception d'un détecteur de virus fondé sur la sémantique des programmes
- en complexité implicite des calculs
    - la notion de quasi-interprétation, une mesure de complexité calculable
    - les polygraphes comme modèle de calcul
    - la caractérisation des classes (NC_k)_k de complexité parallèle
- en traitement automatique des langues
    - algorithmes de désambiguisation exacts
    - emploi de la réécriture de graphe en TAL

### ET MAINTENANT

Mes recherches actuelles  portent en priorité sur la virologie informatique. La complexité implicite des calculs est un point secondaire, mais comme il est encore très fructueux, je conserve cette activité de recherche. Enfin, le traitement automatique des langues (TAL) est en quelque sorte un violon d'Ingres, quoique considéré avec sérieux.

Voici les travaux que je compte développer dans les temps qui viennent. Dans la contribution technique de ma thèse d'habilitation, je parle essentiellement de mes recherches en complexité. Ces travaux sont les mieux établis, et mes contributions témoignent (je l'espère) de mon expertise dans le domaine. On y trouvera d'ailleurs, de manière détaillée, les perspectives de recherches que j'entrevois en complexité implicite des calculs.  Ici, je présente les perspectives en virologie informatique.

## VIROLOGIE INFORMATIQUE

Le laboratoire de haute sécurité a été inauguré le 1er Juillet 2010 à Nancy. C'est une plateforme de recherche unique en France qui permet de conduire des expériences inédites en sécurité informatique, dans un cadre réglementaire. C'est donc une opportunité formidable pour qui veut travailler en virologie. S'il est bien difficile de définir formellement ce qu'est un virus, les problèmes de protection contre ce genre de plaies restent très concrets. Ce qui me plaît dans ce domaine de recherches, c'est que des questions anciennes de calculabilité, de complexité, de logique, de transformation de programme, de réécriture peuvent être relues/ renouvelées dans le cadre de la virologie informatique. Il en ressort un domaine de recherche avec les deux pendants, théoriques et pratiques. Pour illustrer ce propos, dans notre article de CIE 2007, "A Classification of computer viruses through recursion theorems", nous montrons que les théorèmes de récursion sont des "usines" à virus, en outre, plus le théorème est général (récursion paramétrée, récursion double, etc.), plus les virus correspondants disposent de facultés supplémentaires (comme le polymorphisme, l'auto-propagation, etc.). Certaines constructions laissent d'ailleurs envisager des virus encore non observés (avec polymorphisme de la propagation par exemple).

### LA DÉTECTION PAR SIGNATURE

Issu de travaux de recherches théoriques menées au cours de la thèse de Matthieu Kaczmarek, nous avons développé un prototype de moteur de détection de malwares qui emploie des techniques inédites de reconnaissance de fichiers infectés. Si on observe le bon fonctionnement de ce type de détection dans certains cas (code non auto-modifiants, avec graphe de flot de taille suffisante), le principe reste mal compris. Dans le cadre de la compétition internationale dans ce domaine, il est urgent d'en faire une étude plus précise, étude qui permettra une maîtrise de l'outil de détection. Par exemple, la notion de graphe décrivant un programme doit être précisée. En effet, nous employons plusieurs sortes de graphes (graphes de flot de contrôle, graphes de flot de données, graphes de vague, etc.) tous liés à leur programme. Il me semble indispensable de dégager les principes uniformes que nous employons, et de coordonner les différentes vues d'un malware.  En outre, pour rendre la détection plus robuste, il est crucial de prendre en compte les packers, ces programmes d'obfuscation de programmes, avec leurs procédés afférents (chiffrage, auto-modification, etc.). Cela pose de nombreux problèmes que nous évoquons ci-dessous.

### LA DÉTECTION DE PACKER ET OBFUSCATEUR

Un des soucis de la virologie concerne la rétro-ingénierie. Les auteurs de virus, ou plutôt les auteurs de compilateurs de virus (des programmes qui permettent de

synthétiser des virus), ont tout intérêt à cacher le code de leur virus (pour se prémunir contre les anti-virus, mais aussi pour se protéger contre la copie de leur savoir-faire). Ils font appel à de nombreuses techniques d'obfuscation. Le problème général de la désobfuscation est indécidable d'après Barack. Des réponses partielles sont toutefois possibles. Pour aider le travail d'analyse de code de virus, le premier souci consiste grosso modo à "nettoyer" le graphe de flot de contrôle/données tel qu'il est observé en cours d'exécution. Ici encore, on a affaire à la notion de transformation de programmes à sémantique équivalente. L'emploi des interprétations abstraites pour la détection de prédicats d'obfuscation (cf. dalla Preda et al.) est une autre voie de recherches prometteuses. C'est par exemple ce qu'on trouve chez Bardin et al. au CEA. Sur ce sujet, je participe à l'ANR Binsec, justement sous la responsabilité de Sébastien Bardin.

### LA RECHERCHE DE VULNÉRABILITÉ

Un autre élément d'analyse de programme concerne la recherche de vulnérabilités. Il s'agit de faire mal fonctionner un programme afin d'en détourner l'exécution. C'est un procédé standard de piratage. L'analyse statique basée sur la notion d'interprétation abstraite (cf. Cousot et al), par sa capacité à réaliser des exécutions symboliques, peut/devrait être utilisée pour chercher des vulnérabilités. Une manière de trouver de telles failles consiste à exécuter le programme avec toutes sortes de données et d'extraire celles qui peuvent être exploitées. Pour éviter la recherche au hasard, on peut utiliser l'analyse statique pour couvrir le plus parfaitement possible le graphe de flot du programme. Notons que ce problème de la couverture du graphe de flot est également un problème que nous avons identifié pour la détection par signature des programmes. Cette branche de recherche intéresse des entreprises comme "Safe river", avec qui nous envisageons une collaboration.

### POLITIQUE DE SÉCURITÉ

Plutôt que de tenter de reconnaître les programmes malveillants, et de les empêcher de s'exécuter, il est envisageable de définir une politique de sécurité a priori. C'est ce type de démarche qui aboutit à la notion de droit sur les fichiers par exemple. Ce type de compartimentage se retrouve dans la politique de sécurité de l'Android de Google : les données sont relatives à une application. En outre, chaque application doit déclarer les services qu'elle compte employer au cours de son exécution. Ce type de contrainte évite un emploi trop aisé de bibliothèques par les auteurs de malware.  Citons également l'exécution dans une sandbox, c'est-à-dire dans un environnement maîtrisé. Le problème, c'est qu'il est possible pour le programme exécuté d'observer son environnement d'exécution. L'idée de compartimentage mériterait une plus grande attention. Je pense employer à ce sujet le concept de "region based inference" comme on peut le trouver chez Talpin et al.

### ÉTUDE THÉORIQUE DES VIRUS

Si les notions de virus ou de malware ne sont pas définies formellement, c'est qu'elles font référence à l'aspect malicieux/destructeur de l'auteur du virus. Toutefois, le virus, le malware doit réagir avec son environnement. Par exemple, il doit se dupliquer, se transmettre, se transformer, exécuter sa "charge", etc. Pour

toutes ces tâches, il emploie des ressources de son environnement. De ce fait, pour une description théorique des virus, il me semble que l'approche visant à le décrire au travers d'interactions de processus est une bonne entrée en matière. Notons dans cette voie les travaux de master de Than Dinh Ta au LHS, et de Filiol et al. Le premier emploie le pi-calcul pour modéliser les interactions entre les processus. On peut envisager un calcul contraint (en termes de typage par exemple) pour lequel on aiderait le programmeur à éviter des effets imprévus. Je fais ici le parallèle avec la notion de non-interférence, et en particulier aux travaux de Boudol.

## COMPLEXITÉ IMPLICITE DES CALCULS

Mes travaux ont été développés dans le cadre de l'ARC Criss et de l'ANR Complice.

### EXTENSION DU DOMAINE DES PROGRAMMES

Les deux principes moteurs que nous avons employés jusque là, l'étude des preuves de terminaison en réécriture, et l'étude des schémas de récursion, sont toujours actifs : de nouvelles méthodes de preuves de terminaison sont proposées en permanence (voir le concours annuel d'outils de preuve de terminaison organisé par l'Université d'Innsbruck2). Chacune doit être considérée pour la complexité.

Sinon, nous pouvons considérer d'autres paramètres. Nous avons concentré notre analyse sur les programmes fonctionnels du premier ordre. Il est temps de s'attaquer à l'ordre supérieur. Ce qui m'intéresse le plus, c'est le croisement de l'approche logique actuelle et de la méthode des interprétations de programme.

### L'APPLICATION DES INTERPRÉTATIONS, TOUT UN PROGRAMME

On pourrait se dire que les aspects pratiques de l'application de ce type de théorie ne sont qu'une affaire de bonne ingénierie. Il me semble toutefois que certaines questions restent d'ordre scientifique, et j'y vois deux raisons. Premièrement, le calcul des interprétations reste un problème ouvert. Le calcul des coefficients (réels) des polynômes (à variables réelles) est a priori exponentiel relativement à la taille du programme, et doublement exponentiel relativement au degré des polynômes. Deuxièmement, le programme soumis à l'analyse de complexité n'est pas écrit en langage machine, mais dans un langage de haut niveau. Il faut donc assurer l'adéquation entre l'analyse réalisée sur le programme et le coût "réel" de l'exécution du programme. En d'autres termes, il faut assurer une compilation qui reste honnête relativement à notre analyse.

Je propose de passer par l'ordre supérieur pour synthétiser des interprétations au premier ordre.

### UNE THÉORIE ALGORITHMIQUE DE LA RÉÉCRITURE

Vue comme un modèle de calcul, la réécriture est Turing-complète, et comme chaque étape de calcul peut être simulée en temps constant sur une machine de Turing (cf. Martini et dal Lago), la longueur de dérivation est en fait une bonne mesure de complexité. Le souci vient du fait qu'un modèle de calcul n'a pas forcément une notion intrinsèque de coût, du moins une notion compatible avec les autres modèles de calculs. Martini et dal Lago proposent d'appeler ce type de recherche, la complexité implicite des calculs du point de vue micro ("in the small").

Il me semble que l'on devrait faire le pont avec les "Abstract State Machines" de Gurevich, pour lesquels il y a une axiomatisation précise des capacités d'un modèle de calcul "raisonnable". Le problème peut se poser pour des modèles de calculs qui ne sont pas Turing-complets.

L'autre souci vient de la profusion des caractérisations de classes de complexité. Rien qu'en se restreignant à la réécriture, on peut trouver une dizaine de critères pour décrire le temps polynomial. Elles n'ont pas toute la même intentionalité. Habituellement, pour comparer sa théorie avec les autres, chaque auteur met en avant un exemple particulièrement criant, sensé illustrer l'intentionalité de la théorie. Pour aller plus loin, comme la comparaison d'ensembles de programmes (syntaxes différentes, sémantiques différentes) est encore plus difficile que la comparaison d'ensembles de fonctions, nous proposons de les étudier de manière indirecte. L'idée est de transformer deux ensembles de programmes de façon à les distinguer extensionnellement. Nous avons montré ainsi à la conférence DICE 2010 comment l'ajout de la non-confluence permet de séparer des classes de programmes. Ces travaux doivent être formalisés, et approfondis

## TRAITEMENT DE LA LANGUE NATURELLE

Le TAL est pour moi un domaine d'application intéressant de mon savoir-faire en complexité. J'emploie des techniques fondées sur les automates, la programmation dynamique, la réécriture.

Evidemment, je ne suis pas un expert du domaine. Toutefois, en concentrant mon travail sur un domaine bien précis—celui des grammaires d'interactions développées au laboratoire, je peux mettre en oeuvre un travail sérieux de recherches.

Je me suis d'abord intéressé à la désambiguisation lexicale (l'attribution d'une catégorie grammaticale (déterminant, nom, etc) à chaque mot d'une phrase). J'ai fait le choix de ne considérer que des méthodes exactes (donc, en particulier, je n'utilise pas de méthodes statistiques, au demeurant intéressantes). Pour le moment, je n'ai envisagé que des analyses globales de la phrase. Je suis maintenant en mesure d'avoir une approche locale de la désambiguisation. A mon avis, il y a moyen non seulement d'améliorer l'efficacité de la désambiguisation, mais aussi de rendre plus robuste (pour la langue parlée par exemple) ce type d'analyse. Enfin, de tels travaux conduisent à une approche où l'on mêle désambiguisation et analyse grammaticale en même temps.

Ensuite, je me suis intéressé à la transformation d'une analyse grammaticale en une analyse sémantique. Avec Mathieu Morey, un étudiant en thèse, nous avons proposé de travailler avec de la réécriture de graphes. Ces travaux autours de l'outil GREW sont de plus en plus visibles dans la communauté du TAL. De mon point de vue, ce qui reste intéressant sur ce sujet, c'est la difficulté de maîtriser le calcul. Les questions sont très différentes des questions usuelles en réécriture. Par exemple, les calculs sont non déterministes (une phrase, même avec une analyse grammaticale particulière, peut avoir plusieurs significations). Dans ces conditions,

établir un moyen de calcul efficace est à mon avis un défi de grande ampleur. Autre question, autre application : la réversibilité des règles peut être employée en génération, c'est-à-dire pour la reconstruction d'une phrase à partir de sa sémantique.

## RESPONSABILITÉS ADMINISTRATIVES

- Vice-président de la commission de spécialiste section 27 de l'INPL 2004-2006
- Membre élu du Conseil Scientifique de l'INPL. 2003-2006
- Membre nommé de la commission de recrutement des personnels ingénieurs du LORIA (2004-2009)
- Représentant du département informatique à la commission des relations internationales de l'École des Mines (2008-2012)
- Comité de sélection pour un poste de maître de conférences à Paris 13 (2011)
- Membre élu du Conseil d'Administration de l'Ecole des Mines (2012-2016)
- Coordinateur et rédacteur d'une proposition de Master Duby sur la sécurité des systèmes informatique pour l'Ecole des Mines de Nancy (2012)
- Responsable de la *semaine départementale* pour la filière informatique. Deux voyages au Japon de 1 semaine avec 25 étudiants (2009 et 2012) avec recherche des financements (institutions et privés) de 45k€.

## ANIMATION DE LA COMMUNAUTÉ SCIENTIFIQUE

- Membre du comité de programme des Workshops
    - Theory of Computer Viruses  2006, 2007
    - Proof Computation Complexity 2009,
    - Sécurité des Architectures Réseaux et Systèmes d'Information 2009
    - European Institute in Computer Antivirus Research 2011
    - Logic and Computational Complexity 2011
    - Developments in Implicit Complexity 2012
    - Botnet fighting conference, 2013
    - Termgraph 2014

- Membre du comité de programme de la conférence
    - Conference On Malicious And Unwanted Software 2012, 2013

- Membre du comité éditorial du "Journal of Computer Virology"

- Relecteur pour les journaux et conférences :
    - TCS, JCV, CSL, RTA, TLCA, ICALP, STACS, MFCS, etc.

## COLLABORATIONS NATIONALES, INTERNATIONALES

- Chercheur invité (par I. Oitavem), Centro de Matématica e Aplicações Fundamentais. Lisbonne. Mois de Juin et Juillet 2007, 1 semaine en avril 2012, 1 semaine en février 2013
- Invitation par Stanislas Leibler de l'Institute of Advanced Studies, Princeton, juillet 2012. Dans le cadre de l'Ecole d'été PiTP.

- Professeur invité (par E. Mexia), Universidade Nova de Lisboa - Lisbonne. Octobre 2008
- Invitations par des chercheurs :
    - Georg Moser, Université d'Innsbruck, 1 semaine en Mai 2007
    - Florian Deloup, Université Paul Sabatier, département de mathématique : 1 semaine en Février 2009, 1 semaine en Janvier 2013
    - Nao Hirokawa, JAIST (Kanazawa, Japon), 1 semaine en Février 2011

## EXPOSÉS GRAND PUBLIC

- Journées francophones de l'investigation numérique (JFIN), Levallois-Perret, 2012. Titre de l'exposé : «Les virus, amis ou ennemis de l'investigation numérique ? »
- Journées francophones de l'investigation numérique (JFIN), Neuchatel, 2013. Titre de l'exposé : « Analyse de codes binaires malveillants »
- Rendez-Vous du Numérique et de l'Intelligence Economique, Strasbourg - 2012. « Dans les films Terminator, Matrix,... Le système informatique prend le contrôle des réseaux informatiques par le biais de virus ... Fiction ou réalité ? »

## EXPOSÉS SUR INVITATION

- Séminaire de l'institut d'informatique de l'Université technique d'Innsbruck, Mai 2007
- Séminaire Séminaire de logique et mathématique du CMAF, Lisbonne. Juillet 2007, Avril 2012
- Séminaire de logique et modèle de calcul, Université de Coimbra. Juin 2007.
- Séminaire de l'institut de mathématique de l'Université Paul Sabatier. Toulouse. Mars 2009, Janvier 2013
- Séminaire "Rewriting and Logic Seminar", JAIST, Mars 2009, Avril 2012
- Séminaire de l'équipe associée Porgy, INRIA Bordeaux, Juin 2010
- Workshop on Proof Theory and Rewriting 2010, Obergurgl, Autriche, Mars 2010
- Workshop on Logic and Computation, Kanazawa - Japon. Février 2011
- Workshop on Proof Theory and Rewriting, Kanazawa, Japon, Mars 2013.
- Workshop on Implicit Computational Complexity and applications: Resource control, security, real-number computation, Shonan Village, Japon, Octobre 2013

## EXPERTISE

Expertise scientifique pour l'ambassade de France au Vietnam (à destination d'A. Rinckenbach), évaluation des université actives dans le domaine de l'informatique théorique. Décembre 2007. Rapport disponible sur le site d'INRIA

Netherlands Organisation for Scientific Research, Mars 2012, évaluation d'un projet de recherche pour un post-doc de 3 ans.

Netherlands Organisation for Scientific Research, Décembre 2012, évaluation d'un projet de recherche pour un post-doc de 3 ans.

## RECHERCHE CONTRACTUELLE

- Responsable local du Projet ANR COMPLICE (Implicit Computational Complexity, Concurrency and Extraction) (2008-2012 ) 64k€ (pour le site). Dans ce projet, nous développons des outils théoriques et pratiques d'analyse de la complexité implicite de programmes. Pour le site de Nancy, le logiciel CROCUS témoigne du savoir faire développé dans l'équipe.
- Responsable du Projet Emergent INPL "Exploration des procédures de détection de virus informatiques par graphe de flot" (2009-2010). 12 k€. Dans ce projet, nous développons un détecteur de virus par analyse morphologique.
- Responsable de l'opération VVV (Vie des Virus et des Vers) dans le cadre du CPER Intelligence Logicielle- Qualité et Sûreté des Logiciels. (2005-2006), 16k€. Dans ce projet, nous avons initié l'étude de la détection par analyse morphologique.
- Participation à l'ANR Binsec (resp. Sébastien Bardin, CEA) 2013-2016. BINary code analysis for SECurity. Nous apportons notre expertise en virologie.
- Participation à l'Action de Recherches en Amont (aujourd'hui ANR) : Virus (resp. J.-Y. Marion) (2006 - 2009). Nous avons étudié la notion de virus informatique, ses impacts sur la société, et nous avons envisagé des mesures de protection, ce qui sera par la suite la détection morphologique.
- Participation au projet CRISS (Contrôle de Ressources et d'Inférence pour Systèmes Synchrones) de l'ACI Sécurité Informatique (2004 - 2006). Nous avons proposé de nouveaux critères de complexité implicite des calculs.
- Participation au projet GEOCAL (Géométrie du Calcul) de l'ACI Nouvelles Interfaces des Mathématiques (2004 - 2006). C'est dans ce projet que nous avons fait la jonction entre les polygraphes et la complexité implicite des calculs.
- Participation à l'opération ACT (Analyse de la Complexité et Transformation) du CPER QSL (Qualité et Sûreté du Logiciel) du Pôle de recherche Intelligence Logicielle.  Nous avons proposé une première ébauche du logiciel CROCUS.

## VALORISATION DE LA RECHERCHE, DÉVELOPPEMENT LOGICIEL

Lauréat du **concours national de l'innovation OSEO** en 2009 pour un "logiciel détecteur de programmes malveillants". Dotation de 45k€ pour une étude de marché, et la mise en oeuvre d'un brevet. Avec Jean-Yves Marion, Matthieu Kaczmarek.

**MMDEC** : logiciel de détection de virus.
Dépôt APP IDDN.FR.001.300033.000.R.P.2009.000.10000

Ce logiciel permet de construire une base de données de signatures de virus, de reconnaître un virus à l'intérieur d'un programme, de classifier un programme relativement à une base de donnée de signatures. C'est le coeur du projet OSEO.
Volume : 20 000 lignes de C
Avec Jean-Yves Marion, Matthieu Kaczmarek, Fabrice Sabatier, Aurélien Thierry

**CROCUS** : calculateur d'interprétation de programmes.
Ce logiciel calcule une/des interprétations de programmes fonctionnels du premier ordre. L'interprétation peut être employée pour évaluer la complexité du programme.
Volume : 5000 lignes de JAVA

**LEOPAR** : analyseur de grammaires d'interaction.
Dépôt APP IDDN.FR.001.380002.000.R.P.2003.000.10000
Volume : 25 000 lignes de OCAML
Avec Bruno Guillaume, Guy Perrier, Sylvain Pogodalla, et al.

**GREW** : réécriture de graphes pour le traitement de la langue.
Volume : 3000 lignes de OCAML
Avec Bruno Guillaume, Matthieu Morey, et al.

## ENCADREMENT

**Thèse en co-encadrement avec Jean-Yves Marion :**

- Matthieu Kaczmarek, 2005 - 2008. "Des fondements de la virologie informatique, vers une immunologie formelle". Matthieu a reçu le second prix de thèse de la région Lorraine.
- Stéphane Wloka, 2010-2011. "Étude du flot de contrôle d'un programme pour la détection par analyse morphologique des codes malveillants". Thèse interrompue d'un commun accord.
- Thanh Dinh Ta, 2013 -. Détection de malware par analyse de messages.

**Thèse en co-encadrement avec Jean-Yves Marion :**

- Paul Bakouche, 2013 - . Théorie des noeuds et topologie en petite dimension

Collaborations avec des étudiants en thèse (*avec publications*) :

- Jean-Yves Moyen (1 journal, 2 conférences internationales),
- Romain Péchoux (2 conférences internationales),
- Joseph le Roux (2 conférences internationales),
- Daniel Reynaud (1 conférence internationale),
- Mathieu Morey (2 conférences internationales)
- Aurélien Thierry (1 conférence internationale)

**DEA/Master 2A :**

- Matthieu Kaczmarek, 2005
- Éric Therond, 2010.

Projets scientifiques de troisième année à l'École des Mines de Nancy

- 1 par an

Projets de deuxième année à l'École des Mines de Nancy
- 1 groupe de 4 tous les ans

**Jury :**
- thèse : Romain Péchoux (2008), Matthieu Kaczmarek (2009), Mahieu Morey (2011)
- rapporteur de la thèse : Jean-Marie Borello (direction : Ludovic Mé), 2011
- de stage de fin d'étude de l'École des Mines : tous les ans

## PUBLICATIONS

**Complexité implicite des calculs**

Revues internationales

1. Guillaume Bonfante, Adam Cichon, Jean-Yves Marion et Hélène Touzet. Algorithms with polynomial interpretation termination proof. Journal of Functional Programming 41 (0), 2001.
2. Guillaume Bonfante, Yves Guiraud. Programs as polygraphs: computability and complexity, Logical Methods in Computer Science 5(2), 2009.
3. Guillaume Bonfante, Jean-Yves Marion, Jean-Yves Moyen. Quasi-interpretations: a way to control resources. Theoretical Computer Science. 2011
4. Guillaume Bonfante, Bruno Guillaume. Non-size increasing Graph Rewriting for Natural Language Processing. Accepted for publication in Mathematical Structures in Computer Science.

Actes de congrès internationaux à comité de lecture

1. Guillaume Bonfante, Adam Cichon, Jean-Yves Marion et Hélène Touzet. Complexity classes and rewrite systems with polynomial interpretation CSL'98, LNCS 1584. 1998
2. Guillaume Bonfante, Jean-Yves Marion et Jean-Yves Moyen. On lexicographic termination ordering with space bound certifications. PSI ‘01, LNCS 2244. 2001
3. Guillaume Bonfante, Jean-Yves Marion, Jean-Yves Moyen. Quasi-interpretations and small space bounds RTA'05, LNCS 3467. 2005
4. Guillaume Bonfante. Some programming languages for Logspace and Ptime. AMAST‘06, LNCS 4019. 2006
5. Guillaume Bonfante, Jean-Yves Marion, Romain Péchoux, Characterization of ALOGTIME by first order functional programs. LPAR ‘06, LNCS 4246. 2006
6. Guillaume Bonfante, Reinhard Kahle, Jean-Yves Marion et Isabel Oitavem. Towards an implicit characterization of $NC^k$, CSL ‘06. LNCS 4207. 2006
7. Guillaume Bonfante, Jean-Yves Marion and Romain Péchoux. Quasi-interpretations synthesis by decomposition. ICTAC 2007, LNCS 4711. 2007
8. Guillaume Bonfante, Reinhard Kahle, Jean-Yves Marion, Isabel Oitavem, Recursion Schemata for NCk. CSL 2008, LNCS 5213. 2008
9. Guillaume Bonfante, Yves Guiraud: Intensional Properties of Polygraphs. Termgraph ’08. ENTCS. 203(1). 2008
10. Guillaume Bonfante, Non confluence in implicit complexity characterizations. DICE 2010. EPTCS 23. 2010
11. Guillaume Bonfante, Florian Deloup: Complexity Invariance of Real Interpretations. TAMC 2010: LNCS 6108. 2010
12. Guillaume Bonfante, Georg Moser: Characterising Space Complexity Classes via Knuth-Bendix Orders. LPAR 2010, LNCS 6397. 2010
13. Guilaume Bonfante, Course of value distinguishes the intentionality of programming languages. SOICT 2011, IEEE

**Virologie informatique**

Revues internationales

1. Guillaume Bonfante, Matthieu Kaczmarek et Jean-Yves Marion, On abstract computer virology from a recursion-theoretic perspective, Journal in Computer Virology, 2006 (2).
2. Guillaume Bonfante, Matthieu Kaczmarek and Jean-Yves Marion, Architecture of a malware morphological detector, Journal in Computer Virology (5), 2009.

Actes de congrès internationaux à comité de lecture
1. Guillaume Bonfante, Matthieu Kaczmarek et Jean-Yves Marion, Toward an abstract computer virology, ICTAC ‘05, LNCS 3722. 2005
2. Guillaume Bonfante, Matthieu Kaczmarek, and Jean-Yves Marion. A classification of viruses through recursion theorems. CiE 2007, LNCS 4497. 2007
3. Guillaume Bonfante, Matthieu Kaczmarek et Jean-Yves Marion, Morphological Detection of Malware, International Conference on Malicious and Unwanted Software 2008, IEEE.
4. Guillaume Bonfante, Jean-Yves Marion, Daniel Reynaud-Plantey : A Computability Perspective on Self-Modifying Programs. SEFM 2009. IEEE.
5. Guillaume Bonfante, Jean-Yves Marion, Fabrice Sabatier, Aurélien Thierry : Code synchronization by morphological analysis, Malware 2012, IEEE
6. Guillaume Bonfante, Jean-Yves Marion, Fabrice Sabatier, Aurélien Thierry : Analysis and diversion of Duqu's driver, Malware 2013, IEEE

**Traitement automatique des langues naturelles**
Revue en langue française
1. Guillaume Bonfante, Bruno Guillaume et Guy Perrier, Analyse syntaxique électrostatique, Traitement Automatique des Langues(44). 2003

Actes de congrès internationaux à comité de lecture
1. Guillaume Bonfante et Philippe de Groote. Stochastic Lambek categorial grammars FGMOL, 2001, ENTCS 53.
2. Guillaume Bonfante, Bruno Guillaume et Guy Perrier, Polarization and abstraction of grammatical formalisms as methods for lexical disambiguation, COLING'04.
3. Guillaume Bonfante, Joseph Le Roux, Guy Perrier: Lexical Disambiguation with Polarities and Automata. CIAA 2006. LNCS 4094.
4. Guillaume Bonfante, Joseph Le Roux: Intersection Optimization is NP-Complete. FSMNLP 2007.
5. Guillaume Bonfante, Bruno Guillaume, Mathieu Morey: Dependency Constraints for Lexical Disambiguation. IWPT 2009.
6. Guillaume Bonfante, Bruno Guillaume, Mathieu Morey et Guy Perrier : Modular Graph Rewriting to Compute Semantics. IWCS 2011.

Conférence en langue française
1. Guillaume Bonfante, Bruno Guillaume, Mathieu Morey et Guy Perrier. Réécriture de graphes de dépendances pour l'interface syntaxe-sémantique. TALN 2010.

2. Guillaume Bonfante, Bruno Guillaume, Mathieu Morey and Guy Perrier. Enrichissement de structures en dépendances par réécriture de graphes. TALN 2011
3. Bruno Guillaume, Guillame Bonfante, Paul Masson, Mathieu Morey and Guy Perrier. Grew : un outil de réécriture de graphes pour le TAL. TALN 2012

**Edition d'actes de conférences**

1. Guillaume Bonfante et Jean-Yves Marion: Theory of Computer Viruses. Journal in Computer Virology 3(1): (2007)
2. Guillaume Bonfante and Ugo dal Lago: DICE 2012. special issue of Theoretical Computer Science, à paraître.