

1 Curriculum Vitae

Hugo METIVIER (25 ans)
Nationalité Française
Célibataire

Situation actuelle : Doctorant 3^{ème} année au sein du projet Espresso, à l'IRISA (Rennes)

Cursus

- 2005 - 2008 **Doctorant en Informatique** à l'IRISA (Institut de Recherche en Informatique et Systèmes Aléatoires, Rennes, Campus de Beaulieu)
au sein du Projet Espresso (Environnement de Spécification de Programmes Réactifs Synchrones),
Financement : Allocation de recherche
- 2004 - 2005 **Master 2 Recherche Informatique** à l'IFSIC (Institut de Formation Supérieure en Informatique et Communication, Rennes, Campus de Beaulieu), filière Méthodes formelles
– mention assez bien
- 2003 - 2004 **Maîtrise d'Informatique** à l'IFSIC – mention assez bien
- 2002 - 2003 **Licence d'Informatique** à l'IFSIC – mention bien
- 2000 - 2002 **DEUG MIAS** (Mathématiques, Informatique et Applications aux Sciences) à l'Université de Nantes
- Juin 2000 **Baccalauréat Scientifique** au Lycée Guist'hau à Nantes

Enseignement

- 2005 - 2008 **Moniteur en Informatique** à l'IFSIC, par le CIES - Grand Ouest (Centre d'Initiation à Enseignement Supérieur), *192 heures équivalent TD*

Recherche

- 2005 - 2008 **Doctorant en Informatique** à l'IRISA, sous la direction de Jean-Pierre Talpin,
Sujet : *“Calculs formels pour Signal et modèle temps réel”*
- 2004 - 2005 **Stage de Master 2 Recherche Informatique** à l'IRISA, sous la direction de Bertrand Jeannet,
Sujet : *“Représentation d'ensembles de valeurs numériques en vérification de programmes”*

Langues

Français : langue maternelle, Anglais : bon niveau, Espagnol : niveau débutant

Autres activités

- Activités musicales :** Depuis 2000 Création et mixage en musique électronique
Juin 2000 Diplôme de fin d'études en formation musicale
au Conservatoire de Nantes
1990 - 2000 10 ans de pratique et de formation à la Trompette
au Conservatoire de Nantes

- Activités sportives :** badminton, vélo, ski, handball, natation,
formation au BNSSA (Brevet National de Sécurité et de Sauvetage Aquatique)

2 Enseignement

2005 - 2008 **Moniteur en Informatique** à l'IFSIC, 192 heures équivalent TD

Année	Niveau	Matière	Nombre d'heures		
			Cours	TD	TP
2005 - 2008	Licence 1 PCGI	Informatique Scientifique		60	60
2005 - 2008	Licence 3 Miage	Programmation Web			76
2006 - 2007	Master 2 Professionnel	Conception d'applications web	2		8
2006 - 2007	Licence 3 Informatique	Algorithmique des graphes		22	
2005 - 2006	DIIC 2ème année	Méthodes de développement industriel		12	
Total			2	94	144

– *Informatique Scientifique*

Programmation scientifique pour ingénieur, en Licence 1 PCGI, avec Laurent Perraudou

Travaux Dirigés : 60 heures et Travaux Pratiques : 60 heures

Ce cours est une initiation à l'informatique basée sur l'utilisation du logiciel Mathematica. Les notions abordées visent à faire comprendre aux étudiants ce qu'est un programme informatique, par une approche fonctionnelle. L'objectif du cours est que les étudiants aient assimilé les notions d'expression, de type, de fonction, d'expressions conditionnelles, ainsi que le calcul itératif.

– *Programmation Web*

Conception de pages web, en Licence 3 Miage, avec Yannick Le Tertre

Travaux Pratiques : 76 heures

Ce cours est un perfectionnement à la conception de page web. Les étudiants de Miage apprennent à mettre en forme une page avec html, à la rendre dynamique avec JavaScript et à améliorer l'ergonomie avec la nouvelle technologie ajax. L'interactivité avec le serveur est étudié avec php et une base de données MySQL.

– *Conception d'applications web*

Conception de pages web, en Master 2 Professionnel avec Yannick Le Tertre et Yves Bekkers

Cours : 2 heures et Travaux Pratiques : 8 heures

Ce cours passe en revue diverses méthodes de conception d'applications complémentaires ou concurrentes (html, JavaScript, php, ajax, flash, lazlo). J'ai construit et réalisé le début de ce cours/TP avec l'aide de Yannick Le Tertre, qui a été achevé par Yves Bekkers.

– *Algorithmique des graphes*

Théorie, algorithmes et applications des graphes en Licence 3 Informatique, avec Rumen Andonov

Travaux Dirigés : 22 heures

Ce cours vise à enseigner aux étudiants les bases mathématiques de la théorie des graphes, les différentes méthodes d'implémentation et les principaux algorithmes de graphes. L'objectif est de donner aux étudiants une certaine aisance à la manipulation de graphes et à l'adaptation d'algorithmes de graphes à un problème précis.

– *Méthodes de développement industriel*

Méthodes de conceptions et UML, en DIIC 2^e année (diplôme d'ingénieur de l'IFSIC), avec Jean-Marc Jezequel

Travaux Dirigés : 12 heures

La première partie de ce cours vise à donner un aperçu général de toutes les phases que composent le cycle de développement d'un logiciel, et plus particulièrement de celles qui précèdent la phase d'implémentation. De cette introduction découle l'étude d'UML, des patrons de conception et de leurs applications.

2005 - 2008 **Formations** à l'activité d'enseignant-chercheur par le CIES-Grand ouest et l'IRISA (pédagogie, communication scientifique orale et écrite, poster, réunion, intervention dans un lycée ...)

2003 - 2004 **Cours particuliers** d'informatique, niveau Bac+3

2001 - 2002 **Cours particuliers** de mathématiques, niveau Bac

3 Recherche

Publications

Conférence internationale :

- [MTGG08] H. Metivier, J-P. Talpin, T. Gautier and P. Le Guernic. “Analysis of periodic clock relations in polychronous systems” In *DIPES’08, IFIP, Distributed Embedded Systems : Design, Middleware and Ressources*, Milano, Italy, September 2008

Autres participations et documents :

- [EJCP06] École Jeunes Chercheurs en Programmation. 10 jours de formations, de présentations et d’exposés, IRIT, Toulouse, France, Juin 2006
- [Met05] H. Metivier. “Représentation d’ensembles de valeurs numériques en vérification de programmes” Rapport de Master 2 Recherche, IRISA, Rennes, France, Juin 2005

Activités de recherche

- 2005 - 2008 **Doctorant en Informatique** à l’IRISA, sous la direction de Jean-Pierre Talpin, au sein du Projet Espresso (Environnement de Spécification de Programmes Réactifs SynchrOnes), Sujet : “*Calculs formels pour Signal et modèle temps réel*”

Mots clés : Programmation réactive et synchrone, calculs symboliques, analyse statique, abstraction de programme, vérification automatique, langage régulier, automate.

Le langage Signal, développé dans le cadre du projet Espresso, et son modèle polychrone (multi-horloge) permettent d’implémenter et de spécifier des systèmes embarqués. Sous ce formalisme, les systèmes peuvent être représentés à différents niveaux d’abstractions afin d’analyser les contraintes fortes de sûreté que doivent respecter ces systèmes. Signal utilise l’approche *synchrone* dans l’objectif de favoriser l’analyse, la réutilisation et le remplacement de composants. Pour cela, Signal et son calcul d’horloges est basé sur un cadre mathématique.

Le cadre de ce doctorat se situe au niveau de *l’utilisation de mémoire* (buffer). Les systèmes embarqués ont pour beaucoup des contraintes temporelles fortes à respecter mais également un espace mémoire limité. L’objectif est alors de montrer, dès la conception d’un tel système, qu’il n’y aura pas de dépassement de la taille des mémoires et que les contraintes de temps seront respectées.

Le début de mon doctorat a été consacré à l’étude de ces problèmes liés aux mémoires sur des systèmes embarqués effectuant des tâches répétitives (appelés *périodiques*). La conversion à la volée de la définition de vidéos, les composants contrôlant les moteurs ou des composants des satellites sont des exemples de systèmes embarqués périodiques. Les mots binaires infinis ultimement périodiques sont des mots de la forme $u(v)$, où u est un préfixe et v une période. Par exemple, le mot 01(100) représente la suite infinie 01 100 100 100 \dots . Ces mots sont intéressants car ils possèdent quelques bonnes propriétés qui permettent assez facilement la définition d’opérations sur des couples de mots quelque soit la longueur des préfixes et périodes de chacun des mots. Ces mots peuvent aussi servir à décrire les contraintes à respecter par des horloges dont le comportement est périodique, en considérant que l’atome 0 est associé à absent et l’atome 1 à présent. Ces contraintes temporelles périodiques ont été formalisées et s’expriment sous la forme $x = 01(100)@y$ où x et y sont des horloges, cette contrainte signifie “la présence de x correspond aux atomes du mot 01(100) recalés sur les présences de y ”. Une algèbre de mots infinis ultimement périodiques a été définie pour permettre l’extraction et la manipulation de telles contraintes. Ces mots binaires ont été étendus aux mots *ternaires* sur $\{0, -1, 1\}$ afin d’également prendre en compte la valeur des booléens dans l’analyse (les atomes de $\{0, -1, 1\}$ signifient respectivement absent, présent et faux, présent et vrai). Enfin, conjuguée au calcul d’horloge existant de Signal, cette algèbre permet d’effectuer une analyse des systèmes périodiques, qui permet de vérifier si le système peut fonctionner avec des

mémoires de taille bornée. Et dans ce cas, une taille maximale des mémoires à insérer dans le système peut être calculée. Ces résultats ont été publiés dans [MTGG08].

L'analyse basée sur l'algèbre de mots infinis ultimement périodiques est précise pour étudier l'utilisation des mémoires mais limitée aux systèmes embarqués périodiques. Dans la suite, l'idée a été de trouver un autre cadre pour étudier un ensemble plus grand de systèmes embarqués. Le calcul d'horloges de Signal est très fin dans l'expression des *dépendances instantanées* (l'approche synchrone fait l'hypothèse que les calculs au cours d'un même instant sont d'une durée négligeable). Mais l'expression des *dépendances temporelles* est actuellement moins riche. Ces dépendances temporelles ont pu être caractérisées par des schémas d'horloges qui représentent les dépendances d'une variable sur une autre quand elles utilisent des mémoires. Un schéma d'horloges est exprimé sous la forme de suite d'horloges et ils représentent à la fois les dépendances instantanées et les dépendances temporelles. Des schémas sont extraits de chaque équation de base du langage Signal. Toutes les dépendances sont représentées sur un graphe pondéré avec des schémas d'horloges. L'objectif est alors de calculer l'ensemble des schémas d'horloges correspondant aux dépendances d'une variable sur une autre (par exemple les dépendances des entrées sur les sorties d'un composant). L'idée a été de représenter ces ensembles de schémas d'horloges avec des langages réguliers, appelés *langage d'horloges*. Le calcul du langage d'horloges correspondant à une dépendance d'une variable sur une autre a été défini grâce à des algorithmes adaptés d'algorithmes sur les langages réguliers et à quelques opérations sur les schémas et langages. Les applications futures sont le calcul de la taille maximale des mémoires (comme l'analyse basée sur les mots infinis ultimement périodiques mais sur un ensemble de systèmes embarqués plus grand) mais aussi la génération d'automate ou de code associé, le pré-calcul (effectuer des calculs plus tôt pour répartir les coûts de calcul) et des analyses sur les flots de données.

2004 - 2005 **Stage de Master 2 Recherche Informatique** à l'IRISA, sous la direction de Bertrand Jeannet,
Sujet : “*Représentation d'ensembles de valeurs numériques en vérification de programmes*”

Mots clés : Interprétation abstraite, analyse statique, domaines numériques abstraits, algorithmes de graphe.

Le cadre de ce stage de Master 2 Recherche se place dans le contexte de la vérification de programmes manipulant des variables numériques. On s'est intéressé à la vérification de *propriétés d'invariance* sur ces variables. En raison de problèmes d'indécidabilité ayant rapport avec la représentation des valeurs et la terminaison des programmes, on se place dans le cadre de *l'interprétation abstraite*. Ce domaine propose de représenter de manière approchée des ensembles d'états par des valeurs de domaines abstraits. Pour approximer des sous-ensembles de \mathbb{R}^n , on peut utiliser comme domaines abstraits des intervalles, des octogones ou des polyèdres convexes. Le choix d'un domaine abstrait relève d'un compromis entre précision et coût des calculs. Ainsi l'utilisation de polyèdres convexes offre une grande précision, mais au prix d'une complexité élevée, en $\mathcal{O}(2^n)$. On peut à la place utiliser le domaine des *octogones*, dont la complexité est en $\mathcal{O}(n^3)$ qui peut représenter toutes les propriétés du type $\pm x \pm y \leq c$, où x, y sont des variables et c une constante, mais qui ne peuvent pas représenter la propriété linéaire $x + 2y \leq c$.

On a voulu étudier une technique plus fine pour contrôler la complexité des analyses en fonction du nombre de variables pour le domaine des octogones. On peut approcher un octogone $\text{Oct}(x, y, z, w)$ en 4 dimensions par un produit cartésien de 2 octogones en 2 dimensions $\text{Oct}(x, y) \otimes \text{Oct}(y, z)$. On passe d'une complexité de $\mathcal{O}(n^3)$ à une complexité en $\mathcal{O}(2 \cdot (n/2)^3)$. Cependant la propriété $x = w$ n'est plus représentable. L'idée est alors de partitionner les variables avec une certaine redondance. Par exemple, on peut approcher $\text{Oct}(x, y, z, w)$ par un produit dit *réduit* $\text{Oct}_1(x, y, z) \otimes_r \text{Oct}_2(y, z, w)$. Par l'intermédiaire des variables partagées x et y , il est ici possible de représenter de manière indirecte des relations entre x et w . En revanche la redondance doit être prise en compte : le produit $\{x = y = z \geq 0\} \otimes_r \{y \leq z \leq w \leq 0\}$ peut être réduit en $\{x = y = z = 0\} \otimes_r \{y = z = w = 0\}$

La difficulté a été de définir l'opération de réduction de ce nouveau domaine abstrait, appelé *octogones creux*, avec des adaptations d'algorithmes de graphes et de prouver cette réduction. L'abstraction des opérations concrètes (affectation, condition, itération) d'un langage de programmation a également été défini. La complexité de l'opération de réduction du domaine des octogones est en $\mathcal{O}(n^3)$, et celle du domaine des octogones creux est en $\mathcal{O}(ne + n^2 \log(n))$, où n est le nom de variable et e le nombre

de contraintes exprimées (donc $e < n^2$). Ce nouveau domaine est donc moins précis mais aussi moins coûteux en temps que le domaine des octogones. Une librairie de ces travaux a été implémentée dans une suite de ce stage, afin d'être intégrée dans l'outil Nbac de Bertrand Jeannet.