Raphaël Monat

ATER au LIP6, Sorbonne Université
Mots-clés : Sciences du logiciel, Méthodes formelles, Sûreté du logiciel
23 bd de Brandebourg, 94200 Ivry-sur-Seine
raphael.monat@lip6.fr | rmonat.fr | +33 6 42 33 06 18 | 28/07/1995 | gitlab.com/rmonat

Formations

Thèse sous la direction d'Antoine Miné — LIP6, Sorbonne Université
Analyse statique de type et de valeur, par interprétation abstraite, de programmes Python avec des bibliothèques C. — Septembre 2018 – Novembre 2021

	Antoine Miné	Sorbonne Université, France	Directeur
	Isabella Mastroeni	Università di Verona, Italie	Rapportrice
	Anders Møller	Aarhus Universitet, Danemark	Rapporteur
Jury :	Emmanuel Chailloux	Sorbonne Université, France	Président du jury
	Francesco Logozzo	Facebook Seattle, USA	Examinateur
	Peter Müller	ETH Zürich, Suisse	Examinateur
	Alan Schmitt	Inria Rennes, France	Examinateur

Master 2 Master Parisien de Recherche en Informatique — Université Paris Diderot
Moyenne 16.81/20 — 2017-2018

Master 2 Informatique Fondamentale — École Normale Supérieure de Lyon
Moyenne 16.49/20 — 2016–2017

Master 1 Informatique Fondamentale — École Normale Supérieure de Lyon
Moyenne 16.36/20 — 2015–2016

Licence 3 Informatique Fondamentale — École Normale Supérieure de Lyon
Moyenne 16.43/20 — 2014–2015

Entrée à l'École Normale Supérieure de Lyon sur le concours informatique — École Normale Supérieure de Lyon
2014

Classes préparatoires MPSI/MP* option informatique — Lycée Louis-le-Grand, Paris
2012–2014

Baccalauréat scientifique, mention très bien — Lycée Aristide Bergès, Isère
2012

Expériences professionnelles

Attaché temporaire d'enseignement et de recherche — LIP6, Sorbonne Université
Recherche dans l'équipe APR (dirigée par Antoine Miné). Enseignements dans l'UFR d'ingénierie (192h). — Septembre 2021 - Août 2022

Doctorant contractuel avec mission complémentaire d'enseignement — LIP6, Sorbonne Université
Recherche sous la direction d'Antoine Miné. Enseignements dans l'UFR d'ingénierie (64h/an). — Septembre 2018 - Août 2021

Stage de recherche de M2 sous la direction d'Antoine Miné — LIP6, Sorbonne Université
Création d'une analyse relationnelle de typage pour Python. — Mars – Juin 2018

J'ai défini de manière théorique une analyse de programmes Python proche d'un système de type avec du polymorphisme paramétrique. Cette analyse utilisait un unique domaine abstrait monolithique, qui pouvait exprimer que certaines variables avaient un même type, celui-ci variant dans un ensemble. L'implémentation permettait d'obtenir des résultats sur des petits programmes de moins de 50 lignes de code. Ma première année de thèse a consisté à reprendre ces travaux pour séparer le domaine abstrait en plusieurs domaines indépendants, et améliorer significativement le passage à l'échelle de l'analyse.

Stage de recherche de M2 sous la direction d'Eva Darulova — MPI-SWS, Sarrebruck, Allemagne
Extension d'une formalisation en Coq et HOL4 pour un synthétiseur de programmes sur les flottants. — Février – Juin 2017

Le groupe de vérification automatique et d'approximation (AVA) du Max Planck Institute for Software Systems, dirigé par Eva Darulova, développe un outil appelé Daisy, permettant de compiler des programmes numériques idéalisés, définis sur les réels, en des programmes calculant sur les flottants, avec une marge d'erreur donnée en sortie par le compilateur. Ce compilateur était capable d'optimiser l'équilibre entre précision et performance des programmes grâce à l'utilisation de plusieurs précisions pour les flottants. Un doctorant du groupe, Heiko Becker, avait établit une formalisation (en Coq et HOL) permettant de vérifier via des certificats que la compilation utilisant une analyse d'intervalles et une précision fixée était correcte. J'ai généralisé cette formalisation pour prouver la compilation correcte dans le

cas d'une précision mixte. J'ai aussi travaillé à formaliser une analyse basée sur l'arithmétique affine. Une publication de groupe sur ce travail a été acceptée à FMCAD 2018.

Stage de recherche de M1 sous la direction de Hongseok Yang — Oxford, Royaume-Uni

Inférence variationnelle pour les langages de programmation probabilistes. — Mai – Juillet 2016

Les modèles probabilistes sont utilisés dans de nombreux domaines pour résoudre différents problèmes, allant de la reconnaissance d'images au diagnostic de maladies. L'utilisation de modèles permet de séparer l'encodage du problème (dans un modèle probabiliste) de sa résolution, et d'avoir des algorithmes d'inférence spécifiques à chaque classe de modèle. Une approche à l'intersection des langages de programmation et de l'apprentissage automatique est la programmation probabiliste. Cette approche permet de généraliser la notion de modèle probabiliste, et a pour but de pouvoir réaliser automatiquement l'inférence à l'exécution des programmes probabilistes. J'ai travaillé avec Hongseok Yang à formaliser les programmes probabilistes sous une forme de système de transition probabiliste. Le problème d'inférence peut être vu comme un problème d'optimisation, que nous avons simplifié pour dériver un nouvel algorithme d'inférence variationnel général pour les programmes probabilistes.

Stage de recherche de L3 sous la direction d'Antoine Miné — Antique, ENS Ulm

Développement d'une analyse de programmes concurrents avec une mémoire à cohérence séquentielle. — Juin – Juillet 2015

J'ai utilisé le cadre de l'interprétation abstraite pour développer, implémenter et tester de nouvelles méthodes pour analyser plus précisément des programmes concurrents. Le modèle mémoire considéré est celui de la cohérence séquentielle. J'ai étendu une approche d'analyse modulaire sur les threads, développée précédemment par Antoine et basée sur la logique "Rely-Guarantee", pour améliorer la précision de l'analyse dans la prise en compte des interférences entre les threads. J'ai développé à partir de zéro un prototype, appelé Batman, permettant d'analyser des programmes concurrents jouets. Ce prototype est écrit en OCaml et utilise la bibliothèque de domaines abstraits numériques Apron. J'ai comparé les résultats obtenus à ceux d'un autre outil appelé ConcurInterproc. J'ai étendu ces travaux en dehors de mon stage et publié ces résultats avec Antoine Miné à VMCAI 2017.

Enseignements

Sauf mention contraire, les enseignements détaillés ci-dessous ont été effectués à Sorbonne Université, dans l'UFR d'ingénierie, auprès d'étudiants en informatique, et les enseignements en master ont été effectués dans le cadre du master STL "Sciences et Techniques du Logiciel".

Acronymes :

- TME : travaux sur machine encadrés
- CM : cours magistral

Type d'enseignement effectué par niveau

	L1	L2	L3	M1	M2
CM					4h
TD	60h	52h	30h	60.5h	14h
TME	58h	71h	29.5h		24h
Total	118h	104h	59.5h	60.5h	42h

Type d'enseignement effectué par thème

	L1	L2	L3	M1	M2
Info. théorique	20h	64h		52h	
Programmation	98h	40h	31.5h		
Spécialité			28h	8.5h	42h
Total	118h	104h	59.5h	60.5h	42h

Info. théorique : mathématiques discrètes, logique, algorithmique
Spécialité : compilation, analyse et preuve de programmes

L1 **Éléments de Programmation 2 (C)**, Chargé de TD/TME (20 étudiant·e·s, 38.5h) — 2021–2022
Correction des examens.

L1 **Éléments de Programmation 1 (Python)**, chargé de TD/TME (26 étudiant·e·s, 38.5h) — 2019–2020
Création et correction de 4 interrogations de TD, et correction des examens.

L1 **Éléments de Programmation 1 (Python)**, chargé de TME (29 étudiant·e·s, 19.25h) — 2018–2019
Correction d'une épreuve machine. Amélioration des supports.

L1 **Éléments de Programmation 1 (Python)**, remplacement en TD (30 étudiant·e·s, 1.75h) — 2018–2019

L1 **Ateliers de Recherche Encadrée**, chargé de TD/TME (4 binômes, 20h) — 2018–2019
Création des sujets proposés (arbres bicolores et enveloppes convexes).

L2 **Logique**, Chargé de TD/TME (33 étudiant·e·s, 19.25h) — 2021–2022
Correction des examens.

L2 **Mathématiques Discrètes**, chargé de TD/TME (33 étudiant·e·s, 44.5h) — 2021–2022
Deux interrogations de TD à préparer et corriger. Correction des examens et des projets. Amélioration des supports.

L2 **Programmation Fonctionnelle (OCaml)**, chargé de TD/TME (26 étudiant·e·s, 19.25h) — 2021–2022

Création de 2 sujets de TME et intégration dans la plateforme LearnOcaml. Correction des examens. Amélioration des supports.

L2 **Programmation Fonctionnelle (Ocaml)**, remplacement ponctuel en TME (27 étudiant·e·s, 1.75h) 2019–2020

L2 **Fonctions et Procédures de Calcul (Ocaml)**, chargé de TME (10 étudiant·e·s, 19.25h) 2018–2019
Co-création d'un sujet de projet, et correction du projet.

L3 **Programmation Objet Avancée (Java)**, chargé de TD/TME (27 étudiant·e·s, 31.5h) 2020–2021
Correction des examens. Amélioration des supports.

L3 **Compilation**, chargé de TD/TME (12 étudiant·e·s, 28h) 2018–2019
Correction des examens. Amélioration des supports.

M1 **Algorithmique Avancée**, chargé de TD (31 étudiant·e·s, 32h) 2021–2022
Correction des examens. Amélioration des supports. Correction d'un projet de programmation.

M1 **Algorithmique Avancée**, chargé de TD (31 étudiant·e·s, 20h) 2019–2020
Correction des examens. Amélioration des supports.

M1 **Projet STL**, co-encadrant de binômes sur des projets (8.5h) 2019–2020

M2 **MPRI**[1] **: Interprétation abstraite**, intervenant CM (15 étudiant·e·s, 2h) 2021–2022
Création du support pour mon intervention (résultats obtenus durant ma thèse). **Cours donné en anglais.**

M2 **Typage et Analyse Statique**, chargé de TD/TME (31 étudiant·e·s, 28h) 2021–2022
Refonte des sujets de TD. Correction d'un projet, et oral d'évaluation des étudiants sur des articles de recherche.

M2 **Spécification et Validation de Programmes (Coq)**, chargé de TD/TME (10 étudiant·e·s, 10h) 2021–2022
Création des quatre sujets de TME.

M2 **Groupe de Recherche en Algorithmique et en Programmation**, intervenant CM (12 étudiant·e·s, 2h) 2019–2020
Création du support pour mon intervention (présentation de mon sujet de thèse et de ce qu'est une thèse).

Publications

Dans le domaine des méthodes formelles dont je fais partie, *les articles sont majoritairement publiés dans des conférences internationales avec comité de sélection*. Les articles dans les journaux avec comités de sélection sont plus rares. *Les auteur·rice·s sont usuellement ordonné·e·s par contribution décroissante.* Pour les publications parlant de travaux de groupe sur Mopsa (VSTTE 2019 et JFLA 2021), nous avons choisi d'utiliser un ordre alphabétique. Mes publications sont ordonnées par ordre chronologique décroissant. SAS est la conférence spécialisée dans le domaine de l'analyse statique par interprétation abstraite.

Politique de publication. Je préfère soumettre mes travaux à des conférences qui ont des politiques de publication ouverte avec des frais raisonnables pour les auteurs (proches du prix coûtant, $\leq 100€$), ou qui sont atteignables sans prendre l'avion. ECOOP[2] est la conférence du domaine respectant le plus ces critères, puisqu'elle a lieu en Europe et qu'elle utilise les actes ouverts LIPIcs[3] (les autres conférences du domaine utilisent l'ACM ou Springer comme éditeurs).

Légende :

- Titre en noir : auteur principal (gris sinon).
- Artefacts logiciels : (ACM), (autres).
- article double colonne (sinon).
- article démonstration d'un outil.

Les titres sont cliquables et renvoient aux informations de publication sur mon site (avec des liens vers une version publique et la version de l'éditeur, au minimum) ; les acronymes des conférences pointent vers le programme de l'événement correspondant ; les badges d'artefacts logiciels vers l'artefact validé par les pairs.

Conférences Internationales avec Comité de Lecture

A Multilanguage Static Analysis of Python Programs with Native C Extensions SAS 2021
Raphaël Monat, *Abdelraouf Ouadjaout, Antoine Miné* 23 pages

A Modern Compiler for the French Tax Code CC 2021
Denis Merigoux, Raphaël Monat, *Jonathan Protzenko* 11 pages

Static Type Analysis by Abstract Interpretation of Python Programs ECOOP 2020
Raphaël Monat, *Abdelraouf Ouadjaout, Antoine Miné* 27 pages

1. MPRI : master parisien de recherche en informatique, spécialisé en informatique fondamentale et à destination des étudiants poursuivant dans la recherche. Formation supervisée par l'université de Paris, ENS Ulm, ENS Paris-Saclay, École Polytechnique, Telecom Paris.
2. `https://ecoop.org/`
3. `https://www.dagstuhl.de/en/publications/lipics`

Combinations of Reusable Abstract Domains for a Multilingual Static Analyzer (Invité) — VSTTE 2019
Matthieu Journault, Antoine Miné, Raphaël Monat*, Abdelraouf Ouadjaout* — 16 pages

A Verified Certificate Checker for Finite-Precision Error Bounds in Coq and HOL4 — FMCAD 2018
Heiko Becker, Nikita Zyuzin, Raphaël Monat*, Eva Darulova, Magnus O. Myreen, Anthony Fox* — 8 pages

Precise Thread-Modular Abstract Interpretation of Concurrent Programs using Relational Interference Abstractions — VMCAI 2017
Raphaël Monat*, Antoine Miné* — 17 pages

Workshop Internationaux avec Comité de Lecture

Value and Allocation Sensitivity in Static Python Analyses — SOAP 2020
Raphaël Monat*, Abdelraouf Ouadjaout, Antoine Miné* — 6 pages

Conférences Nationales avec Comité de Lecture

Mlang : an Open-Source Toolchain for the Income Tax Computation — JFLA 2021
Denis Merigoux, Raphaël Monat — 2 pages

Démonstration de la plateforme Mopsa d'analyse statique de programmes par interprétation abstraite — JFLA 2021
Matthieu Journault, Antoine Miné, Raphaël Monat*, Antoine Miné* — 2 pages

Étude formelle de l'implémentation du code des impôts — JFLA 2020
Denis Merigoux, Raphaël Monat*, Christophe Gaie* — 16 pages

Manuscrits et rapports techniques

Static Type and Value Analysis by Abstract Interpretation of Python Programs with Native C Libraries — Manuscrit de thèse
Raphaël Monat — 275 pages

Static Analysis by Abstract Interpretation Collecting Types of Python Programs — Rapport de stage (M2 2018)
Raphaël Monat — 20 pages

Certificate Checking in Coq and HOL4 for Static Analyses of Mixed-Precision Floating-Point Arithmetic — Rapport de stage (M2 2017)
Raphaël Monat — 24 pages

Variational Inference in Probabilistic Programs : Formal Derivation of a Black-box Approach — Rapport de stage (M1)
Raphaël Monat — 26 pages

Thread-Modular Analysis Designing Relational Abstractions of Interferences — Rapport de stage (L3)
Raphaël Monat — 21 pages

Exposés

Le temps indiqué est celui de l'exposé, sans les questions. Titres cliquables vers mon site.

Exposé invité

01/12/21 **A Multilanguage Static Analysis of Python/C Programs with Mopsa**, 25 minutes — **Facebook TAV**

"Testing and Verification (TAV) Symposium brings together academia and industry in an open environment to exchange ideas and *showcase the top experts from testing and verification scientific research and practice*." (source)

Exposés dans des conférences

18/10/21 **A Multilanguage Static Analysis of Python Programs with Native C Extensions**, 15 minutes — SAS
07/04/21 **Mlang : an Open-Source Toolchain for the Income Tax Computation**, 15 minutes — JFLA
03/03/21 **A Modern Compiler for the French Tax Code**, 12 minutes — CC
15/11/20 **Static Type Analysis by Abstract Interpretation of Python Programs**, 15 minutes — ECOOP
15/06/20 **Value and Allocation Sensitivity in Static Python Analyses**, Best Presentation Award. 20 minutes — SOAP
30/01/20 **Étude formelle de l'implémentation du code des impôts**, 20 minutes — JFLA
07/10/19 **Static Type Analysis of Python Programs : A Type Abstract Domain for Python**, 20 minutes — DS@FM

Séminaires

28/04/22	**Formal methods for realistic systems : a study of two cases**, 45 minutes	LIP, ENS de Lyon
17/03/22	**A Multilanguage Static Analysis of Python/C Programs with Mopsa**, 30 minutes	SRG, Imperial, London
31/01/22	**Static Type and Value Analysis by Abstract Interpretation of Python Programs with Native C Libraries**, 40 minutes	MTV, LaBRI, Bordeaux
31/01/22	**A Multilanguage Static Analysis of Python/C Programs with Mopsa**, 30 minutes	SyCoMoRES, CRIStAL, Lille
02/12/21	**A Modern Compiler for the French Tax Code**, 40 minutes	IRILL, Paris
29/11/21	**Static Analysis of Python Programs with Native C Libraries**, 30 minutes	CASH, LIP, Lyon
26/11/21	**A Multilanguage Static Analysis of Python/C Programs with Mopsa**, 30 minutes	Binsec, CEA, Saclay
19/11/21	**A Multilanguage Static Analysis of Python/C Programs with Mopsa**, 30 minutes	Celtique, IRISA, Rennes
05/07/21	**A Multilanguage Static Analysis of Python Programs with Native C Extensions**, 20 minutes	LIP6, Paris
12/03/21	**Mopsa, a Multi-lingual Static Analysis Platform**, 20 minutes	GT LVP, GDR GPL
04/11/19	**Formal study of the French tax code's implementation**, 60 minutes	INRIA Cambium, Paris
23/10/18	**Static Analysis by Abstract Interpretation of Dynamic Programming Languages**, 20 minutes	LIP6, Paris

Poster

02/07/19	**Semantics & Static Type Analysis of Python Programs**	Journée doctorants SIF

Participation à la vie de la communauté

Membre du collège code source et logiciel libre du comité national pour la science ouverte

Mars 2022-...

Conseil du laboratoire du LIP6 (membre élu, 1h/mois)

Avril 2021-...

Conseil des doctorants du LIP6 (1h/mois)

Avril 2021-...

Comité de programme

SAS'22

Comité d'évaluation d'artefacts logiciels (30 à 40h par conférence)

SPLASH'22, PLDI'22, CAV'22, ECOOP'21, PLDI'21, POPL'21, SAS'20

Comité de revue externe

SPLASH'22

Relecteur externe

Science of Computer Programming, SAS'21, ACM TECS, SOAP'21, LOPSTR'19

Étudiant bénévole

POPL'17

Logiciels développés

Noms des logiciels cliquables.

Mopsa, plateforme open-source d'analyse statique de programmes — Depuis Septembre 2018

Contributeur principal. 60kLoc Ocaml. Développement et maintenance de l'analyse Python (13kLoc) et multilangage Python/C (2.7kLoc). Licence LGPL v3. Développement suite à l'octroi d'un financement "Consolidator Grant" de l'ERC à Antoine Miné.

Autoévaluation INS2I : `A3 SO4 SM3 EM2 SDL4 DA4 CD4 MS4 TPM4`.

Mlang, compilateur open-source pour le code des impôts — Depuis Mai 2019

Contributeur principal. 10kLoc OCaml. Mlang permet de répliquer le calcul de l'impôt sur le revenu. Licence GPL v3.

Autoévaluation INS2I : `A4 SO2 SM3 EM4 SDL4 DA4 CD4 MS4 TPM4`. Accompagnement de la DGFiP pour la migration d'un ancien compilateur vers Mlang (lien vers communiqué).

Daisy, compilateur de programmes numériques sur les réels vers des flottants. — Février – Juin 2017

J'ai participé au développement de Daisy. Daisy est écrit en 14kLoc Scala.

Autoévaluation INS2I : `A3 SO3 SM3 EM3 SDL4 DA1 CD2 MS1 TPM1`.

FloVer, vérification mécanisée de certificats de correction générés par Daisy. Février – Juin 2017

J'ai généralisé la formalisation de FloVer, qui contient 10kLoc HOL et 25kLoc Coq. Je faisais partie des deux développeurs principaux (avec Heiko Becker) durant ma période de stage. La généralisation effectuée avait augmenté la taille des formalisations de 30% pour Coq et 18% pour HOL. Daisy et FloVer ont été l'objet d'une publication de groupe à FMCAD'18.

Autoévaluation INS2I : `A3 SO3 SM3 EM3 SDL4 DA2 CD2 MS1 TPM1`.

Batman, preuve de concept d'un analyseur statique capable d'analyser les différents threads de manière modulaire. Juin 2015 – Janvier 2017

Seul contributeur. 5,5kLoc OCaml. Licence GPL v3. Ce prototype a été utilisé pour l'évaluation expérimentale de notre papier à VMCAI'17.

Autoévaluation INS2I : `A1 SO2 SM1 EM1 SDL4 DA4 CD4 MS4 TPM4`.

Langues

- Français, langue maternelle
- Anglais, niveau B2 certifié par le CLES
- Allemand, niveau B1

Programmation & outils

- Pratique courante : OCaml, Python, C, LaTeX, Beamer
- Pratique régulière : Java, Bash, GoHugo, OBS (cours à distance et enregistrement de présentations virtuelles), assistants de preuve Coq et HOL4

Engagements associatifs

Animation bénévole de cours d'escrime pour étudiants, 2h à 3h30 par semaine, 40 semaines par an 2018–2020

Suivi d'une formation fédérale à l'animation de cours d'escrime, 70 heures 2018–2019

Médaille de bronze lors de la finale régionale (SWERC) du concours de programmation ACM ICPC, 2017

Rang 12/75 (médailles jusqu'à la place 14), équipe ENSL2 avec Simon Mauras et Jean-Yves Franceschi.

Délégué lors du M1 d'Informatique Fondamentale à l'ENS de Lyon 2015–2016