2024

DOSSIER DE CANDIDATURE

APPLICATION

Cochez le concours sur lequel vous candidatez
Check the competition exam for which you are applying

☒ **ISFP - (Inria Starting Faculty Position / Inria Starting Faculty Position)**

☒ **CRCN - (Chargés de recherche de classe normale / Young graduate scientist position)**

☐ **DR2 - (Directeurs de recherche de deuxième classe / Senior researcher position)**

Nom[1] **:** NGUYEN
Last name

Prénom : Le Thanh Dung
First name

Sexe : ☐ **F** ☒ **M**
Sex

Nom utilisé pour vos publications (facultatif) : Lê Thành Dũng (Tito) Nguyễn
Name used for your publications (optional):

[1] Il s'agit du nom usuel figurant sur vos pièces d'identité
It is the name appearing on your identity cards

DEPOT DE VOTRE CANDIDATURE
SUBMITTING YOUR APPLICATION

Le dossier de candidature doit comprendre :

- Formulaire 1 : Parcours professionnel
- Formulaire 2 : Description synthétique de l'activité antérieure
- Formulaire 3 : Contributions majeures
- Formulaire 4 : Programme de recherche
- Formulaire 5 : Liste complète des contributions

CRCN & ISFP :

- Les rapports de thèse ou de doctorat (si disponibles)
- Une copie des derniers titres et diplômes
- Une photographie récente de la candidate / du candidat (facultative)

DR2 :

- Les rapports d'habilitation à diriger des recherches (si applicable)
- Une copie des derniers titres et diplômes
- Une photographie récente de la candidate / du candidat (facultative)

The application file must include:

- *Form 1: Professional history*
- *Form 2: Summary of your past activity*
- *Form 3: Major contributions*
- *Form 4: Research program*
- *Form 5: Complete list of contributions*

CRCN & ISFP:

- *PhD dissertation reports (when available)*
- *A copy of most recent titles and diplomas*
- *A recent photography of the applicant (optional)*

DR2:

- *Habilitation dissertation reports (if applicable)*
- *A copy of most recent titles and diplomas*
- *A recent photography of the applicant (optional)*

SOMMAIRE / ***SUMMARY***

Formulaire 1 — PARCOURS PROFESSIONNEL
Form 1 — PROFESSIONAL HISTORY

1) Parcours Professionnel / *Professional history*

Situation professionnelle actuelle / ***Current professional status***

Statut et fonction[2] / *Position and Status*[2]: Post-doctorant
Etablissement (ville - pays) / *Institution (city - country)*: École normale supérieure de Lyon, France
Date d'entrée en fonction / *Start*: 1er septembre 2022
[ ] Sans emploi / *Without employment*

Expériences professionnelles antérieures /***Previous professional experiences***

Déroulez votre parcours professionnel antérieur à Inria, chez Inria, en détachement ou en mise à disposition.
Detail your professional history, before Inria, at Inria, on secondment or leave.

Date début *Start*	Date fin *End*	Etablissement *Institution*	Fonction et statut[2] *Position and status*[2]
03/2022	08/2022	École polytechnique	Post-doctorant
09/2021	02/2022	IRISA (Rennes)	Ingénieur de recherche (CDD)
09/2018	08/2021	Université Paris XIII	Contrat doctoral + mission d'enseignement (64h/an)
09/2012	08/2018	ÉNS Paris (rue d'Ulm)	Élève fonctionnaire stagiaire + 2 années de césure

Nombre d'années d'exercice des métiers de la recherche après la thèse / *Number of years of prof<essional research experience after the PhD:* deux (2)

Si vous le jugez nécessaire, vous pouvez apporter des précisions sur votre parcours ou votre situation actuelle **en quelques lignes**. / *If you find it necessary, you can give some details about your professionnal history or your current situation **in a few lines**.*

2) Interruptions de carrière/*Career breaks*

Date début *Start*	Date fin *End*	Motif de l'interruption/*Reason for interruption*
.....	.....	.....
.....	.....	.....
.....	.....	.....

Conformément à l'article 3 de la « Charte Inria Égalité des chances », vous pouvez, si vous le souhaitez, préciser vos éventuelles périodes d'interruption de carrière.
In accordance with Article 3 of the "Inria Charter Equal Opportunities", you may, if desired, specify your possible career break periods.

https://www.inria.fr/sites/default/files/2019-12/13878.pdf
https://www.inria.fr/sites/default/files/2020-01/charter%20GEO.pdf

[2]Indiquez avec précision chaque situation statutaire. Par exemple : pour une situation d'agent titulaire de la fonction publique, précisez le corps et le grade de rattachement, pour une situation de salarié(e) du secteur privé ou d'agent non titulaire d'un établissement public, précisez la nature du contrat salarial, etc.
[2]*For each position, indicate grade or rank. For example, for a tenured civil servant position, indicate the branch and rank, for a private sector position or non-tenured position in a public institution, indicate the nature of the work contract, etc.*

3) Encadrement d'étudiants et de jeunes chercheurs / *Supervision of students and early-stage researchers*

4) Encadrement de développements technologiques (logiciel, matériel, robotique) / *Supervision of technological development (software, hardware, robotics)*

5) Responsabilités collectives / *Responsibilities*

Comité d'organisation *Undone Computer Science* 2024 https://undonecs.sciencesconf.org/ – Financement de 2000€ obtenu auprès du LabEx MILyon pour ce workshop

Comité de programme Workshop *Linearity & Trends in Linear Logic and Applications* 2022 https://lipn.univ-paris13.fr/TLLA/2022/

Relecteur externe pour des conférences (sous réserve d'*open access* : j'ai signé https://nofreeviewnoreview.org) :

- *Computer Science Logic* (CSL) chaque édition de 2021 à 2024
- *Mathematical Foundations of Computer Science* (MFCS) 2022
- *Formal Structures for Computation and Deduction* (FSCD) 2023
- *Mathematical Foundations of Programming Semantics* (MFPS) 2023
- *Conference on Algebra and Coalgebra in Computer Science* (CALCO) 2023
- *Foundations of Software Science and Computation Structures* (FoSSaCS) 2024

Adminstration Représentant des non-permanent·e·s au conseil du Laboratoire d'Informatique de Paris Nord pendant deux années de thèse (où j'ai joué un rôle assez passif)

6) Management (si pertinent) / *Management (if relevant)*

7) Mobilité (si pertinent) / *Mobility (if relevant)*

Je n'ai effectué aucun séjour long à l'étranger, mais j'ai donné des exposés en séminaire d'équipe à Oxford, Bruxelles (ULB), Varsovie (MIMUW), Birmingham, Rome (Roma Tre), Kyoto (RIMS), souvent accompagnés de visites d'1 à 3 semaines. En particulier, mon article avec Mikołaj Bojańczyk [5] à ICALP 2023 est le résultat direct de mon travail à Varsovie.
Concernant la mobilité thématique : mes premiers travaux concernaient l'application de l'algorithmique des graphes à la théorie de la démonstration, tandis que je me suis spécialisé plus tard à l'interface entre λ-calcul et théorie des automates.

8) Enseignement (si pertinent) / *Teaching (if relevant)*

- De novembre à décembre 2023, j'ai donné 20h de cours magistral en M2 recherche à l'ÉNS Lyon, sur la théorie des catégories pour l'informatique : https://nguyentito.eu/categories.html
- En septembre 2023, je suis intervenu à l'EPITA Lyon (école d'ingénieur privée) pour une semaine intensive de cours – 6h de CM+TD+TP par jour – sur la théorie des langages rationnels (niveau L2).
- J'ai effectué un monitorat de 64h/an pendant 3 ans à l'Université Paris XIII durant ma thèse (TDs/TPs pour les cours suivants : Architecture et systèmes (L2) ; Structures de données et algorithmes (L2) ; Fondements de la programmation (M1) ; Web sémantique (L3) ; Programmation fonctionnelle (L2)).
- J'ai été vacataire à l'Université Paris Descartes (désormais fusionnée dans l'Université Paris Cité) en 2016, et à l'Université Rennes 1 en 2021.

9) Diffusion de l'information scientifique (si pertinent) / *Dissemination of scientific knowledge (if relevant)*

Articles pour le magazine *Tangente* co-écrits avec Jérémy Ledent, parus dans la version livre du hors-série 55 "Démontrer : l'art de convaincre" (2015) :

- "Une preuve de maths est un programme informatique !"
- "La théorie des types homotopique : vers de nouveaux fondements des mathématiques ?"

10) Visibilité (si pertinent) / *Visibility (if relevant)*

Exposés invités (dont je reparlerai avec plus de contexte dans les fiches de contributions) :

- Colloque en l'honneur des 60 ans de Christian Retoré (théorie de la démonstration) en 2023
  https://sites.google.com/view/workshop-cr60/program
- Journées nationales 2021 du groupe de travail Scalp du GDR-IM (λ-calcul)
  https://www.irif.fr/gt-scalp/journees-2021
- Workshop international *Trends in Transformations 2021* (automates) rattaché à FSTTCS
  https://www.cse.iitb.ac.in/~krishnas/trends2021/main.html

Mes travaux en théorie des automates m'ont également valu d'être invité au séminaire Dagstuhl "Regular Transformations" en 2023 (https://drops.dagstuhl.de/entities/document/10.4230/DagRep.13.5.96).

11) Eléments divers / *Other relevant information*

Indiquez ici les autres informations **importantes** susceptibles d'intéresser le jury.

Mention here other **important** *information which you feel may be relevant to the committee.*

. . . . . .

Formulaire 2 — DESCRIPTION SYNTHÉTIQUE DE L'ACTIVITÉ ANTÉRIEURE

Form 2 — SUMMARY OF YOUR PAST ACTIVITY

Mes recherches portent principalement sur de l'informatique théorique, plus précisément sur des aspects liés à la logique, la vérification formelle ou la sémantique de programmes – c'est-à-dire les sujets abordés dans le Volume B du *Handbook of Theoretical Computer Science*. Deux de mes sujets principaux de recherche sont la *logique linéaire* et les *transducteurs*. Ce sont des objets d'étude classiques de deux courants assez distincts du "Volume B" : respectivement la théorie des langages de programmation et la théorie des automates. Ainsi, mes travaux cherchent en grande partie à *relier entre eux différents pans de l'informatique théorique* (y compris le "Volume A", voir plus loin).

λ-calcul et logique linéaire. Via la *correspondance preuves-programmes* (ou "correspondance de Curry–Howard"), divers systèmes de preuves formelles s'avèrent être isomorphes à des langages de programmation théoriques. C'est par exemple la base de l'assistant de preuve Coq, qui combine programmation et certification dans un même cadre.
La *logique linéaire* est issue de cette correspondance. Côté programmation, les systèmes de types linéaires contrôlent l'usage de la duplication ; des idées semblables interviennent au sein du langage de programmation système Rust pour contrôler des ressources. Notons que Rust s'inspire en grande partie de l'expérience passée des langages fonctionnels fortement typés (Haskell, OCaml, ...) dont les modèles théoriques idéalisés sont des *λ-calculs typés*.
Un de mes thèmes de recherche principaux (Fiche 2) est d'étudier le *pouvoir calculatoire* de λ-calculs à typage linéaire. Et du côté "preuves" de la correspondance preuves-programmes, certains de mes travaux (Fiche 1) portent sur la *théorie de la démonstration structurelle*, étudiant la combinatoire de systèmes de preuves formelles pour des variantes de la logique linéaire.

Transducteurs. La théorie des automates s'intéresse à des modèles de calcul "à états finis", ce qui les rend suffisamment restreints pour qu'on sache les exécuter efficacement. Cela peut présenter un intérêt pour traiter de gros volumes de données. Souvent, ces automates servent à reconnaître des langages ; autrement dit, ils calculent des fonctions à sortie booléenne. On parle de *transducteurs* pour des automates à sortie plus riche – typiquement, calculant une fonction des mots vers les mots. Les transducteurs contemporains sont en fait plus expressifs qu'on pourrait s'y attendre ; ainsi, par exemple, la fonction $w_0\#\ldots\#w_n \mapsto (w_0)^n\#\ldots\#(w_n)^n$ (où # est une lettre servant de séparateur, on a donc $aa\#ba\#b \mapsto aaaa\#baba\#bb$), appelée "inner squaring", est calculable par un transducteur à jetons (cf. Fiche 3).
La théorie des transducteurs compare donc les classes de fonctions définies par ces modèles de calcul, par des résultats d'équivalence, d'inclusion, de séparation... ou parfois par des algorithmes pour des problèmes d'appartenance à une sous-classe $\mathcal{C}' \subseteq \mathcal{C}$: "étant donnée une fonction $f \in \mathcal{C}$ en entrée, f est-elle en fait dans $\mathcal{C}'$?". Ces problèmes sont le plus souvent indécidables sur des programmes quelconques, mais on peut espérer trouver des algorithmes pour certains modèles d'automates ou de transducteurs. Ceci s'applique plus généralement à des problèmes qui s'apparentent à de l'analyse statique, et c'est là un intérêt majeur de se restreindre à un cadre "à états finis".

Théorie des catégories. Pour démontrer des liens entre automates et λ-calcul (cf. Fiche 2), **Cécilia Pradic** et moi avons été amenés à utiliser le cadre mathématique unificateur des *catégories*. Plus généralement, les catégories sont apparues dans plusieurs de mes travaux sous deux aspects : théorie catégorique des automates [5], et sémantique catégorique (dénotationnelle) des langages de programmation [4,9].

Liens avec l'informatique théorique "Volume A". Je me suis aussi intéressé (de façon moins centrale) à l'algorithmique, la combinatoire et la théorie de la complexité. Ainsi, j'ai suivi durant ma formation initiale un M2 en recherche opérationnelle. Quelques-uns de mes travaux [1,2,3,9,15] consistent à étudier la complexité de problèmes ou à caractériser des classes de complexité, et certains s'appuient sur une connaissance de la littérature en algorithmique des graphes (Fiche 1). Enfin, je participe à (et ai été employé comme post-doctorant sur) un projet ANR se situant à l'interface entre λ-calcul et combinatoire (https://www.lix.polytechnique.fr/LambdaComb/).

Considérations sur mes contributions personnelles. Je considère qu'il est difficile de départager les apports des différents coauteurs d'un travail théorique, et qu'une approximation raisonnable est de considérer que chacun a contribué à part égale. Pour ma part, j'estime avoir contribué substantiellement, dans chaque article que j'ai signé, à la fois aux idées – aussi bien concernant les détails techniques que la vision d'ensemble, malgré les différences d'ancienneté avec certains coauteurs – et à la rédaction. En particulier, durant mon doctorat, j'ai pleinement contribué à élaborer le programme de recherche des "automates implicites" (Fiche 2). C'est pourquoi **je m'autorise à omettre la section "contribution personnelle du candidat" dans chacune des fiches qui suit**.

Formulaire 3 — CONTRIBUTIONS MAJEURES
Form 3 — MAJOR CONTRIBUTIONS

Fiche 1 : Combinatoire des réseaux de preuve

1. Description de la contribution / *Description of the contribution*

L'un des premiers apports de la logique mathématique a été de voir les démonstrations elles-mêmes comme des objets mathématiques : naïvement, une liste de formules logiques dont chacune découle des précédentes. Il s'est vite révélé avantageux de concevoir des formalismes de preuve arborescents (calcul des séquents, déduction naturelle, ...) aux propriétés structurelles plus riches. La logique linéaire va plus loin en proposant de représenter les preuves comme des sortes de *graphes* : les *réseaux de preuve*. Ceux-ci présentent de nombreux avantages, mais posent quelques nouveaux défis techniques ; notamment, vérifier la correction d'une preuve devient un problème combinatoire subtil. La communauté de logique linéaire a obtenu des résultats non triviaux en développant une théorie *ad hoc* des réseaux de preuve.
La seule tentative de relier ces derniers à la théorie des graphes "mainstream" était le travail de Christian Retoré dans les années 1990 réduisant la correction des réseaux de preuve de la logique MLL+Mix – un sous-système simple et couramment étudié de la logique linéaire – à un problème classique : vérifier qu'un couplage parfait donné dans un graphe est unique. J'ai commencé par approfondir les liens entre réseaux de preuve pour MLL+Mix et couplages parfaits, et j'en ai déduit quelques résultats concernant la complexité algorithmique de problèmes sur les réseaux de preuve et leurs propriétés structurelles [3]. J'ai également remarqué dans [3] qu'on peut reformuler une propriété sur les "dépendances logiques" dans les réseaux de preuve en théorème équivalent sur les couplages, susceptible d'intéresser un public plus large (dont une preuve self-contained est donnée dans [20]).
Une conséquence inattendue – en fait la plus importante – de ce travail concerne son prolongement à la *logique pomset* qu'on doit également à Retoré. Cette logique étend MLL+Mix avec un connecteur non-commutatif intermédiaire entre "et" et "ou". Elle se formule facilement en réseaux de preuve, mais pas en calcul des séquents. Cela a motivé Alessio Guglielmi à introduire l'*inférence profonde*[2] – un paradigme où les preuves peuvent être vérifiées facilement mais qui permet plus de liberté que les séquents – et à s'en servir comme base du *système BV*. Ce dernier étend aussi MLL+Mix avec un connecteur non-commutatif, et l'équivalence entre BV et la logique pomset était même conjecturée depuis 2 décennies. **Lutz Straßburger** et moi avons *réfuté cette conjecture* [2]. Notre point de départ a été que les outils de théorie des graphes susmentionnés permettent de montrer que la correction des réseaux pomset est coNP-complète, et que la prouvabilité en logique pomset est Σ_2^p-complète. Ce serait strictement plus difficile que la prouvabilité en système BV – qui est "seulement" NP-complète – si l'on suppose NP $\neq$ coNP. Cela donnait dans un premier temps une réfutation conditionnelle ; nous avons ensuite trouvé un contre-exemple explicite. L'article [2] contient quelques autres résultats autour de ces deux logiques.

3. Originalité et difficulté / *Originality and difficulty*

Mes premiers résultats sur les réseaux de preuve ont, lors de leur présentation en 2017 (avant ma thèse) au workshop Trends in Linear Logic and Applications, surpris la communauté qui estimait que le sujet des réseaux MLL était à ce stade essentiellement épuisé. En particulier, j'ai mis en évidence des différences subtiles entre MLL sans Mix et MLL+Mix qui étaient mal comprises. Quant au résultat pomset logic $\neq$ BV obtenu, il contredit une conjecture à laquelle croyaient les experts en inférence profonde (y compris Lutz Straßburger avant notre collaboration). L'originalité de l'approche a été de s'appuyer intensivement sur la littérature en algorithmique des graphes comme source d'inspiration, là où beaucoup des travaux antérieurs sur les réseaux de preuve redéveloppent eux-même la combinatoire dont ils ont besoin.

4. Validation et impact / *Validation and impact*

Ce travail m'a valu d'être le seul non-permanent à donner un exposé invité au colloque en l'honneur des 60 ans de Christian Retoré.[3] (Mon article court [12] avec **Thomas Seiller** est également lié à des travaux de Retoré.) Aleks Kissinger et Will Simmons ont d'ores et déjà[4] repris notre contre-exemple explicite séparant logique pomset et BV, auquel ils donnent une interprétation "physique" dans le cadre d'une modélisation de la causalité dans des processus d'ordre supérieur.

5. Diffusion / *Dissemination*

Deux publications en conférence [10,6], toutes deux suivies de versions longues [3,2] invitées aux numéros spéciaux de revue dédiés aux meilleurs articles des conférences respectives.

[2] *Deep inference*, un sujet de recherche actif, voir http://alessio.guglielmi.name/res/cos/
[3] https://sites.google.com/view/workshop-cr60/program
[4] Travail non publié à ce stade, mais présenté en séminaire : https://chocola.ens-lyon.fr/events/meeting-2023-09-28/talks/simmons/

Fiche 2 : De la complexité implicite au programme de recherche des automates implicites

1. Description de la contribution / *Description of the contribution*

Complexité implicite. La correspondance preuves-programmes implique souvent des langages de programmation qui garantissent que les programmes *terminent* car cela implique, côté preuves, la cohérence logique. Ainsi, le λ-calcul simplement typé ajoute par-dessus du λ-calcul pur (exemple historique de langage Turing-complet) un système de typage ; on peut alors exclure la possibilité de non-terminaison en ne gardant que les programmes "bien typés". Une telle garantie de terminaison peut même venir avec des bornes de complexité algorithmique, d'où le domaine appelé *complexité implicite*, qui cherche à caractériser des classes de complexité en employant des langages "de haut niveau", sans mentionner explicitement des bornes en temps ou en espace. La duplication des données étant une source majeure de complexité algorithmique, son contrôle au moyen de variantes de la logique linéaire a donné lieu à de nombreux travaux de complexité implicite, à commencer par les caractérisation du temps polynomial (P) dans les années 1990 par J.-Y. Girard.
Dans [11], je résous un problème ouvert posé par P. Baillot dans un article[5] de complexité implicite : la caractérisation de P qu'il propose fonctionne-t-elle toujours quand on ôte les types récursifs du langage ? La réponse est non : on obtient à la place les *langages rationnels* (ou réguliers). Mon inspiration centrale était le *théorème de Hillebrand et Kanellakis*[6] : les fonctions des mots vers les booléens[7] définissables en *λ-calcul simplement typé* en utilisant une représentation "naïve" des entrées correspondent exactement aux langages rationnels, c'est-à-dire au pouvoir des *automates finis*.

Automates implicites. Ceci suggère d'établir, via cette représentation naïve (codage de Church), des liens entre le pouvoir expressif de λ-calculs typés "naturels" et de modèles d'automates (une idée semblable existait aussi dans le domaine connexe du *model-checking d'ordre supérieur*). Avec **Cécilia Pradic**, nous avons initié un programme de recherche sur cette piste de la "complexité implicite pour les automates" ; c'est devenu le sujet central de mon manuscrit de thèse [18].
Notre premier résultat [8] concerne les *langages sans étoile*. Ils forment une sous-classe célèbre des langages rationnels, qui admet de nombreuses définitions équivalentes entre elles (expressions régulières sans étoile, logique du premier ordre, etc.). Nous sommes parvenus à la caractériser dans un λ-calcul muni d'un *typage non commutatif* : on peut forcer les arguments d'une fonction à être utilisés dans l'ordre où ils lui sont passés.
En utilisant un système de types linéaire (mais commutatif), nous caractérisons deux classes de fonctions entre mots calculées par des transducteurs. Une de ces deux classes peut être définie par des *streaming string transducers* (SST) restreints à être *sans copie*. Cette condition analogue au typage linéaire porte sur la manipulation des registres constituant la mémoire des SST. Nous traitons également les fonctions régulières entre arbres, en remarquant que la différence entre "sans copie" et des formes de "single use restriction" plus permissives (et moins aisées à décrire) de certains transducteurs d'arbres peut s'exprimer par les connecteurs additifs de la logique linéaire.
Enfin, dans un travail récent (présenté pour l'instant en workshop [14]), **Gabriele Vanoni** et moi avons caractérisé encore d'autres classes de fonctions entre arbres et résolu une question ouverte que Pradic et moi avions posée dans [8].

3. Originalité et difficulté / *Originality and difficulty*

Si l'idée du typage non commutatif remonte aux années 1950,[8] notre résultat dans [8] constitue, à ma connaissance, la première conséquence *calculatoire* non-triviale d'un système de types non-commutatif à être découverte. (En le rendant commutatif, on obtient les langages rationnels au lieu des langages sans étoile.)
Une caractérisation des fonctions régulières d'arbres assez proche de la nôtre a été obtenue de façon indépendante par Gallot, Lemay et Salvati.[9] Mon travail avec Vanoni poursuit en fait leur approche utilisant des transducteurs qui stockent des λ-termes en mémoire. Cela dit, notre second résultat avec Pradic sur les fonctions entre mots, dont je reparlerai dans la Fiche 3, ne rentre pas dans ce cadre.

4. Validation et impact / *Validation and impact*

Malgré un certain succès d'estime,[10] ces travaux n'ont pas été réutilisés par d'autres collègues. Cependant, la plupart des travaux dont je parle dans la Fiche 3 suivante ont été nourris par ce programme de recherche. Les automates implicites ont aussi été une inspiration cruciale pour mes solutions à deux problèmes ouverts [1,15] et pour le travail récent [4].

5. Diffusion / *Dissemination*

Mon travail en complexité implicite [11] est paru en *post-proceedings* de workshop. Mis à part un article publié à ICALP 2020 [8], mes résultats d'automates implicites avec Pradic ont pour seule source *peer-reviewed* mon manuscrit de thèse.

[5] *On the expressivity of elementary linear logic: Characterizing Ptime and an exponential time hierarchy.* Information and Computation, 2015.
[6] *On the Expressive Power of Simply Typed and Let-Polymorphic Lambda Calculi.* LICS 1996.
[7] Subtilité : on autorise une substitution dans le type d'entrée, comme expliqué dans l'introduction de mon manuscrit de thèse [18].
[8] Joachim Lambek. *The Mathematics of Sentence Structure.* American Mathematical Monthly, 1958.
[9] *Linear High-Order Deterministic Tree Transducers with Regular Look-Ahead*, MFCS 2020.
[10] D'où un exposé invité : https://www.irif.fr/gt-scalp/journees-2021

Fiche 3 : Théorie des transducteurs : fonctions (poly)régulières

1. Description de la contribution / *Description of the contribution*

Fonctions régulières et théorie des catégories. Pour prouver nos résultats d'automates implicites, **Cécilia Pradic** et moi (dans mon manuscrit de thèse [18]) montrons qu'une variante des SST sans copie est étroitement liée à une sémantique du λ-calcul linéaire, c'est-à-dire une structure algébrique vers laquelle on peut "compiler" des λ-termes linéaires. Cela revient à exhiber une structure de "catégorie monoïdale close". Nous démontrons aussi quelques résultats visant à montrer que cette notion catégorique a une pertinence pour la théorie des automates qui dépasse le simple lien avec le λ-calcul.
Plus tard, **Mikołaj Bojańczyk** et moi avons donné [5] une caractérisation particulièrement concise des fonctions régulières (celles calculées par les SST sans copies), utilisant des notions basiques de théorie des catégories. Elle est analogue à la reconnaissance des langages rationnels par des morphismes de semigroupes (ou monoïdes).
Enfin, des liens[11] entre une sémantique catégorique de la logique linéaire et les transducteurs finis bidirectionnels – autre caractérisation des fonctions régulières – ont inspiré un travail récent avec Pradic [16] sur des transducteurs "planaires".

Transducteurs à jetons (sans comparaison). Mon travail sur les automates implicites (Fiche 2) avec Pradic faisait apparaître deux définitions naturelles de classes de fonctions sur les mots calculées par des λ-termes à typage linéaires. L'une d'entre elles donne les fonctions régulières, que je viens de mentionner, et qui ont été intensément étudiées depuis une vingtaine d'années. Par contre, à notre connaissance, l'autre n'existait pas encore dans la littérature.
Elle s'est néanmoins révélée naturelle au vu des travaux récents de Bojańczyk et al. sur les *fonctions polyrégulières*. Ces dernières peuvent être par les "transducteurs à jetons" (*pebble transducers*), qui examinent leur entrée avec plusieurs têtes de lecture (appelées jetons) bidirectionnelles. En ôtant à ces machines la capacité à *comparer les positions de leurs jetons* on caractérise en fait les fonctions survenues dans nos recherches sur le λ-calcul linéaire.
Ceci nous a motivé à étudier cette classe des *fonctions polyrégulières sans comparaison* du point de vue de la théorie des automates "pure", dans l'article [7]. Nous y établissons quelques caractérisations alternatives, des propriétés souhaitables (par exemple la clôture par composition), ainsi que des résultats de séparation avec d'autres classes.

Vitesse de croissance et optimisation. Notre démonstration la plus technique dans l'article [7] susmentionné est un théorème de *minimisation des jetons* : si une fonction polyrégulière sans comparaison f vérifie $|f(w)| = O(|w|^k)$, alors elle peut être calculée par un transducteur à k jetons. (La réciproque est une observation immédiate.)
Bojańczyk a plus tard montré[12] que cela ne marche pas pour les polyrégulières générales : pour tout $k \in \{3, 4, \ldots\}$, on peut trouver une fonction polyrégulière à croissance $O(|w|^2)$ nécessitant au moins k jetons. Gaëtan Douéneau-Tabot et moi sommes co-crédités pour la découverte du contre-exemple pour $k = 3$, qui n'est autre que la fonction "inner squaring" que j'ai présentée dans le formulaire "description synthétique". Cette découverte s'est produite au sein d'une conversation par mail incluant également **Sandra Kiefer** et (encore) **Cécilia Pradic**. Dans [17], Kiefer, Pradic et moi démontrons plus rapidement et moins techniquement un résultat légèrement plus fort que celui de Bojańczyk (séparant les *langages de sortie* de fonctions).

3. Originalité et difficulté / *Originality and difficulty*

Le point de vue des automates implicites (Fiche 2) a été décisif : ainsi, c'est grâce à lui que nous avons soupçonné que les fonctions polyrégulières sans comparaison méritaient d'être étudiées, et plus concrètement, qu'elles étaient stables par composition (mais nous en fournissons une preuve *self-contained* sans λ-termes dans [7]). Notre preuve de minimisation des jetons pour les polyrégulières sans comparaison s'inspirait d'un article publié à LICS 2020 prétendant prouver le résultat pour les polyrégulières générales, qui s'est avéré faux ; il a fallu examiner de près cette preuve assez technique pour s'en rendre compte et malgré tout en extraire des idées pour un argument juste dans notre cas. Mon travail récent avec Kiefer et Pradic [17] procède par réduction à de vieux résultats puissants sur une hiérarchie de transducteurs d'arbres (travaux d'Engelfriet et Maneth), littérature que la communauté connaît de loin mais qui me semble sous-exploitée.

4. Validation et impact / *Validation and impact*

Suite à [7], les fonctions polyrégulières sans comparaison ont fait l'objet d'un exposé invité au workshop international *Trends in Transformations 2021*[13] et sont devenues un thème majeur de la thèse de Gaëtan Douéneau-Tabot (récemment soutenue). La réfutation par Bojańczyk de la minimisation des jetons a aussi été en partie motivée par nos travaux.

5. Diffusion / *Dissemination*

Deux articles à ICALP [5,7] et deux soumissions en cours à des journaux [16,17].

[11] Peter Hines. *A Categorical Framework For Finite State Machines.* Mathematical Structures in Computer Science, 2003.
[12] *On the growth rate of polyregular functions.* LICS 2023.
[13] https://www.cse.iitb.ac.in/~krishnas/trends2021/main.html

Formulaire 4 — PROGRAMME DE RECHERCHE
Form 4 — RESEARCH PROGRAM

☒ Je souhaite candidater dans l'équipe-projet, ou les équipes-projets suivante(s) : PARTOUT (Saclay) / LINKS (Lille)

Intitulé du programme de recherche : **Calculabilité par états finis, optimisation et phénomènes d'ordre supérieur**
Title of research program

Polyrégularité = calculabilité *efficace* par états finis ? Tout d'abord, je propose de m'attaquer à des conjectures de la forme suivante, où $\mathcal{C}$ est une classe de fonctions entre mots :

Conjecture 1 (dépendant de $\mathcal{C}$) *Toute fonction à croissance polynomiale dans $\mathcal{C}$ est polyrégulière, c'est-à-dire calculée par un transducteur à jetons.*
De plus, il existe un algorithme prenant en entrée une fonction $f \in \mathcal{C}$ décrite dans un formalisme caractérisant la classe $\mathcal{C}$ (langage de programmation ou modèle de machines ou système logique ou ...) qui détermine si f est à croissance polynomiale et, si elle l'est, renvoie une description de f par un transducteur à jetons qui la calcule.

On peut envisager un intérêt pratique : mettons que $\mathcal{C}$ soit la classe des fonctions définissables dans un langage de programmation assez expressif, laissant beaucoup de liberté pour décrire des transformations de données, mais en restant "à états finis"[14] pour que la conjecture soit plausible. Même si $\mathcal{C}$ contient des fonctions à croissance exponentielle (ou pire), on voudra souvent écrire des programmes "raisonnables" qui ne produisent pas des sorties trop grandes. Si la conjecture s'avérait vraie pour la classe $\mathcal{C}$, alors ces programmes raisonnables pourraient être traduits vers des transducteurs à jetons (qui servent à définir les fonctions polyrégulières). Or on connaît des algorithmes d'évaluation efficaces pour les transducteurs à jetons. Cela donnerait donc une forme de *compilation optimisante*.
Notons un parallèle avec la complexité implicite (cf. Fiche 2) : les recherches dans ce domaine ont inspiré des optimisations dans des compilateurs[15] ainsi que des analyses statiques de complexité de programmes (dans de nombreux travaux). Ces analyses sont correctes mais incomplètes, puisque savoir si un programme tourne en temps polynomial (par exemple) est en général indécidable. Par contre, dans un cadre "à états finis", des problèmes analogues admettent parfois des procédures de décision correctes et complètes à la fois ; c'est souvent le cas pour l'appartenance d'un langage rationnel donné à une sous-classe fixée provenant de la théorie algébrique des automates. À ce propos, signalons que quelques-unes de ces sous-classes – notamment la classe des langages sans étoile évoquée dans la Fiche 2 – ont récemment été mises en lien avec des problématiques d'optimisation bas niveau (instructions SIMD) pour le traitement de fichiers texte.[16]
Des réponses positives pour certaines instances de la Conjecture 1 auraient aussi l'intérêt théorique de témoigner de la canonicité de la classe des fonctions polyrégulières, qui jouerait alors en théorie des transducteurs le rôle que le temps polynomial joue en théorie de la complexité. Ce serait d'autant plus convaincant que les classes $\mathcal{C}$ considérées sont grandes. Je propose de regarder la classe des **fonctions définissables en λ-calcul simplement typé**. Justifier la pertinence de ce choix nécessite un peu de contexte, que je vais désormais rappeler.

Une classe importante de fonctions à états finis entre arbres. La classe de fonctions que j'appelle $\mathcal{E}$ dans l'introduction de ma thèse (et que j'utilise dans [17]), introduite dans les travaux d'Engelfriet et Vogler des années 1980, inclut à ma connaissance toutes les autres sérieusement étudiées dans la littérature sur les transducteurs. Les fonctions $f \in \mathcal{E}$ peuvent être caractérisées comme les composées $f = f_1 \circ \ldots \circ f_k$ ($k \in \mathbb{N}$) de celles calculées par des transducteurs relativement simples (transducteurs "macro" ou cheminants ("tree-waking")).
Ces transducteurs calculent des fonctions entre *arbres*, et non seulement entre mots (les mots pouvant être représentés comme arbres unaires). Cela fait une vraie différence dans le cas de ces machines trop faibles pour parser la sérialisation d'un arbre. La théorie des automates d'arbres a ainsi été appliquée au traitement de documents à structure hiérarchique (un exemple classique : valider un document XML ou JSON par rapport à un schéma) tandis que les transducteurs d'arbres servent à calculer des transformations de tels documents (c'est pourquoi, typiquement, XSLT est cité comme motivation de l'article introduisant les transducteurs d'arbres à jetons).
De plus, la classe $\mathcal{E}$ vérifie une propriété proche de la Conjecture 1, mais pour la croissance *linéaire* plutôt que polynomiale.

[14] En particulier, tous les modèles de transducteurs dont il sera question ici donnent, quand on les restreint à produire un booléen, la classe des *langages rationnels*. En effet, s'agissant des fonctions des mots vers des booléens – c'est-à-dire des "langages" ou "problèmes de décision" – de nombreuses définitions (automates déterministes ou non-déterministes, reconnaissance par semigroupes, ...) convergent vers les langages rationnels, qu'on considère donc comme une notion robuste et canonique de *calculabilité par états finis*.

[15] Voir la thèse de Thomas Rubiano (2017) : https://theses.hal.science/tel-02362912

[16] Je pense à la thèse récemment soutenue de Claire Soyez-Martin. Voir aussi l'article suivant de Gienieczko, Murlak et Paperman à paraître à ASPLOS'24 : https://github.com/V0ldek/rsonpath/blob/main/pdf/supporting-descendants-in-simd-accelerated-jsonpath.pdf

Théorème 2 (Engelfriet, Inaba & Maneth 2021) *Les fonctions à* croissance linéaire ($|f(w)| = O(|w|)$) *dans $\mathcal{E}$ sont exactement les fonctions* régulières *entre arbres. De plus, on a un algorithme qui décide si une fonction dans $\mathcal{E}$ est à croissance linéaire et, le cas échéant, en renvoie une représentation en tant que fonction régulière.*

Il pourrait donc être raisonnable à court terme de s'attaquer à l'instance de la Conjecture 1 elle-même pour $\mathcal{E}$. Un détail : la conjecture parle de fonctions entre mots, et non entre arbres ; mais je m'attends à voir une bonne notion de fonction polyrégulière entre arbres émerger de l'équipe d'automates de Varsovie dans le futur proche.
Je travaille en ce moment avec Paul Gallot et Nathan Lhote sur l'importation d'outils inspirés des fonctions polyrégulières pour reprouver de façon plus élégante le Théorème 2. Ce travail devrait être présenté au séminaire LX à Bordeaux en février 2024.[17]

Transducteurs d'ordre supérieur et λ-calcul. Deux autres modèles de machines caractérisent directement $\mathcal{E}$ sans parler de composition : les transducteurs à piles imbriquées et les "high level tree transducers". Ces derniers stockent en mémoire des *fonctions d'ordre supérieur*. Les transducteurs "macro", dont la composition engendre utilisent en fait des contextes d'arbres qu'on déjà peut voir comme des fonctions "d'ordre 1" très restreintes : le contexte $a(a(\square_1, b), \square_2)$ avec deux trous $\square_1, \square_2$ représente $(x, y) \mapsto a(a(x, b), y)$. Et lorsqu'on compose $k \geqslant 2$ transducteurs "macro", la fonction obtenue peut être calculée directement par un transducteur stockant des fonctions "d'ordre k", c'est-à-dire prenant des arguments eux-même d'ordre au plus $k-1$: ce sont des fonctions sur les fonctions, manipulées comme données. Ainsi, la composition nous amène sur le terrain de la *programmation fonctionnelle*, et donc du *λ-calcul*.
Cela rejoint donc les travaux que j'ai menés durant mon doctorat avec Pradic sur les "automates implicites", et plus récemment avec Vanoni sur les "λ-transducteurs" – voir Fiche 2. Les "high level transducers" peuvent être vus comme des λ-transducteurs utilisant un fragment légèrement restreint du *λ-calcul simplement typé* (STλC) – j'en reparlerai plus loin.
À ce stade, il est naturel de se pencher sur les "λ-transducteurs simplement typés" sans cette légère restriction. Ils sont déjà connus sous le nom de *transducteurs d'ordre supérieur*, mais semblent étonnamment n'apparaître quasiment pas dans la littérature.[18] En fait, on voit facilement qu'ils sont capables de calculer précisément les fonctions entre arbres définissables à l'intérieur du λ-calcul simplement typé (avec des codages de Church[19]) – c'est le point de vue "automates implicites", qui s'avère ici équivalent aux "λ-transducteurs". Ce qu'on sait de la STλC-définissabilité se résume à quelques résultats classiques d'inexpressibilité – par exemple, sur des codages d'entiers, on ne peut pas exprimer la soustraction – qu'on peut en fait retrouver comme corollaires triviaux du théorème de Hillebrand et Kanellakis (cf. Fiche 2).
La proximité avec $\mathcal{E}$ et le lien formel avec les langages rationnels donné par Hillebrand–Kanellakis suggèrent fortement que l'usage du λ-calcul simplement typé pour décrire des fonctions devrait être vu moralement comme une sorte de modèle de calcul "à états finis" – malgré les phénomènes d'ordre supérieur, on reste en quelque sorte à l'intérieur de la théorie des transducteurs. Un autre argument pour cela est l'usage habituel de sémantiques dénotationnelles finitaires pour des analyses statiques de λ-termes simplement typés, de façon analogue à ce qui se passe en théorie des automates – voir par exemple la section "related work" de [4]. C'est ce qui fait qu'on peut espérer avoir des propriétés telles que la Conjecture 1 pour les fonctions STλC-définissables.

Safe vs unsafe : un saut qualitatif potentiel. On peut caractériser la classe $\mathcal{E}$ dans un style "automates implicites" au moyen du λ-calcul *safe* – celui à propos duquel j'ai résolu un problème ouvert dans [1]. En effet, les "high level transducers" qui caractérisent $\mathcal{E}$ reposent sur la notion de "derived type" des années 1980, dont une relecture basée sur la "safety" a été proposée ultérieurement dans le cadre de travaux sur les *schémas récursifs d'ordre supérieur*.
L'idée est la suivante : on a deux formalismes naturels pour décrire des arbres infinis, l'un étant basé sur le λ-calcul simplement typé (étendu avec de la récursivité, d'où le nom de "schémas récursifs") et l'autre étant un modèle d'automates. Ils correspondent presque… mais la version automates est un peu moins expressive. Pour remédier à ce décalage, on peut soit imposer la safety côté λ-calcul pour le brider, soit, dans l'autre sens, rendre les automates plus puissants.
Le modèle d'automates "naturel" susmentionné est celui des automates à piles imbriquées, et son extension pour coller avec les schémas unsafe consiste à lui rajouter une opération de "collapse". Rappelons maintenant que les transducteurs à piles imbriquées caractérisent $\mathcal{E}$. En vertu des analogies fortes qu'on voit apparaître entre transducteurs calculant une fonction entre structures finies, et schémas engendrant une structure infinie, on devrait avoir :

Affirmation 3 *Il existe un modèle de "transducteurs à piles imbriquées avec* collapse*" qui sait calculer précisément les fonctions STλC-définissables.*

Je me suis déjà convaincu que je savais prouver cette affirmation, mais reste à écrire la démonstration rigoureusement… Au fond, l'intérêt de cet énoncé est surtout psychologique : il fournit un argument supplémentaire pour convaincre la

[17] https://lx.labri.fr/#seminar – diapositives à venir à l'adresse https://nguyentito.eu/2024-02-labri.pdf

[18] Avant 2020, la principale mention semble être l'exposé d'Inaba en 2013 que je cite à la fin de la prochaine section. Citons aussi les travaux sur la vérification formelle de programmes fonctionnels, qui utilisent des machines avec quelques points communs avec les transducteurs d'ordre supérieur (Kobayashi, Tabuchi & Unno, POPL 2010 ; Ong & Ramsay, POPL 2011).

[19] Avec la même convention d'entrée-sortie que dans le théorème de Hillebrand et Kanellakis, autorisant une substitution dans le type d'entrée. Sans cette substitution, on obtient une classe bien plus petite qui admet des caractérisations alternatives (travaux de Marek Zaionc).

communauté de théorie des automates que la STλC-définissabilité relève bien de la calculabilité par états finis, puisqu'elle peut être décrite par un modèle de transducteurs.
Un fait qui me semble plus intrinsèquement intéressant est que les schémas safe sont strictement moins expressifs que les schémas unsafe – c'est un théorème de Paweł Parys. Par analogie, on soupçonne donc que :

Conjecture 4 *Il existe une fonction STλC-définissable entre arbres qui n'est pas dans $\mathcal{E}$.*

Pour attaquer cette conjecture, le point de départ évident est d'étudier la preuve de Parys pour les schémas. Le souci, c'est que c'est l'une de ces démonstrations très longues et techniques que quasiment personne ne comprend. À mon avis, c'est à cause de l'utilisation dans cette preuve d'automates à piles imbriquées, dont la combinatoire est complexe.
Dans le cas des schémas unsafe, certains résultats démontrés dans un premier temps à l'aide des automates à collapse ont été reprouvés de façon bien plus simple techniquement et éclairante conceptuellement par des méthodes sémantiques.[20] Idéalement, on souhaiterait obtenir une démonstration sémantique du théorème de Parys, éventuellement en reformulant sémantiquement les idées combinatoires de la preuve existante. Un obstacle à cela est que la safety n'est pas très naturelle d'un point de vue sémantique – en fait, elle a été inventée pour les besoins de la théorie des schémas récursifs, et n'apparaît pas ailleurs. Mais de fortes ressemblances entre la safety et des conditions de "stratification" utilisées pour la complexité implicite en logique linéaire m'amènent à croire (et je pense savoir le prouver pour les fonctions entre arbres finis) que :

Affirmation 5 *La* logique linéaire élémentaire[21] propositionnelle *définit exactement les fonctions entre arbres finis dans $\mathcal{E}$, et les schémas récursifs linéaires élémentaires engendrent exactement les mêmes arbres potentiellement infinis que les schémas safe.*

Comme on comprend bien les sémantiques dénotationnelles de la logique linéaire élémentaire propositionnelle, cela pourrait ouvrir la voie à l'extension des méthodes sémantiques aux schémas safe ainsi qu'aux fonctions dans $\mathcal{E}$, par exemple pour avancer sur la Conjecture 4.
Inversement, un outil théorique dont on dispose dans le cas safe mais pas dans le cas unsafe est la décomposition en suite de transducteurs plus simples – c'est la première caractérisation de $\mathcal{E}$ qui a été présentée ici. (Une décomposition analogue existe pour les schémas safe sous le nom de "hiérarchie de Caucal".) C'est notamment cette caractérisation de $\mathcal{E}$ qui est utilisée pour démontrer le Théorème 2 par récurrence sur la longueur de la décomposition. Il n'est pas clair que les fonctions STλC-définissables admettent un théorème semblable de décomposition : la question a été posée pour les transducteurs d'ordre supérieur dans un exposé de Kazuhiro Inaba à Dagstuhl il y a une décennie[22] et n'a toujours pas reçu de réponse à ce jour. Cela signifie qu'on ne peut pas facilement adapter la preuve du Théorème 2 pour l'étendre de $\mathcal{E}$ aux fonctions STλC-définissables ; et, a fortiori, que la Conjecture 1 pour ces dernières s'annonce difficile.

(Je donne ci-dessous mon projet d'intégration pour une candidature à Inria Saclay. Alternativement, pour le dossier de candidature à Inria Lille (au lieu de Saclay), un paragraphe d'intégration différent est proposé en page suivante. Il me semble que pour chaque candidature, il ne faut inclure que l'un des deux paragraphes dans la soumission.)

Intégration à l'équipe PARTOUT. Je connais déjà bien les membres de l'équipe, puisque j'y ai travaillé comme post-doctorant pendant 6 mois en 2022, et que j'ai collaboré avec **Lutz Straßburger** [2] (cf. Fiche 1) et **Noam Zeilberger** [15].
Un autre de mes collaborateurs, Gabriele Vanoni [14], a eu comme co-encadrant de thèse **Beniamino Accattoli**, qui est également membre de Partout. Le travail mené par Accattoli depuis une dizaine d'années sur les mesures "raisonnables" de complexité en λ-calcul pur est proche dans l'esprit de mon projet de recherche, puisqu'il s'agit de comprendre le pouvoir calculatoire d'un langage de programmation en présence d'ordre supérieur. On observe également une convergence des outils déployés dans nos travaux (la machine abstraite d'interaction intervient dans [14] ; je prévois d'utiliser la machine abstraite de Krivine pour démontrer l'Affirmation 3).
Plus généralement, l'équipe Partout est reconnue pour son expertise concernant la structure fine des langages liés à la correspondance preuves-programmes, ce qui constitue l'un des deux thèmes principaux auquel se rattache mon projet de recherche. La thématique "automates et transducteurs" est certes bien moins représentée, mais des travaux récents de Zeilberger[23] portent sur la théorie catégorique des automates ; il co-organise aussi à ce sujet le séminaire "Categories for Automata and Language Theory" (https://autcat.github.io/). Ce séminaire a lieu à Paris, à l'Institut de Recherche en Informatique Fondamentale (IRIF) ; travailler en région parisienne me permettrait d'interagir régulièrement avec de nombreux experts sur les automates et transducteurs à l'IRIF.

[20] Citons deux exemples de la décennie 2010 : la décidabilité du problème de sélection MSO (par Grellois et Melliès), et un lemme de pompage appliqué à séparer les niveaux de la hiérarchie des schémas unsafe (par Kobayashi). Ce dernier cas utilise un système de types intersection, mais cela revient au même qu'une sémantique dénotationnelle, selon une correspondance classique.

[21] J'ai travaillé sur la logique linéaire élémentaire *du second ordre* – et non propositionnelle – dans les articles [11,9].

[22] Diapositives : https://www.kmonos.net/pub/Presen/dagstuhl.pdf ; page web du séminaire Dagstuhl en question : https://www.dagstuhl.de/en/seminars/seminar-calendar/seminar-details/13192

[23] *Parsing as a lifting problem and the Chomsky-Schützenberger representation theorem*, avec Paul-André Melliès. MFPS 2022.

Intégration à l'équipe LINKS. Mon collaborateur principal concernant le projet présenté ici serait **Sylvain Salvati**, qui a apporté avec Igor Walukiewicz des contributions majeures aux schémas récursifs. De plus, ses travaux récents à l'interface entre λ-calcul et transducteurs d'arbres (thèse de Paul Gallot, co-encadrée par **Aurélien Lemay**) rejoignent, comme expliqué en Fiche 2, les recherches en "automates implicites" que Pradic et moi avons menées.
Charles Paperman s'intéresse aux liens entre théorie algébrique des automates et optimisation bas niveau dont j'ai parlé au début de ce projet de recherche. Enfin, l'usage de spécifications logiques (MSO interprétations) pour les fonctions polyrégulières fait apparaître des liens forts entre vitesse de croissance de ces fonctions, optimisation et reparamétrisation de requêtes en logique monadique du second ordre (MSO), comme on le voit dans un article récent[24] de Mikołaj Bojańczyk. Ce dernier thème des requêtes MSO est assez proche de questions d'énumération étudiées en théorie des bases de données (spécialité de l'équipe Links), par exemple par **Mikaël Monet**.
Enfin, je pourrais également échanger avec d'autres membres du laboratoire CRIStAL (dont fait partie l'équipe Links), notamment **Patrick Baillot** qui est un expert reconnu en complexité implicite basée sur la logique linéaire, et qui a été un de mes examinateurs de thèse.

[24] *On the growth rate of polyregular functions*, LICS 2023 (déjà cité auparavant à propos d'un autre résultat).

Formulaire 5 — LISTE COMPLÈTE DES CONTRIBUTIONS
Form 5 — COMPLETE LIST OF CONTRIBUTIONS

1. Publications caractéristiques/*Representative publications*

En plus des DOI renseignés plus bas, je fournis pour chacune des 3 publications caractéristiques un lien vers une version avec annexes techniques hébergée sur l'archive institutionnelle HAL.

[3] *Unique perfect matchings, forbidden transitions and proof nets for linear logic with Mix*. (cf. Fiche 1)
https://hal.science/hal-04405117

[5] *Algebraic recognition of regular functions*, avec Mikołaj Bojańczyk. (cf. Fiche 3)
https://hal.science/hal-03985883

[8] *Implicit automata in typed λ-calculi I: aperiodicity in a non-commutative logic*, avec Cécilia Pradic. (cf. Fiche 2)
https://hal.science/hal-02476219

Ma sélection vise à couvrir raisonnablement l'ensemble de mes thèmes de recherche – d'où l'indication, pour chacune des publications, de la fiche de contribution en lien avec elle – tout en évitant des articles trop longs (c'est pourquoi j'inclus [3] au lieu de [2] ; un autre avantage de [3] étant aussi que c'est un article écrit seul, dont la version courte [10] est parue avant le début de mon doctorat).

2. Politique de publication/*Publication policy*

Dans les domaines de l'informatique théorique où je travaille, les publications les plus prestigieuses sont les actes des conférences sélectives (par example ICALP dans mon cas). Cependant, je me suis mis récemment à privilégier, quand les délais de publication importent peu, des soumissions d'articles directement en revue [1,16,17]. En effet, je trouve ce mode de publication préférable aussi bien au niveau écologique (évitant l'empreinte carbone des déplacements loin à l'étranger ; j'ai signé à ce propos le manifeste https://tcs4f.org/) que concernant le contrôle de la qualité scientifique (les deadlines strictes des conférences, aussi bien pour les soumissions que les reviews, encouragent le travail bâclé).
Depuis toujours, je n'accepte de publier des articles que dans des actes de conférences et des revues en accès libre (*open access*). Comme dit précédemment, étant signataire de https://nofreeviewnoreview.org/, j'applique également cet engagement au travail de reviewing.
L'ordre des auteurs et autrices est systématiquement alphabétique sur mes articles, suivant l'usage habituel du domaine. C'est pourquoi j'omettrai mon nom pour ne renseigner que ceux de mes coauteurs (le cas échéant). À noter que :

- Camille Noûs est une autrice fictive : https://www.cogitamus.fr/camille.html ;
- Cécilia Pradic utilisait un autre prénom avant 2023.

3. Publications

3.1 Revues internationales/*International journals*

1. *Simply typed convertibility is TOWER-complete even for safe λ-terms* (auteur unique). Article accepté à Logical Methods in Computer Science (LMCS) sous réserve de révisions mineures. https://arxiv.org/abs/2305.12601
2. *A System of Interaction and Structure III: The Complexity of System BV and Pomset Logic*, avec Lutz Straßburger. LMCS 2023. Version significativement étendue de notre article [6] invitée à un numéro spécial dédié à une sélection des meilleurs articles présentés à CSL 2022. https://doi.org/10.46298/lmcs-19(4:25)2023
3. *Unique perfect matchings, forbidden transitions and proof nets for linear logic with Mix* (auteur unique). LMCS 2020. Version étendue de article [10] invitée à un numéro spécial. https://doi.org/10.23638/LMCS-16(1:27)2020

3.2 Conférence internationales avec comité de lecture/*Reviewed international conferences*

Suivant les recommandations officielles pour le dossier, je mets en gras les publications au *International Colloquium on Automata, Languages and Programming* (ICALP) : elles apparaissent toutes dans le Track B, et ce track est considéré comme la "meilleure" conférence du domaine (ex-æquo avec *Logic in Computer Science* (LICS) qui a cependant le défaut de ne pas être open access, contrairement à ICALP). Cependant, je rappelle à toutes fins utiles qu'Inria est signataire de la *Declaration on Research Assessment*[25] qui insiste sur "the need to assess research on its own merits rather than on the basis of the journal in which the research is published".

[25] https://sfdora.org/read/ | https://www.inria.fr/fr/inria-signataire-de-la-declaration-de-san-francisco

4. *Syntactically and semantically regular languages of λ-terms coincide through logical relations*, avec Vincent Moreau. Computer Science Logic (CSL) 2024, à paraître. https://arxiv.org/abs/2308.00198

5. **Algebraic recognition of regular functions**, avec Mikołaj Bojańczyk. ICALP 2023. https://dx.doi.org/10.4230/LIPIcs.ICALP.2023.117

6. *BV and Pomset Logic are not the same*, avec Lutz Straßburger. CSL 2022. https://doi.org/10.4230/LIPIcs.CSL.2022.32

7. **Comparison-free polyregular functions**, avec Camille Noûs et Cécilia Pradic. ICALP 2021. https://doi.org/10.4230/LIPIcs.ICALP.2021.139

8. **Implicit automata in typed λ-calculi I: aperiodicity in a non-commutative logic**, avec Cécilia Pradic. ICALP 2020. https://doi.org/10.4230/LIPIcs.ICALP.2020.135

9. **From normal functors to logarithmic space queries**, avec Cécilia Pradic. ICALP 2019. https://doi.org/10.4230/LIPIcs.ICALP.2019.123

10. *Unique perfect matchings and proof nets* (auteur unique). Formal Structures in Computation and Deduction 2018. https://doi.org/10.4230/LIPIcs.FSCD.2018.25

3.3 Ateliers internationaux avec comité de lecture/*Reviewed international workshops*

Je n'inclus ici que les *post-proceedings* ayant bénéficié d'un second tour de relecture par les pairs après la tenue de l'atelier. Les articles présentés dans des workshops sans actes officiels sont renseignés dans "Autres publications internationales".

11. *On the elementary λ-calculus with and without fixed points* (auteur unique). DICE-FOPARA 2019. http://eptcs.web.cse.unsw.edu.au/paper.cgi?DICEFOPARA19.2

12. *Coherent Interaction Graphs*, avec Thomas Seiller. Linearity-TLLA 2018. http://eptcs.web.cse.unsw.edu.au/paper.cgi?LinearityTLLA2018.6

3.4 Livres et chapitres de livre/*Books and book chapters*

3.5 Autres publications internationales (posters, articles courts)/*Other international publications (posters, short papers)*

Comme indiqué auparavant, j'inclus ici des *extended abstracts* soumis en workshop. Je me contente d'une sélection d'abstracts récents, couvrant des travaux qui n'ont pas (encore) été publiés ou soumis en revue/conférence.

13. *Papers have bugs – what is to be done?*, avec Enka Blanchard. Présenté au workshop Undone Computer Science 2024. https://undonecs.sciencesconf.org/data/Undonecs_2024_abstract_30.pdf

14. *Invisible pebbles and the geometry of affine higher-order transducers*, avec Gabriele Vanoni. Présenté au workshop Games for Logic and Programming Languages (GaLoP) 2024. https://nguyentito.eu/tmp/galop24.pdf

15. *On the complexity of normalization for the planar λ-calculus*, avec Anupam Das, Damiano Mazza et Noam Zeilberger. Présenté au workshop Trends in Linear Logic and Applications (TLLA) 2023. https://lipn.univ-paris13.fr/TLLA/2023/abstracts/07-Final.pdf

3.6 Revues nationales/*National journals*

3.7 Conférence nationales avec comité de lecture/*Reviewed national conferences*

3.8 Ateliers nationaux avec comité de lecture/*Reviewed national workshops*

3.9 Rapports de recherche et articles soumis/*Research reports and publications under review*

16. *Two-way automata and transducers with planar behaviours are aperiodic*, avec Camille Noûs et Cécilia Pradic. Soumis à Discrete Mathematics and Theoretical Computer Science. https://arxiv.org/abs/2307.11057

17. *Refutations of pebble minimization via output languages* (avec Sandra Kiefer et Cécilia Pradic). Soumis à Fundamenta Informaticae. https://arxiv.org/abs/2301.09234

18. *Implicit automata in linear logic and categorical transducer theory.* Manuscrit de thèse. https://theses.hal.science/tel-04132636

19. *Implicit automata in typed λ-calculi II: streaming transducers vs categorical semantics*, avec Camille Noûs et Cécilia Pradic. Soumission journal rejetée pour longueur excessive (108 pages), dont le contenu apparaît dans mon manuscrit de thèse. https://arxiv.org/abs/2008.01050

20. *Constrained path-finding and structure from acyclicity* (auteur unique). Note recensant des résultats de théorie des graphes que je n'ai pas trouvés (du moins explicitement) dans la littérature ; non destinée à publication "formelle". https://arxiv.org/abs/1901.07028

4. Développements technologiques : logiciel ou autre réalisation / *Technology development : software or other realization*

J'ai été pendant 5 mois ingénieur de recherche à l'IRISA, travaillant avec David Baelde et Stéphanie Delaune sur Squirrel (https://squirrel-prover.github.io/), un assistant de preuve dédié aux protocoles cryptographiques. Étant donné que je terminais ma thèse au même moment, mes contributions ont été très modestes :

- début d'intégration avec des solveurs SMT pour décharger automatiquement des buts réduisibles à de la logique du premier ordre au sein des preuves ;
- migration du système de build à Dune afin de gérer les dépendances optionnelles.

5. Impact socio-économique et transfert / *Socio-economic impact and transfer*