TITRE DE LA THESE: Vérification des idiomes à base de destructeur dans le langage Rust

Direction de thèse : Rémi Douence (maître-assistant IMT, HDR)

Co-encadrant·es : Guillaume Munch-Maccagnoni (CR Inria)

Laboratoire(s) :

☐ GEPEA ☐ IRISA ☐ Lab-STICC ☐ LATIM
☐ Lego ☐ LEMNA ✓ LS2N ☐ hors Laboratoire

Equipe(s) de recherche : Gallinette

Département(s) IMT Atlantique :

✓ DAPI ☐ DSEE ☐ INFO ☐ ITI ☐ LCI ☐ LUSSI
☐ MEE ☐ MO ☐ OPT ☐ SSG ☐ SRCD ☐ SUBATECH

S'agit-il d'une thèse en cotutelle internationale ?
☐ **Oui** ✓ **Non**
Si oui, organisme avec lequel la cotutelle est envisagée :

Le sujet proposé présente-il un caractère interdisciplinaire ?
☐ **Oui** ✓ **Non**
Si oui, expliquer brièvement pourquoi (2 ou 3 lignes) :

La source du co-financement est-elle identifiée ?
✓ **Oui** ☐ **Non**
Si oui, préciser quel co-financement est envisagé : financement du centre Inria Rennes

Autres informations :
Informations utiles que vous souhaiteriez communiquer (si pertinent) :

Cette thèse permettra d'augmenter la cohésion dans l'équipe Gallinette, en rapprochant les parties IMT Atlantique et Nantes Université de l'équipe, grâce au co-encadrement entre Rémi Douence et Guillaume Munch-Maccagnoni. Une précédente collaboration à la faveur d'un étudiant de stage avait permis de déboucher sur une publication.

Un candidat possédant les compétences demandées plus bas a déjà été identifié, il s'agit de Sidney Congard (CV joint).

Contexte ou état de l’art scientifique :
Décrire en 5 à 10 lignes le contexte de la thèse.

Le langage de programmation Rust a été développé comme une alternative à C++ dans le contexte de la parallélisation de larges bases de code comme les navigateurs Internet, et implique aujourd'hui des industriels comme Amazon et Microsoft. Il représente une nouvelle classe de langages de programmation promettant expressivité, performance et sûreté. Les destructeurs, essentiels pour assurer l’efficacité de la gestion des ressources, la sûreté des exceptions et l'intégrité des programmes Rust, n'ont pas été suffisamment étudiés, notamment dans le cadres des efforts existant de vérification formelle des programmes Rust. Des travaux récents montrent qu'il est possible de reconstruire certains aspects des destructeurs dans le cadre de la logique linéaire. Cela permet d'améliorer les techniques de vérification des programmes Rust en incorporant ces concepts, tout en contribuant à l’évolution future des langages de cette classe.

Objectifs de la thèse :
Décrire en 10 à 15 lignes les résultats attendus.

Le but de la thèse est d'étudier comment étendre l'approche de vérification de Rust basée sur les traductions fonctionnelles (Ho et Protzenko, 2022), en utilisant les idées du modèle des destructeurs issu de la de logique linéaire (Combette et Munch-Maccagnoni, 2018). Les objectifs incluent:

1. Développer la notion de traduction fonctionnelle issue du modèle en question, et la comparer à celle utilisée dans l'approche de vérification de Rust.
2. Développer une extension du cadre conceptuel de vérification de Rust donné par Ho et Protzenko (2022) pour prendre en compte les destructeurs et leurs interactions avec les exceptions et autres fonctionnalités du langage.
3. Étendre l'implémentation de l'outil de vérification existant pour incorporer les concepts de destructeurs, en traitant notamment les points délicats connus à ce sujet dans Rust.
4. Évaluer l'impact de ces modifications sur la vérification des programmes Rust, notamment en ce qui concerne la vérification des idiomes basés sur les typestates (ou protocoles), et des programmes utilisant les fonctionnalités de tolérance aux pannes (unwind-safety).

Ceci améliorera la vérification des programmes utilisant des fonctionnalités essentielles à leur sûreté, et mènera à proposer des améliorations du langage sur des aspects actuellement obscurs.

Compétences attendues du ou de la candidat·e :
Lister les principales compétences nécessaires pour ce sujet de thèse.

1. Capacité à acquérir une connaissance bibliographique approfondie d'un sujet, à travailler en équipe, et à synthétiser les résultats de la recherche et à les communiquer clairement à la fois aux experts et aux non-experts.
2. Connaissance approfondie des langages de programmation, en particulier Rust et ses concepts clés, tels que les destructeurs et la gestion des ressources.
3. Maîtrise de la logique linéaire et des modèles catégoriels associés.
4. Connaissance des méthodes de vérification de programmes, notamment les approches basées sur les traductions fonctionnelles.
5. Compétences en programmation et en développement logiciel pour étendre et améliorer les outils de vérification existants.

Bibliographie

Son Ho and Jonathan Protzenko. 2022. *Aeneas : Rust Verification by Functional Translation.* Proc. ACM Program. Lang. 6, ICFP, Article 116 (aug 2022), 31 pages. https://doi.org/10.1145/3547647

Guillaume Combette and Guillaume Munch-Maccagnoni. 2018. *A resource modality for RAII*. Technical Report. INRIA. https://hal.inria.fr/hal-01806634