Colloquium d'informatique de Sorbonne Université

Masterclass avec Sandrine Blazy
mardi 26 novembre 2024
16:00-17:00
Sorbonne Université, LIP6, Campus Jussieu, salle 24-25/405

Mamy Razafintsialonina, doctorant au CEA LIST et au LIP6 (équipe APR)
Titre : Réutilisation de caches et d'invariants pour une analyse statique incrémentale correcte et efficace
Résumé :
Mes travaux présentent une approche pour améliorer l'efficacité en temps de calcul de l'analyse statique de programmes C par interprétation abstraite fondée sur l'incrémentalité et appliquée au greffon EVA de la plateforme Frama-C.
Notre approche comprend deux techniques. La première est le résumé de fonction qui permet de réutiliser les résultats d'analyses sauvegardés pour des fonctions avec des entrées similaires, réduisant ainsi le temps d'analyse lorsque les changements apportés sont mineurs ou locaux. La seconde est la réutilisation des invariants de boucle, qui permet d'accélérer l'analyse des boucles en commençant les itérations à partir des invariants précédemment inférés, évitant ainsi des itérations abstraites.
Nous avons formalisé les techniques mentionnées, dont une partie de la sémantique concrète avec Coq. Les résultats montrent une réduction significative du temps d'analyse sans affecter la consommation de mémoire ni la précision.

Milla Valnet, doctorante au LIP6 (équipe APR)
*Titre :*Analyse statique de valeurs par interprétation abstraite de langages fonctionnels
Résumé :
Cette thèse a pour but de développer une analyse de valeurs par interprétation abstraite des langages fonctionnels.
Nos premiers résultats, basés sur des domaines abstraits relationnels et des résumés des champs récursifs des types algébriques, permettent d'analyser des fonctions récursives d'ordre supérieur manipulant des types algébriques récursifs et d'inférer dans un domaine abstrait leur relation entrée-sortie. Ils adaptent des méthodes de partitionnement pour l'ordre supérieur, de sorte à abstraire les fonctions comme des résumés disjonctifs. Ces méthodes sont implémentées avec succès sur la plateforme d'analyse multi-langages MOPSA pour le langage OCaml.
Pour la suite, nous souhaiterions développer des domaines abstraits pour prouver automatiquement des propriétés plus fonctionnelles sur les programmes (tri de liste, profondeur de structures de données, arbre équilibrés).

Naïm Moussaoui Remil, doctorant à l'École normale supérieure (équipe Antique)
*Titre :*Analyse statique par interprétation abstraite de propriétés temporelles robustes des programmes
Résumé :
L'étude de la robustesse des propriétés de programmes consiste à se poser la question : « Est ce que le comportement d'un programme peut être décidé par (une partie de) ses canaux externes ». Dans notre contexte, les canaux externes seront les *inputs* lus par le programme ainsi que ses variables d'entrée. Mon exposé sera composé de deux parties, chacune présentant une analyse statique (par interprétation abstraite) vérifiant la robustesse de propriétés CTL (in)-désirable de programme vis à vis des *inputs* et des variables d'entrée du programme.
La première analyse est une inférence d'ensembles minimaux de variables à contrôler en entrée d'un programme afin d'assurer une propriété CTL de programme. Notre analyse retourne, pour chacun de ces ensembles, une condition suffisante permettant d'assurer la propriété CTL. Dans un contexte de sécurité, où la propriété CTL exprime une propriété indésirable, notre analyse retourne les vecteurs d'attaques minimaux pour exploiter un programme.
Notre deuxième analyse statique nous permet de vérifier la résilience de la terminaison d'un programme. La terminaison de programme est dite résiliente si, pour toute séquence de valeurs qu'un programme peut lire via ses *inputs*, il existe une exécution ayant lu cette séquence et qui termine. Par négation, nous obtenons la non-terminaison robuste : il existe une séquence *d'inputs*, telle que toute exécution ayant lu cette séquence diverge.
La notion de robustesse d'une propriété de programme est de plus en plus étudiée de façon locale avec une notion de robustesse propre à chaque article. Il semblerait qu'un *framework* plus général manque afin d'exprimer les différentes notions de robustesses. Ainsi, je terminerai cet exposé sur une discussion / un commentaire sur ce problème.