Proposition de stage niveau L3 : génération d'assertions exécutables symboliquement pour un langage de spécification depuis C vers WebAssembly

Léo Andrès, Pierre Chambart, Arthur Carcano

1 Modalités

Lieu du stage Le stage aura lieu à l'adresse suivante :
Équipe WebAssembly
OCamlPro SAS
21 rue de Châtillon
75014 Paris

Dates Le stage durera trois mois et aura lieu du 6 juin au 6 septembre 2024.

Encadrants

- Léo Andrès, ingénieur recherche et développement et doctorant - OCamlPro et Université Paris-Saclay, CNRS, ENS Paris-Saclay, Inria, Laboratoire Méthodes Formelles.
- Arthur Carcano (PhD), ingénieur recherche et développement - OCamlPro.
- Pierre Chambart (PhD), CTO et ingénieur recherche et développement sénior - OCamlPro.

2 Contexte

Le contexte de ce stage est celui de l'analyse statique de programmes WebAssembly [3] et plus précisément de l'outil Owi [1] développé chez OCamlPro

dans l'équipe WebAssembly en collaboration avec l'INESC-ID/Instituto Superior Técnico de l'Université de Lisbonne. Avec cet outil, l'utilisateur peut exécuter symboliquement des programmes WebAssembly et obtenir un modèle donnant les valeurs des entrées pour lesquelles le programme lève une erreur ou atteint une violation d'assertion. Il est également possible d'exécuter symboliquement du code d'autres langages tels que Rust ou C que l'outil se chargera de compiler vers WebAssembly.

3 Sujet

Un outil de détection de bogues dans du code C `owi c` est développé dans le cadre d'Owi. Pour l'instant, l'utilisateur peur déclarer des variables symboliques, ajouter des hypothèses et des assertions à travers des fonctions C fournies par l'outil. Ces assertions sont donc écrites et limitées à des expressions exprimables dans le langage C, ce qui peut les rendre laborieuses à écrire. ACSL [2] est un langage de spécification pour le langage C, il permet d'écrire des propriétés fonctionnelles de façon expressive en écrivant par exemple des pré-conditions et des post-conditions de fonctions ou des invariants de boucles. E-ACSL [4] est un langage similaire à ACSL mais qui permet, à partir d'un programme C annoté, de générer un programme C équivalent contenant des assertions exécutables. Cela permet notamment de vérifier que les propriétés attendues sont bien respectées lors de l'exécution du programme, dans un jeu de test ou dans du code en production. Le but de ce stage est tout d'abord d'intégrer E-ACSL à `owi c` afin de permettre d'utiliser le moteur d'exécution symbolique pour du code annoté. On explorera ensuite comment le fait de disposer de variables symboliques peut permettre de générer des nouveaux types d'assertions, cela peut impliquer de faire évoluer le langage d'E-ACSL, voire de se rapprocher d'ACSL si par exemple les conditions sur les quantifications peuvent être relâchées. On pourra également regarder comment les invariants de boucles peuvent être utilisés pour améliorer l'exécution symbolique. Enfin, il pourra être envisagé d'explorer la conception d'un langage de spécification dédié à Wasm, par exemple basé sur le *custom annotations proposal* [5] et la traduction de spécifications C vers des spécifications Wasm.

4 Prérequis

- programmation OCaml
- méthodes formelles
- sémantique opérationnelle
- techniques de compilation

Références

[1] Léo Andrès, Pierre Chambart et Filipe Marques. *Owi*. 2021. url : https://github.com/ocamlpro/owi.

[2] Patrick Baudin et al. « Acsl : Ansi c specification language ». In : *CEA-LIST, Saclay, France, Tech. Rep. v1* 2 (2008).

[3] Andreas Haas et al. « Bringing the web up to speed with WebAssembly ». In : *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*. 2017, p. 185-200.

[4] Julien Signoles, Nikolai Kosmatov et Kostyantyn Vorobyov. « E-ACSL, a Runtime Verification Tool for Safety and Security of C Programs (tool paper). » In : *RV-CuBES*. 2017, p. 164-173.

[5] WebAssembly Community Group participants. *Custom Annotations Proposal for WebAssembly*. 2020. url : https://github.com/WebAssembly/annotations.