Proposition de stage - 2ème année de Master Recherche

Vérification par preuve formelle de propriétés fonctionnelles d'algorithmes de classification

Lieu du stage
Équipe-Projet Inria Toccata
Laboratoire de Recherche en Informatique
Bâtiment 650 « Ada Lovelace »
Université Paris-Sud
91405 Orsay cedex
France
http://toccata.lri.fr

Encadrement
Claude Marché
Tél: +33 1 69 15 66 08
Email : Claude.Marche@inria.fr
Jean-Christophe Filliâtre
Email : Jean-Christophe.Filliatre@lri.fr

Contexte

Why3 (http://why3.lri.fr [4, 5]) est un environnement généraliste pour la preuve de programmes. Il propose un langage de programmation ad-hoc qui permet de spécifier des propriétés complexes du comportement fonctionnel attendu des programmes. Le générateur d'obligations de preuves de Why3 produit des formules logiques (en logique du premier ordre), qui doivent être prouvées valides par des outils de preuve externes, aussi bien des prouveurs automatiques comme Alt-Ergo (http://alt-ergo.lri.fr [3]), CVC4 (http://cvc4.cs.stanford.edu/web/ [1]) ou Z3 (https://github.com/Z3Prover/ [7]), que interactif comme Coq (http://coq.inria.fr/ [2]). La validité de ces formules implique que le programme respecte ses spécifications. Des exemples de programmes spécifiés et prouvés en Why3 se trouvent sur la galerie Web http://toccata.lri.fr/gallery/why3.en.html.

Parcoursup (https://www.parcoursup.fr/) est le système informatique national français utilisé pour l'orientation des nouveaux bacheliers dans les établissements d'enseignement supérieur. Ce système utilise des algorithmes spécifiques pour établir des classements des candidats pour chaque établissement auxquels ils postulent. Ces classements sont produits en fonction des vœux des candidats, du classement de leurs dossiers par les établissements, ceci en tenant compte de contraintes générales sur des taux de boursiers et de taux de non-résidents.

Les propriétés attendues des algorithmes en question sont documentées par un document en français https://framagit.org/parcoursup/algorithmes-de-parcoursup/blob/master/

doc/presentation_algorithmes_parcoursup.pdf. Ces algorithmes sont implantés en langage Java, et leur code source est disponible sous une licence libre et ouverte https://framagit.org/parcoursup/algorithmes-de-parcoursup.

Description du travail de stage

Le travail proposé se décline en deux axes.

1. Le premier axe consiste à étudier la faisabilité d'une preuve formelle que les algorithmes de Parcoursup respectent les propriétés attendues telles que décrites dans le document de référence en francais. La démarche proposée est de commencer par coder dans le langage de Why3 une version de ces algorithmes, puis de formaliser chaque propriété attendue par une spécification formelle, toujours dans le langage de Why3. Enfin, on essaiera d'établir une preuve formelle grâce aux outils de preuve de l'environnement Why3.

2. Le second axe concerne l'étude d'approches permettant de prouver directement le code Java de ces algorithmes, par exemple via des environnements comme Krakatoa [11] ou openJML [6]. Une difficulté qui sera à considérer pour la preuve du code Java est l'existence d'alias de pointeurs qui est connue comme source importante de difficulté pour la preuve de programmes [8]. On recherchera à réutiliser et adapter les approches récentes à base d'ownership et de borrow pour contrôler statiquement les alias [10].

Ce stage ouvre la porte à plusieurs variations et extensions qui pourront être étudiées. On envisagera en particulier de prouver formellement l'algorithme connu dits de mariages stables de Gale et Shapley [9] (https://fr.wikipedia.org/wiki/Probl%C3%A8me_des_mariages_stables).

Prérequis souhaités

Connaissances de niveau Master en logique, théorie de la démonstration et en méthodes de spécification formelle de programmes. Une expérience de programmation en Objective Caml serait un plus apprécié.

References

[1] Clark Barrett, Christopher L. Conway, Morgan Deters, Liana Hadarean, Dejan Jovanović, Tim King, Andrew Reynolds, and Cesare Tinelli. CVC4. In *Proceedings of the 23rd international conference on Computer aided verification*, CAV'11, pages 171–177, Berlin, Heidelberg, 2011. Springer-Verlag.

[2] Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development*. Texts in Theoretical Computer Science. Springer-Verlag, 2004.

[3] François Bobot, Sylvain Conchon, Évelyne Contejean, Mohamed Iguernelala, Stéphane Lescuyer, and Alain Mebsout. The Alt-Ergo automated theorem prover, 2008. http://alt-ergo.lri.fr/.

[4] François Bobot, Jean-Christophe Filliâtre, Claude Marché, and Andrei Paskevich. Why3: Shepherd your herd of provers. In *Boogie 2011: First International Workshop on Intermediate Verification Languages*, pages 53–64, Wrocław, Poland, August 2011. https://hal.inria.fr/hal-00790310.

[5] François Bobot, Jean-Christophe Filliâtre, Claude Marché, and Andrei Paskevich. Let's verify this with Why3. *International Journal on Software Tools for Technology Transfer (STTT)*, 17(6):709–727, 2015. See also http://toccata.lri.fr/gallery/fm2012comp.en.html.

[6] David R. Cok. OpenJML: Software verification for Java 7 using JML, OpenJDK, and Eclipse. In Catherine Dubois, Dimitra Giannakopoulou, and Dominique Méry, editors, *Proceedings 1st Workshop on Formal Integrated Development Environment*, volume 149 of *EPTCS*, pages 79–92, 2014.

[7] Leonardo de Moura and Nikolaj Bjørner. Z3, an efficient SMT solver. In *TACAS*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340. Springer, 2008.

[8] Jean-Christophe Filliâtre, Léon Gondelman, and Andrei Paskevich. A pragmatic type system for deductive verification. Research report, Université Paris Sud, 2016. https://hal.archives-ouvertes.fr/hal-01256434v3.

[9] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[10] Georges-Axel Jaloyan, Claire Dross, Maroua Maalej, Yannick Moy, and Andrei Paskevich. Verification of programs with pointers in SPARK. working paper https://hal.inria.fr/hal-01936105, November 2018.

[11] Claude Marché. The Krakatoa tool for deductive verification of Java programs. Winter School on Object-Oriented Verification, Viinistu, Estonia, January 2009. http://krakatoa.lri.fr/ws/.