Proposition de stage niveau bac+5

Analyse statique pour optimiser un générateur de code

Mots-clés : analyse statique, génération de code, spécification formelle, *runtime assertion checking*

Cadre

Le CEA LIST est un centre de recherche technologique sur les systèmes à logiciel prépondérant qui mène ses recherches en partenariat avec les grands acteurs industriels du nucléaire, de l'automobile, de l'aéronautique, de la défense et du médical pour étudier et développer des solutions innovantes adaptées à leurs besoins. Au sein du CEA LIST, le Laboratoire de Sûreté des Logiciels (LSL), localisé à Saclay (Essonne, 91), développe des outils d'aide à la validation et à la vérification de logiciels et de systèmes matériels/logiciels, tout particulièrement dans le domaine des systèmes embarqués critiques.

L'un des nos outils, nommé *Frama-C* (`http://frama-c.com`), est une plate-forme logicielle facilitant le développement d'outils d'analyses de programmes C. Le stage se déroulera au sein de l'équipe de R&D développant *Frama-C*.

Objectifs

Chaque programme C analysé par *Frama-C* peut être annoté par des spécifications formelles, écrites dans un langage appelé *ACSL* [1]. *Frama-C* offre alors différentes techniques de vérification pour garantir que le programme satisfait sa spécification. Une des techniques a pour but de traduire une sous-classe des annotations *ACSL*– celles dites exécutables – en instructions C intégrées au programme sous analyse [2]. Cette transformation permet d'obtenir un nouveau programme C dont la correction vis-à-vis de sa spécification est vérifiée dynamiquement, pendant son exécution : cette technique est appelée le *runtime assertion checking*.

Une des difficultés principales de cette transformation réside dans la prise en compte du modèle mémoire du langage C afin d'être en mesure de traduire correctement, par exemple, l'expression *ACSL* `\valid(p)` qui permet de spécifier que le pointeur `p` est valide (*i.e.* non nul et accédant à une zone mémoire licite). Ainsi, un accès à un tableau hors limites (e.g. avec un indice trop grand), ou à une zone mémoire allouée dynamiquement et ensuite libérée, serait invalide. Pour ce faire, la transformation instrumente notamment le programme initial pour collecter ses allocations, dé-allocations et initialisations *via* des appels de fonctions vers une bibliothèque C dédiée préalablement développée [3]. Cette instrumentation est néanmoins très invasive. Pour la rendre plus légère et moins couteuse en temps et en mémoire, une analyse statique flot de données a été développée de façon à n'instrumenter que les opérations sur la mémoire réellement requises. Cette analyse demeure néanmoins perfectible.

Le but du stage est de définir une nouvelle analyse plus précise que celle existante, de la formaliser et de prouver sa correction. Il faudra également l'implémenter en *OCaml* dans un greffon *Frama-C*.

Références

[1] P. Baudin, J.-C. Filliâtre, C. Marché, B. Monate, Y. Moy, and V. Prevosto. *ACSL : ANSI/ISO C Specification Language, version 1.7*, 2013. `http://frama-c.com/acsl.html`.

[2] M. Delahaye, N. Kosmatov, and J. Signoles. Common specification language for static and dynamic analysis of C programs. In *Symposium on Applied Computing (SAC'13)*, pages 1230–1235, 2013.

[3] N. Kosmatov, G. Petiot, and J. Signoles. An optimized memory monitoring for runtime assertion checking of C programs. In *International Conference on Runtime Verification (RV 2013)*, volume 8174 of *LNCS*, pages 167–182. Springer, September 2013.

Candidatures

Maîtriser les langages C et *OCaml* est nécessaire pour ce stage. Avoir des connaissances en analyse de programmes est un plus.

Contacts : Julien Signoles et Nikolaï Kosmatov (`prenom.nom@cea.fr`)

Les délais administratifs de recrutement au CEA étant de 2 à 3 mois minimum, merci de prendre contact le plus tôt possible.