Proposition de stage de BAC+4 ou BAC+5 Pro ou Recherche

Vérification d'un protocole de réseau de capteurs à l'aide d'outils de vérification automatique

Cadre du stage :

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), développe les outils d'aide à la validation et à la vérification de logiciels et de systèmes matériels/logiciels.

L'un des nos outils, nommé Frama-C [1], permet d'utiliser l'analyse statique pour calculer les valeurs possibles des variables à chaque point de programme, trouver des menaces d'erreurs à l'exécution, prouver des propriétés du programme, etc. Un autre outil, nommé PathCrawler, permet de générer des cas de test et de les exécuter afin d'activer tous les chemins d'exécution possibles d'un programme C. Un troisième outil, WP, permet la vérification déductive du programme spécifié à l'aide d'un langage d'annotations de la plate-forme Frama-C. Dans le cadre d'un projet collaboratif, le CEA LIST travaille sur l'implémentation et la validation de protocoles de réseaux de capteurs qui seront utilisés dans des domaines critiques (avionique, transport, etc.)

Objectifs du stage :

Ce stage vise à vérifier des logiciels embarqués dans un réseau de capteurs à l'aide des outils de vérification automatique. Dans un premier temps, il s'agira d'identifier et de spécifier des propriétés de sûreté et sécurité de fonctionnement, notamment, liées à la communication et diffusion de messages. Ces propriétés seront ensuite spécifiées dans le langage de spécification de la plate-forme Frama-C, incluant des préconditions, postconditions, assertions, etc. Ensuite, des outils de vérification seront appliqués pour vérifier les propriétés spécifiés et identifier des éventuelles erreurs.

Les algorithmes seront spécifiés et ensuite prouvés à l'aide des greffons de preuve de Frama-C. Un protocole existant sera utilisé comme le point de départ des travaux. Les techniques de test pourront être utilisées pour la validation des parties du code qui ne pourront pas être entièrement prouvées. Ce stage permettra au stagiaire de découvrir divers outils de vérification de logiciels et les technologies utilisées pour assurer la sûreté et la sécurité de logiciels, les appliquer à la vérification d'un cas d'étude réel, et d'acquérir ainsi des compétences de plus en plus demandées par les entreprises. Il existe des possibilités de continuer en thèse au CEA après le stage.

Profil des candidats :

- Des connaissances en génie logiciel, un goût pour les mathématiques et la logique.
- Bonnes connaissances du langage C, notions en protocoles de communication souhaitées.
- Capacité de travail en équipe.

Conditions : stage indemnisé, aide au logement possible, transport CEA en Ile-de-France gratuit.

Encadrement : Nikolay Kosmatov, email : nikolay.kosmatov@cea.fr

(Les délais administratifs au CEA étant assez longs, merci de nous contacter le plus tôt possible.)

Références

[1] Pascal Cuoq, Florent Kirchner, Nikolai Kosmatov, Virgile Prevosto, Julien Signoles, Boris Yakobowski: Frama-C - A Software Analysis Perspective. SEFM 2012: 233-247

Proposition de stage de BAC+5 Pro ou Recherche

Synthèse d'invariants de boucles pour un outil de vérification automatique

Cadre du stage :

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 Sûreté des Logiciels (LSL), localisé à Palaiseau (Essonne), développe les outils d'aide à la validation et à la vérification de logiciels et de systèmes matériels/logiciels.

L'un des nos outils, nommé Frama-C (http://frama-c.com), permet d'utiliser l'analyse statique pour calculer les valeurs possibles des variables à chaque point de programme, trouver des menaces d'erreurs à l'exécution, prouver des propriétés du programme, etc. Dans Frama-C, un programme C peut être spécifié, ou annoté, par des propriétés à vérifier (son contrat, ou sa spécification) exprimées dans un langage de spécification formelle. Un des greffons de Frama-C, nommé WP, permet de prouver qu'un programme spécifié respecte sa spécification. En général, la preuve en présence de boucles nécessite des annotations spécifiques, appelées *invariants de boucles*, qui décrivent le comportement de la boucle après un nombre quelconque d'itérations.

Objectifs du stage :

Certaines annotations de boucles nécessaires pour la preuve peuvent être déduites grâce à d'autres analyses. La version actuelle de Frama-C ne permet pas de le faire de manière automatique. Ce stage vise donc à rajouter un module de synthèse d'invariants de boucles. Il pourra explorer différentes pistes et s'appuyer en partie sur des travaux existants [1].

Une première piste consiste à appliquer le greffon d'analyse de valeurs Value afin de calculer des informations sur les valeurs possibles des variables modifiées par les boucles. Il existe un greffon de Frama-C Fusy qui récupère une partie de ces d'informations. Ces informations pourront ensuite être utilisées pour produire des invariants (au moins partiels).

D'autre part, des méthodes dynamiques (runtime assertion checking, génération de tests, exécution symbolique à base de solveur de contraintes etc.) peuvent produire des candidats d'invariants à partir d'un nombre fini d'exécutions (voire toutes les exécutions examinées symboliquement). Ces informations pourront être obtenues grâces aux analyseurs existants dans Frama-C.

Enfin, la déduction de candidats d'invariants de boucles basée sur une analyse de patterns [1] à partir de la postcondition sera également explorée.

Ce stage permettra au stagiaire de découvrir diverses méthodes de vérification de programmes, les combiner et d'acquérir ainsi des compétences en vérification de plus en plus demandées par les entreprises. Il existe des possibilités de continuer en thèse au CEA après le stage.

Profil des candidats :

- Très bonnes connaissances en génie logiciel, notamment en vérification déductive.
- Bonne maîtrise du langage OCAML.
- Capacité de travail en équipe.

Conditions : stage indemnisé, aide au logement possible, transport CEA en Ile-de-France gratuit.

Encadrement : Nikolay Kosmatov, François Bobot

Contact : Nikolay Kosmatov, email : nikolay.kosmatov@cea.fr

(Les délais administratifs au CEA étant assez longs, merci de nous contacter le plus tôt possible.)

Référence : [1]M.-V.Aponte et al. *Maximal and Compositional Pattern-Based Loop Invariants,*FM 2012, pp. 37-51.

Proposition de stage Pro niveau bac+4 ou bac+5

Monitoring optimisé de la mémoire lors la vérification à l'exécution des programmes C

Mots-clés : allocation dynamique, validité des pointeurs, vérification des programmes C, 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 Palaiseau (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'analyse 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. Par exemple, 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 en C. Une bibliothèque a été développée [4, 3] pour collecter les allocations, dé-allocations et initialisations effectuées par le programme C et contrôler ensuite la validité (et d'autres propriétés) des accès mémoires.

Ce stage vise à développer une version étendue et optimisée de la bibliothèque de monitoring qui intégrera de nouvelles fonctionnalités pour une meilleure détection de certaines erreurs. Notamment, des initialisations de variables (clause ACSL `\intitialized`) à partir d'autres variables (intitialisées ou non), des tentatives de modifications non-autorisées de variables (clause ACSL `assigns`), des tentatives d'utilisation d'une zone mémoire libérée et ré-allouée à nouveau seront mieux prises en compte. Un deuxième axe des travaux sera l'optimisation en utilisant des techniques de monitoring récentes consistant à surveiller la validité de la mémoire grâce à une copie (*shadow page*) avec des accès fortement optimisés qui pourront améliorer les performances de la bibliothèque [3]. La nouvelle version sera testée sur de nombreux exemples et pourra participer dans une compétition. Ce stage sera l'occasion d'acquérir une bonne expérience de développement pointu en C ainsi qu'une expertise en gestion de la mémoire et détection des anomalies.

Candidatures

Le candidat aura une très bonne maîtrise du langage C, notamment en gestion de la mémoire. Des connaissances en vérification de programmes seraient un plus. Les délais administratifs de recrutement au CEA étant de 2 à 3 mois minimum, merci de prendre contact le plus tôt possible.

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

Références

[1] P. Baudin, J.-C. Filliâtre, C. Marché, B. Monate, Y. Moy, and V. Prevosto. *ACSL : ANSI/ISO C Specification Language*, 2014. `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] A. Jakobsson, N. Kosmatov, and J. Signoles. Fast as a shadow, expressive as a tree : Hybrid memory monitoring for C. In *the 30th ACM/SIGAPP Symposium On Applied Computing (SAC 2015)*, 2015. To appear.

[4] 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)*, pages 167–182, 2013.

Proposition de stage Recherche niveau bac+5

Preuve d'une analyse statique pour 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 Palaiseau (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'analyse 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 [4]. 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, et partiellement formalisée dans [3].

Le but du stage est de finaliser la formalisation de cette analyse et de prouver sa correction. Des extensions et des améliorations de la version actuelle pourront etre apportées pour traiter certaines erreurs de manière plus efficace et pour rendre l'analyse plus précise. La nouvelle version sera implémentée en *OCaml* dans un greffon de *Frama-C*.

Candidatures

Le candidat devra maîtriser les langages C et *OCaml*. Des connaissances en analyse de programmes et en preuve de théorèmes avec un assistant de preuve (e.g. Coq) sera un plus. Les délais administratifs de recrutement au CEA étant de 2 à 3 mois minimum, merci de prendre contact le plus tôt possible.

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

Références

[1] P. Baudin, J.-C. Filliâtre, C. Marché, B. Monate, Y. Moy, and V. Prevosto. *ACSL : ANSI/ISO C Specification Language*, 2014. `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] A. Jakobsson, N. Kosmatov, and J. Signoles. Rester statique pour devenir plus rapide, plus précis et plus mince. In *les 26mes Journés Francophones des Langages Applicatifs (JFLA 2015)*, 2015. To appear.

[4] 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)*, pages 167–182, 2013.

Proposition de stage de master

Génération automatique de code à partir de spécifications formelles

Mots-clés : génération de code, spécification formelle.

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`) et développé en *OCaml*, est une plateforme logicielle facilitant le développement d'analyses de programmes C. Le stage se déroulera au sein de l'équipe de R&D développant *Frama-C*.

Objectifs

Frama-C permet d'annoter formellement un programme C grâce au langage *ACSL* (`http://frama-c.com/acsl`) afin de spécifier le comportement attendu de ce programme. Cette plateforme propose aussi différentes techniques d'analyses pour vérifier que le programme satisfait bien sa spécification *ACSL*.

Lorsqu'un programme n'est que partiellement défini car le code de certaines fonctions (généralement issues de bibliothèques externes) n'est pas connu, il est toujours possible de le vérifier statiquement (*i.e.* sans l'exécuter) en spécifiant en *ACSL* le comportement attendu de ces fonctions.

Néanmoins, reposer uniquement sur ces spécifications en l'absence de code est problématique pour effectuer des vérifications dynamiques (*i.e.* avec exécution du programme sous analyse). Une solution est alors de fournir une implémentation alternative respectant la spécification *ACSL* de chaque fonction manquante, mais ce procédé peut vite devenir fastidieux.

Le but du stage est de développer un nouveau module *Frama-C* permettant de générer automatiquement un code correct des fonctions inconnues à partir de leurs spécifications en *ACSL*. Par exemple, lorsque la spécification indique que la fonction doit retourner 0, il suffit de générer `return 0;`. La génération – est-il besoin de le préciser ? – est cependant loin d'être toujours aussi simple et il conviendra d'identifier le schéma de génération pour le sous-ensemble le plus large possible du langage *ACSL*. Les travaux pourront s'appuyer sur les recherches récentes dans le domaine de la synthèse de fonctions, comme [1].

[1] V. Kuncak, M. Mayer, R. Piskac, P. Suter. Complete Functional Synthesis. Proceedings of the 2010 Programming Language Design and Implementation (PLDI'10). 2010.

Candidatures

Maîtriser les langages *OCaml* et *C* est nécessaire pour ce stage. Posséder des notions en compilation et en spécification formelle est un plus, mais n'est pas indispensable.

Contact : Julien Signoles (*julien.signoles*@cea.fr)

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