Implication de l'équipe MeASI dans le projet EVA-Flo

Compte rendu de la discussion
Secrétaires de séance : Adrien Panhaleux et Nathalie Revol

8 avril 2008

Sujet de la discussion : Matthieu Martel, qui a initié le projet EVA-Flo côté MeASI, est désormais à Perpignan. Comment s'organise la relève ?

Constat : avec le départ de Matthieu et le financement ANR insuffisant pour financer une thèse, manque de main d'œuvre pour produire une interface Fluctuat/LEMA ou une partie du chef d'orchestre.
Idée : regarder ce que donne Fluctuat sur un petit code, comme par exemple une racine carrée (fournie par Claude-Pierre Jeannerod et Guillaume Revy).
Également, participation aux conférences du domaine : SCAN 2008 (El Paso, Texas) et bien sûr SCAN 2010 (Lyon, organisé dans le cadre d'EVA-Flo).

Travail en cours : sur-approximations en utilisant l'arithmétique par intervalles, sous-approximations (cf. exposé de Sylvie), génération de cas les pires (càd d'entrées qui atteignent - presque - les limites des approximations données).

Compétences côté compilation : besoin de discuter sur ce que l'on peut raisonnablement envisager d'automatiser lors de la génération de code.

Mise au point sur les différences entre Fluctuat et Gappa : Gappa est un outil qui fournit des preuves fortes, mais qui fonctionne avec des indications de l'utilisateur éventuellement, et est donc moins automatisé que Fluctuat, alors que Fluctuat a pour but de valider un code mais sans avoir besoin de demander des précisions à l'utilisateur (qui souvent n'est ni celui

qui sait quel problème le code résout, ni celui qui a programmé le code).
L'outil à utiliser dépend donc de l'objectif, du code à analyser et surtout de la taille du code et de qui a construit le code. Par exemple, Fluctuat trouve lui-même un invariant du code et n'a pas besoin de le demander, alors que Gappa permet de prouver formellement qu'un invariant fourni par l'utilisateur est bien un invariant. Ce n'est pas tout à fait vrai, Gappa n'est pas un assistant de preuve et cherche à synthétiser tout seul des invariants, pour ensuite les prouver, mais il peut parfois avoir besoin d'indications extérieures, qui sont exclues en Fluctuat.

Plus précisément, différences entre analyse statique et preuve formelle : l'analyseur statique génére les invariants (annotations selon le vocabulaire donné dans la proposition du projet, ou selon Why), alors qu'un assistant de preuve demande à ce qu'on lui fournisse les invariants. Tout cela même si l'analyse statique et la preuve sont basées sur les mêmes abstractions logiques du calcul numérique en ce qui nous concerne.
L'objectif est de traiter de gros codes (quelques dizaines ou centaines de milliers de lignes de code, voire plus), avec des boucles, pour fournir des informations sur l'erreur, c'est-à-dire quelles sont les sources majeures d'erreur. Bien entendu, ces informations seront plus grossières que celles qui sont le fond de commerce d'Arenaire, telles que "prouver que l'on obtient un arrondi correct". De plus, ces codes sont externes, c'est-à-dire que l'on n'a pas accès au programmeur, et sans annotations. Au contraire, dans EVA-Flo, l'outil qui appliquera des transformations de code sera capable de générer en même temps les annotations.

Existence de problèmes très spécifiques au calcul embarqué : précision variable, calcul en virgule fixe etc. et donc l'automatisation de preuve ou de génération de code dédié aura sûrement des débouchés. Pour la virgule fixe les questions sont de garantir le "range", ou la précision (en essayant du calcul à virgule fixe avec différentes précisions).

En conclusion : l'échange pourra aussi se faire, très concrètement, par des échanges de petits codes pour le calcul embarqué. Par exemple MeASI pourra fournir à Arenaire des codes de filtres numériques, de calculs sur des quaternions (des imitations de codes réels, puisque les codes réels ne peuvent pas être diffusés), ou de petites fonctions telles que le calcul de la racine carrée.