Analyse d'accessibilité pour le langage GAL
A l'aide d'un SAT-solver

Yann Thierry-Mieg, Yann.Thierry-Mieg@lip6.fr

Souheib Baarir, Souheib.Baarir@lip6.fr

Introduction

L'objectif du stage est de réaliser une transformation d'un énoncé d'accessibilité GAL en une formule booléenne. Cette dernière sera exprimée sous une forme normale conjonctive (une conjonction de disjonctions de variables booléennes). La résolution d'une telle formule est réalisée par un SAT-solveur (Minisat, glucose,…). Une technologie qui devient très puissante et permettant de résoudre des problèmes à forte combinatoire.

La réalisation de la transformation permettra, donc, de se comparer aux performances de l'outil existant, utilisant des technologies basées sur les diagrammes de décision (BDD).

Objectifs :

1. Etat de l'art : étude des (nombreuses) approches permettant d'exprimer une relation de transition comme un problème de satisfaisabilité (SAT). Cette étude prendra comme point d'appui les techniques utilisées dans les outils *VIS* ou *LTSA.*
2. Construction de la transformation en appui sur les outils appropriés sélectionnés dans l'état de l'art.
3. Tests sur des benchmarks, mesures de performance, consolidation.
4. Documentation de l'outil (objectif de publication si le niveau atteint le permet)

Exemple de source GAL :

```
reachable : phil_0.state==2 && phil_1.state==2

GAL philo
{
   //variables
   fork[0]=0;
   fork[1]=0;
   phil_0.state=0;
   phil_1.state=0;
   //transitions
   transition t0  [ ( ( phil_0.state == 0 ) && ( fork[0] == 0 ) ) ]
      { phil_0.state = 1;
        fork[0] = 1;    }
```

```
transition t1  [ ( ( phil_0.state == 1 ) && ( fork[1] == 0 ) ) ]
   { phil_0.state = 2;
     fork[1] = 1;    }
```

La cible est une formule booléenne qui soit vraie si et seulement si un état vérifiant (phil_0.state==2 && phil_1.state==2) est accessible. Si c'est le cas, le solver exhibe une affectation des variables qui satisfait le problème. On peut en déduire une trace d'exécution témoin.

Pré-requis et apports du projet :

Le langage d'implémentation reste libre à ce stade, mais une bibliothèque de manipulation des GAL existe déjà en C++ et en Java, donc un de ces langages sera sans doute utilisé. Les bibliothèques SAT sont principalement implémentées en C ou C++.

Il est fortement souhaitable d'avoir suivi des cours introductifs au model-checking (et d'avoir trouvé ça intéressant), mais le bagage nécessaire sera complété au cours du stage (réunion hebdomadaire).

L'outil sera développé en utilisant des outils classiques du développement collaboratif : gestion de versions (svn), serveur d'intégration continue (teamcity), conventions de codage (nommage, commentaires…), configurations de build (maven), métriques de qualité de code (Sonar) etc… déjà mis en place.

Si la réalisation est de bonne qualité, le projet sera intégré dans la plateforme développée au sein de l'équipe Move, et distribué publiquement. Ce stage a vocation à déboucher sur une publication et peut mener à poursuite en thèse.

Le stage se passe dans nos locaux tours 25/26 2éme étage à Jussieu.

Références :

A. Biere et al. Handbook of Satisfiability. Frontiers in Artificial Intelligence and Applications. 2009.

Sachoun et al. SAT Based Verification Tool for Labeled Transition System. Proceedings of the 5th ACIS'2007.

R. Brummayer, and A. Biere. Boolector: An Efficient SMT Solver for Bit-Vectors and Arrays. TACAS'2009.

ITS et GAL : http://ddd.lip6.fr/

Vis : http://vlsi.colorado.edu/~vis/

Minisat : http://minisat.se