Sujet de stage
Model-checking distribué : algorithmes multi-cœurs efficaces

Encadrement : Sami Evangelista, Laure Petrucci
Lieu : LIPN, CNRS UMR 7030
Université Paris 13
99 avenue Jean-Baptiste Clément
93430 VILLETANEUSE
Contacts : {Sami.Evangelista,Laure.Petrucci}@lipn.univ-paris13.fr
Financement : Indemnités de stage

Le model-checking a pour but de vérifier si un système logiciel ou matériel satisfait sa spécification. L'analyse de propriétés de la logique temporelle LTL revient à vérifier que le langage d'un automate de Büchi, produit synchronisé du système et d'un automate représentant la négation de la formule, est vide. La vérification revient alors à la recherche d'un cycle acceptant dans un graphe, c'est-à-dire un cycle contenant au moins un état acceptant dans l'automate de Büchi.

Ce problème a été largement étudié, et des techniques de recherche en profondeur d'abord (DFS — Depth-First Search) permettent de faire la vérification en temps linéaire. Ces techniques se répartissent en deux grandes familles :

- *Nested* DFS (NDFS) [CVWY90] comprend deux procédures : la première recherche des états acceptants, tandis que la seconde, imbriquée dans la première procédure, cherche si ces états font partie d'un cycle acceptant ;
- les algorithmes qui s'appuient sur la recherche de composantes fortement connexes (SCC — Strongly Connected Components) [Cou99, GV04] exploitent la condition nécessaire et suffisante à l'existence d'un contrexemple qu'est l'accessibilité, à partir de l'état initial, d'une composante fortement connexe contenant un état acceptant.

Le problème majeur auquel se heurte la vérification est l'explosion combinatoire de l'espace d'états, tant en espace (mémoire machine limitée) qu'en temps (temps de calcul trop importants). Différentes techniques ont été proposées pour y remédier.

Les gains en mémoire peuvent être obtenus par des codages appropriés des états [Hol95] ou des réductions de l'ensemble des transitions [CGMP99].

La réduction du temps de calcul peut être réalisée au moyen d'algorithmes distribués, pour des architectures à mémoire distribuée [BBC03, BBS01, BCKP01, BCMS04, CP03] ou partagée [BBR07, HB07].

Des travaux récents ont mis en œuvre et prouvé l'efficacité d'algorithmes NDFS pour des architectures multi-cœurs [LLvdP+11, EPY11, LvdP11], MC-NDFS.

Objectifs du stage : L'utilisation de techniques de réduction dans un environnement multi-cœur permettra de repousser les limites de l'analyse à la fois en temps et en mémoire. Il est clair que certaines approches de représentation des états telles que le bitstate hashing [Hol95] sont indépendantes de l'algorithme de recherche et la combinaison devrait être aisée. Par contre l'utilisation d'approches portant sur les transitions, telles que la réduction par ordre partiel [CGMP99] n'est pas triviale.

L'implémentation de ces techniques utilise typiquement deux composants : un mécanisme de sélection indépendant de l'algorithme de recherche et par conséquent compatible avec MC-NDFS ; et un solveur du *ignoring problem* assurant qu'une transition ne sera pas systématiquement ignorée par la fonction de sélection. Ce second composant dépend de l'algorithme de model-checking utilisé.

Le stage étudiera les combinaisons de ces techniques de réduction avec l'algorithme multi-cœurs MC-NDFS, prouvera les algorithmes ainsi obtenus, et les mettra en œuvre au sein de l'environnement LTSMIN (`http://fmt.cs.utwente.nl/tools/ltsmin/`), permettant ainsi des tests comparatifs.

Références

[BBC03] J. Barnat, L. Brim, and J. Chaloupka. Parallel Breadth-First Search LTL Model-Checking. In *ASE'03*, pages 106–115. IEEE Computer Society, 2003.

[BBR07] J. Barnat, L. Brim, and P. Rockai. Scalable Multi-core LTL Model-Checking. In *SPIN'07*, volume 4595 of *LNCS*, pages 187–203. Springer, 2007.

[BBS01] J. Barnat, L. Brim, and J. Stríbrná. Distributed LTL Model-Checking in SPIN. In *SPIN'01*, volume 2057 of *LNCS*, pages 200–216. Springer, 2001.

[BCKP01] L. Brim, I. Cerná, P. Krcál, and R. Pelánek. Distributed LTL Model Checking Based on Negative Cycle Detection. In *FSTTCS'01*, volume 2245 of *LNCS*, pages 96–107. Springer, 2001.

[BCMS04] L. Brim, I. Cerná, P. Moravec, and J. Simsa. Accepting Predecessors Are Better than Back Edges in Distributed LTL Model-Checking. In *FMCAD'04*, volume 3312 of *LNCS*, pages 352–366. Springer, 2004.

[CGMP99] Edmund M. Clarke, Orna Grumberg, Marius Minea, and Doron Peled. State Space Reduction Using Partial Order Techniques. *STTT*, pages 279–287, 1999.

[Cou99] J.-M. Couvreur. On-the-Fly Verification of Linear Temporal Logic. In *FM'1999*, volume 1708 of *LNCS*, pages 253–271. Springer, 1999.

[CP03] I. Cerná and R. Pelánek. Distributed Explicit Fair Cycle Detection (Set Based Approach). In *SPIN'03*, volume 2648 of *LNCS*, pages 49–73. Springer, 2003.

[CVWY90] C. Courcoubetis, M. Y. Vardi, P. Wolper, and M. Yannakakis. Memory Efficient Algorithms for the Verification of Temporal Properties. In *CAV'1990*, volume 531 of *LNCS*, pages 233–242. Springer, 1990.

[EPY11] S. Evangelista, L. Petrucci, and S. Youcef. Parallel nested depth-first searches for LTL model checking. In *Proceedings of the 9th International Symposium on Automated Technology for Verification and Analysis (ATVA11), Taipei, Taiwan*, LNCS. Springer Verlag, October 2011.

[GV04] J. Geldenhuys and A. Valmari. Tarjan's Algorithm Makes On-the-Fly LTL Verification More Efficient. In *TACAS'04*, volume 2988 of *LNCS*, pages 205–219. Springer, 2004.

[HB07] G.J. Holzmann and D. Bosnacki. The Design of a Multi-Core Extension of the Spin Model Checker. *IEEE Trans. on Software Engineering*, 33(10) :659–674, 2007.

[Hol95] G.J. Holzmann. An Analysis of Bistate Hashing. In *PSTV'1995*, pages 301–314, 1995.

[LLvdP+11] A. Laarman, R. Langerak, J. van de Pol, M. Weber, and A. Wijs. Multi-core nested depth-first search. In T. Bultan and P.-A. Hsiung, editors, *ATVA 2011*, LNCS. Springer Verlag, 2011.

[LvdP11] A. Laarman and J. van de Pol. Variations on multi-core nested depth-first search. In *PDMC 2011*, LNCS. Springer Verlag, 2011.