Déterminer le WCET d'applications temps-réel en présence de latences d'exécution variables

Zhenyu Bai, Hugues Cassé, Marianne de Michiel, Thomas Carle, Christine Rochange

Déterminer le WCET d'applications temps-réel en présence de latences d'exécution variables

Zhenyu Bai, Hugues Cassé, Marianne De Michiel,
Thomas Carle, Christine Rochange
CNRS - IRIT - Université de Toulouse,
118 route de Narbonne, Toulouse, 31062

Les contraintes temporelles des systèmes temps-réel doivent être vérifiées afin de garantir une execution correcte. Il faut alors borner le pire temps d'exécution (WCET) des tâches qui les composent. Une difficulté majeure est de prendre en compte la variation du temps d'exécution des instructions. Par exemple, un accès mémoire dans une mémoire cache a une durée de 1 cycle (*hit*) ou de N cycles (*Miss*). Une telle variation est appelée *évènement*.

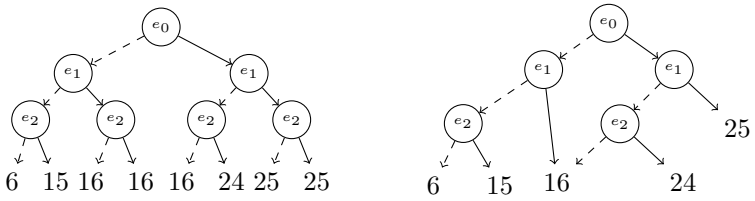
En se basant sur l'approche *Implicit Path Enumeration Technique* [4], le WCET d'un programme est calculé grâce aux WCETs des séquences d'instructions le composant, et grâce aux contraintes appliquées aux nombres d'exécution de ces séquences. De plus, le temps d'exécution d'une séquence d'instructions dépend de la combinaison des évènements de la séquence. En conséquence, le calcul exhaustif de tous les temps possibles ne passe pas à l'échelle. Comme les pipelines modernes sont conçus pour amortir les latences variables, plusieurs configurations d'évènements peuvent mener à la même durée d'exécution. Dans ce travail, nous introduisons une structure de données appelée *eXecution Decision Diagram* (XDD) [2] qui représente les états du pipeline de manière compacte. L'implantation de cette structure dans notre analyse de WCET [5] a permis de réduire l'explosion combinatoire du calcul.

Definition 1 (XDD)**.** Un XDD est défini par :

$$\text{XDD} = \text{LEAF}(k) \mid \text{NODE}(e, \overline{f}, f)$$

avec $k \in \mathbb{Z}$, $e \in \mathcal{E}$ l'ensemble d'évènements impliqués et $\overline{f}, f \in \text{XDD}$.

Un XDD est proche d'un arbre binaire de décision (BDD) [1] i.e. soit un noeud avec deux sous-arbres (un sous-arbre – $\overline{f}$ désignant les cas où l'évènement e est inactif et le cas opposé –f) ; soit une feuille, désignant le temps d'exécution correspondant à la combinaison des évènements le long du chemin de la racine à cette feuille. La figure 1a montre un exemple de XDD complet où le temps d'exécution est 6 si aucun évènement n'est actif, 15 si seulement e_2 est actif, etc. Il existe des sous-arbres identiques dans le XDD, par exemple : $\text{NODE}(e_2, \text{LEAF}(16), \text{LEAF}(16))$.

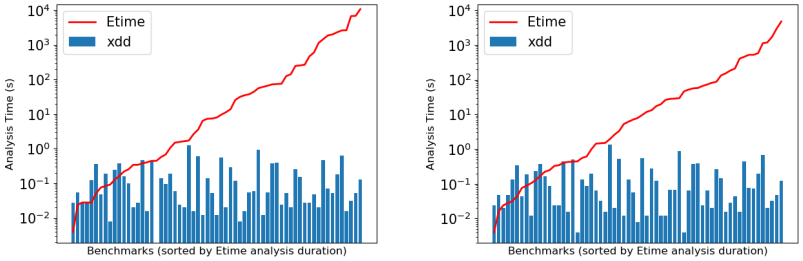


(a) XDD complet (b) XDD compact

Figure 1: Exemple de représentation du temps par XDD.

Pour compacter, la propriété de canonicité des sous-arbres est introduite dans le XDD i.e. les sous arbres identiques sont factorisés, ce qui donne le XDD de la figure 1b.

Nos expérimentations sur le TACLeBench [3] montrent un taux de compactage significatif des XDDs. Une réduction importante du temps d'analyse est également observée sur les deux modèles d'architectures de pipeline présentés dans la figure 2.



(a) architecture simple (b) architecture complexe

Figure 2: Temps d'analyse de XDD vs. Etime (exhaustif).

References

[1] Sheldon B. Akers. Binary decision diagrams. *IEEE Computer Architecture Letters*, 27(06):509–516, 1978.

[2] Zhenyu Bai, Hugues Cassé, Marianne De Michiel, Thomas Carle, and Christine Rochange. Improving the performance of wcet analysis in the presence of variable latencies. In *The 21st ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems*, pages 119–130, 2020.

[3] Heiko Falk, Sebastian Altmeyer, Peter Hellinckx, Björn Lisper, Wolfgang Puffitsch, Christine Rochange, Martin Schoeberl, Rasmus Bo Sorensen, Peter Wägemann, and Simon Wegener. Taclebench: A benchmark collection to support worst-case execution time research. In *16th International Workshop on Worst-Case Execution Time Analysis*, 2016.

[4] Yau-Tsun Steven Li and Sharad Malik. Performance analysis of embedded software using implicit path enumeration. In *Proceedings of the ACM SIGPLAN 1995 workshop on Languages, compilers, & tools for real-time systems*, LCTES '95, pages 88–98, La Jolla, California, USA, November 1995. Association for Computing Machinery.

[5] Christine Rochange and Pascal Sainrat. A Context-Parameterized Model for Static Analysis of Execution Times. In Per Stenström, editor, *Transactions on High-Performance Embedded Architectures and Compilers II*, Lecture Notes in Computer Science, pages 222–241. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.