La prédiction de valeur comme moyen d'augmenter la performance des processeurs superscalaires

Arthur Perais [1]

Accessit du prix de thèse Gilles Kahn 2016

Arthur Perais a soutenu sa thèse [2] en septembre 2015 à Inria Rennes – Bretagne Atlantique, sous la direction d'André Seznec. Après une année en tant qu'ingénieur de recherche au sein de la même équipe (EPI ALF, maintenant PACAP), il a rejoint Qualcomm Datacenter Technologies en janvier 2017.

Tout programme informatique est écrit avec pour finalité d'être exécuté sur un ordinateur, qu'il s'agisse d'un smartphone ou d'un supercalculateur. Cependant, quelle que soit la classe de processeur considérée, la performance est contrainte par les différentes dépendances inhérentes au programme, c'est-à-dire par le graphe de flot de contrôle ainsi que le graphe de flot de données. En particulier, si grand soit le nombre de transistors gravés dans le silicium, et si élevée soit leur fréquence, deux instructions dépendantes devront toujours être exécutées en séquence, comme exprimé par la sémantique purement séquentielle du langage machine. De plus, bien que certains programmes puissent – indirectement –

1. http://people.irisa.fr/Arthur.Perais/

2. *Increasing the performance of superscalar processors through value prediction*, manuscrit consultable à l'adresse https://tel.archives-ouvertes.fr/tel-01282474/

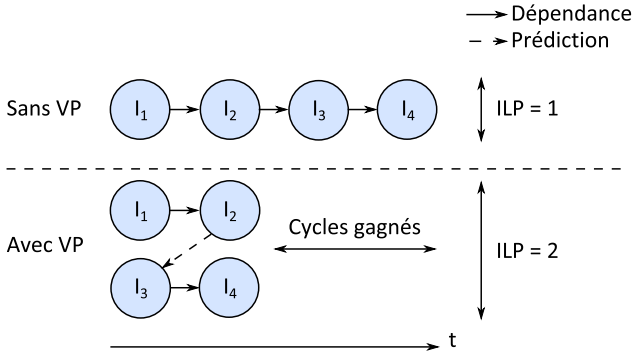

FIGURE 1. Impact de la prédiction de valeurs (VP) sur la chaîne de dépendances séquentielles exprimée par un programme.

outrepasser cette limite de par leur parallélisme, de nombreux programmes restent purement séquentiels. Pour ces programmes, c'est l'augmentation de la performance d'un seul cœur qui réduira le temps d'exécution du programme, et non l'addition de cœurs supplémentaires sur la même puce. Il est donc important pour l'académie comme pour l'industrie de continuer à améliorer la performance séquentielle, même à l'ère des multi- et many-cœurs.

Un moyen d'augmenter la performance séquentielle est de casser les dépendances au niveau du matériel, spéculativement, tout en s'assurant que l'exécution respecte le modèle de programmation séquentiel. Cela permet d'augmenter le nombre d'instructions indépendantes, et donc d'augmenter le nombre d'instructions pouvant être traitées à chaque cycle du processeur.

Un exemple de spéculation, étudié dans cette thèse, consiste à prédire le résultat des instructions lorsqu'elles sont récupérées depuis la mémoire, afin que les instructions suivantes puissent être exécutées plus tôt. Cela revient à casser la dépendance entre deux instructions dans le graphe de flots de données, comme illustré par la Figure 1.

Bien qu'ayant déjà été étudiée en profondeur pendant les années 90 [1, 2, 8], et bien qu'un fort potentiel pour augmenter la performance ait été observé, la prédiction de valeurs (*Value Prediction*, VP) est tombée en désuétude au début des années 2000 car l'industrie comme l'académie considéraient le mécanisme comme trop complexe pour voir le jour dans un processeur.

Cependant, les contributions de cette thèse montrent au contraire qu'implémenter la VP dans un processeur moderne n'est pas rédhibitoire, et qu'elle permet même de réduire la complexité de certaines parties du processeur. En particulier, ces travaux montrent qu'il est possible de déterminer quelles prédictions ont des chances extrêmement fortes d'être correctes. En forçant le processeur à n'utiliser que cette

classe de prédictions, le nombre total de mauvaises prédictions – après lesquelles l'état interne du processeur doit être réparé – devient très faible. De ce fait, il est possible d'implémenter un mécanisme de validation des prédictions (et de réparation des mauvaises prédictions) très simple, qui traite les prédictions dans l'ordre, au retirement [3] [4].

Puisque les mauvaises prédictions sont très rares, le coût élevé – en terme de performance – d'un tel mécanisme importe peu, pour autant, la performance est améliorée de par la présence de la VP. Précédemment, la VP requerrait un mécanisme complexe de ré-exécution des instructions directement depuis l'ordonnanceur, dans le désordre.

De plus, le fait de prédire les résultats permet de modifier l'ordre d'exécution des instructions, de manière à simplifier un des composants les plus complexes du processeur : le moteur d'exécution dans le désordre (*Out-of-Order*, OoO). En effet, une instruction prédite n'a pas besoin d'être exécutée au plus tôt, puisque son résultat est déjà disponible. Il est donc possible d'exécuter les instructions prédites dans l'ordre au retirement [3, 6, 7].

Ces instructions n'entrent pas dans le moteur OoO. De même, certaines instructions ont leurs opérandes prédits, ce qui permet de les exécuter juste après qu'elles soient récupérées depuis la mémoire, dans l'ordre. Ces instructions ne sont pas non plus insérées dans le moteur OoO. De ce fait, le nombre d'instructions entrant dans le moteur OoO est grandement réduit, et la capacité dudit moteur peut donc diminuer. En particulier, nous avons pu réduire le nombre d'instructions traitées par le moteur OoO de 6 à 4 par cycle sans réduire la performance. Étant donné que la complexité (nombre de transistors, délai, dissipation thermique) du moteur OoO augmente de manière quadratique avec le nombre d'instructions traitées à chaque cycle, une telle réduction est un argument fort pour l'ajout de la VP dans les processeurs futurs, et ce malgré l'ajout de matériel pour exécuter certaines instructions dans l'ordre.

Finalement, les processeurs modernes sont capables de récupérer plusieurs instructions depuis la mémoire à chaque cycle. Il faut donc fournir plusieurs prédictions de valeurs à chaque cycle, ce qui pose des problèmes au niveau de l'implémentation, en particulier en présence d'un jeu d'instructions à encodage variable tel que x86. Nous avons proposé une organisation en blocs pour le prédicteur de valeurs : chaque entrée du prédicteur contient n prédictions, qui sont attribuées séquentiellement aux instructions du bloc d'instructions récupéré par le processeur à ce cycle [5]. Cette organisation permet de prédire plusieurs instructions par cycle en effectuant un unique accès au prédicteur, qui peut donc être implémenté avec des mémoires possédant autant de ports d'accès que le cache d'instructions.

3. Le retirement consiste à rendre visible les effets d'une instruction sur l'état de la machine. Il est effectué dans l'ordre par le processeur afin de respecter la sémantique séquentielle du programme.

Réunies, ces trois contributions proposent une implémentation réaliste de la prédiction de valeurs dans un processeur moderne, que j'espère voir un jour implémentée sur une puce. En effet, dans un contexte ou la performance séquentielle est critique pour de nombreux algorithmes mais particulièrement difficile à améliorer, un mécanisme à coût matériel raisonnable tel que celui développé dans mes travaux de thèse est attrayant.

Références

[1] M.H. Lipasti, C.B. Wilkerson, and J.P. Shen. Value locality and load value prediction. *ASPLOS-VII*, 1996.

[2] A. Mendelson and F. Gabbay. Speculative execution based on value prediction. Technical Report TR1080, Technion-Israel Institute of Technology, 1997.

[3] A. Perais and A. Seznec. EOLE : Paving the Way for an Effective Implementation of Value Prediction. In *ACM/IEEE International Symposium on Computer Architecture*, pages 481–492. Minneapolis, US, Jun. 2014.

[4] A. Perais and A. Seznec. Practical Data Value Speculation for Future High-end Processors. In *IEEE International Symposium on High Performance Computer Architecture*, pages 428–439. Orlando, US, Feb. 2014.

[5] A. Perais and A. Seznec. BeBoP : A Cost Effective Predictor Infrastructure for Superscalar Value Prediction. In *IEEE International Symposium on High Performance Computer Architecture*, pages 13–25. San Fancisco, US, Feb. 2015.

[6] A. Perais and A. Seznec. EOLE : Toward a Practical Implementation of Value Prediction. In *IEEE Micro's Top Picks*, volume 35, pages 114–124, May 2015.

[7] A. Perais and A. Seznec. EOLE : Combining Static and Dynamic Scheduling through Value Prediction to Reduce Complexity and Increase Performance. In *ACM Transactions on Computer Systems*, volume 34, pages 4.1–4.33, Apr. 2016.

[8] Y. Sazeides and J.E. Smith. The predictability of data values. In *Proceedings of the International Symposium on Microarchitecture*, pages 248–258, 1997.