Increasing the Performance of Superscalar Processors through Value Prediction

Arthur Perais

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. Cet ordinateur consiste en un ou plusieurs processeurs ainsi que des périphériques (ex., stockage). Tous ces composants influent sur le temps requis l'ordinateur considéré pour exécuter un programme donné, mais le processeur possède une place particulière, car c'est à lui que revient le rôle de suivre l'algorithme implémenté par le programme. Ainsi, sa performance est contrainte par les différentes dépendances inhérentes à l'algorithme, 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 séquentiellement, comme exprimé par la sémantique séquentielle du langage machine.

De ce fait, un moyen d'augmenter la performance, afin de réduire le temps d'exécution du programme, est de casser ces dépendances au niveau du matériel, spéculativement, tout en s'assurant que l'exécution respecte le modèle de programmation purement séquentiel. Cela permet d'augmenter le nombre d'instructions indépendantes, et donc d'augmenter le nombre d'instructions pouvant être traitées à chaque cycle 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. Bien qu'ayant déjà été étudiée en profondeur pendant les années 90, 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. 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 re-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. 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. 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.

Réunies, ces trois contributions proposent une implémentation réaliste de la prédiction de valeurs dans un processeur moderne.