Architecture et performance des interpréteurs

Pascal Rigaux

29 juillet 1998

Les interpréteurs sont de plus en plus utilisés de nos jours. Les avantages des interpréteurs sont nombreux : vitesse de développement, facilité d'utilisation, portabilité, sécurité. Néanmoins leur lenteur est un défaut majeur. Le but de cet exposé est l'étude du comportement des interpréteurs et des possibilités d'améliorer leur performance.

Description des interpréteurs

Quatre interpréteurs sont étudiés : MIPSI est un émulateur binaire pour MIPS R3000, Java un langage orienté objet, Perl un langage de script et Tcl un langage de commande permettant l'interfaçage avec d'autre langages.

Architecture logicielle et performance des interpréteurs

En comparant la perte de performance par rapport au langage C (de 1 à 6500 fois plus lent), on s'aperçoit de l'importance des librairies natives.

Les interpréteurs s'appuient sur une machine virtuelle. MIPSI et Java ont une machine virtuelle et donc un temps de décodage des instructions court. Par contre un programme nécessite un nombre important d'instructions. A l'inverse, Perl et Tcl ont une machine virtuelle complexe et un temps de décodage des instructions long. Ce temps de décodage long est compensé par la complexité des instructions.

La gestion mémoire est très différente sur les quatre interpréteurs. Java et Perl utilise des techniques efficaces. MIPSI simule la mémoire virtuelle et les accès mémoire coûte donc assez cher. Tcl utilise une table des identificateurs sous la forme de texte et chaque accès à une variable coûte cher.

En étudiant des applications réelles, on s'aperçoit des faiblesses potentielles de chaque interpréteur. Le coût du décodage des instructions est important dans MIPSI. La performance de Java repose en grande partie sur les librairies natives. Perl et Tcl n'ont pas de comportement homogène suivant les programmes, les goulots d'étranglement varient.

Comportement des interpréteurs vis-à-vis de l'architecture

Le comportement de l'interpréteur l'emporte sur les spécificités des applications. La performance ne dépend plus des applications. Ainsi un des programmes de test provoque beaucoup de défaut de data-TLB. Réécrit dans un langage interprété, cette caractéristique disparaît.

Les machines virtuelles simple ont une bonne localité dans le cache. Un cache d'instruction de 8 Ko direct-mapped suffit. Au contraire les machines virtuelles complexe possèdent peu de localité et nécessitent un cache de 32 Ko associatif.

Cette étude montre que l'utilisation de matériel spécialisé pour un interpréteur donné n'est pas nécessaire, de meilleurs gains en performance peuvent être obtenu logiciellement en améliorant l'interpréteur et les librairies natives.