Machine virtuelle d'ICFP 2006

Jordi Bertran de Balanda

16/02/2016

Utilisation

Compilation et exécution

Après avoir réuni les fichiers dans un nouveau dossier avec en sus sandmark.umz et éventuellement codex.umz:

```
$ make
$ ./universal-machine <fichier>
```

Fichiers

- universal-machine.c: corps de la VM - lecture des fichiers de code octet, calcul des instructions et exécution.
- tools.c/h: Structures et fonctions de gestion (création, destruction)

Features

Allocation

On choisit la stratégie classique en cas de manque de mémoire pour stocker des plateaux de code, et on multiplie par 2 la mémoire disponible pour la VM en utilisant realloc pour minimiser la complexité.

Freelist

On conserve une freelist pour réallouer des plateaux dans les trous créés par les instructions ABANDON, donc les deux opérations sont push (ajout en tête) et

pop (récupération de la tête de liste), tous deux en O(1). La complexité de cette utilisation ne serait pas énorme s'il n'était pas nécéssaire de malloc à chaque ABANDON, et free à chaque ALLOC. En l'état, c'est probablement la chose qui rajoute le plus d'overhead quand on compare avec des VMs plus performantes comme celle de Darius, avec ~25% de pertes.

Goto calculé

La VM choisit le traitement à effectuer en fonction de l'instruction avec des goto calculés. Ceci induit un gain de 30% par rapport à l'utilisation d'un switch case qui implique d'itérer à travers les options du switch pour trouver la bonne instruction - par exemple, le sandmark s'exécute en 25s avec goto contre 35s avec le switch case.

Debugging

On peut optionnellement décommenter la ligne #define DEBUG 1 pour obtenir un affichage des instructions exécutées et de leurs arguments (très lourd lorsque la machine s'exécute correctement).

Améliorations possibles

Gestion de la fragmentation

On pourrait attaquer la fragmentation causée par les instructions ABANDON en conservant un entier qui pointe vers la permière case vide, et le mettre à jour à chaque ALLOC réussie avec une boucle, ainsi qu'à chaque ABANDON. En comparant avec d'autres VMs, malgré la boucle, il semblerait que ce soit plus efficace en moyenne que de conserver une freelist qu'il faut allouer souvent.