Unité d'enseignement NI135-2013oct du Master SESI

TME-2 : évaluation expérimentale de performances sur la cible ALMOS/TSAR

Objectif

L'objectif de ce TME est d'évaluer expérimentalement l'impacte de la stratégie d?affinité mémoire automatique Auto-Next-Touch.

Auto-Next-Touch

Vous allez comparer les performances de la stratégie Auto-Next-Touch avec deux autres stratégies First-Touch et Interleave. La première est utilisée par défaut dans les noyaux monolithiques existants (p. ex : Linux, Solaris, etc.). La deuxième consiste à distribuer l?allocation des pages physiques, demandées par l?application lors des défauts de pages, sur l?ensemble des noeuds cc-NUMA. Étant donné la faible vitesse du simulateur du processeur many-core TSAR et le temps imparti de cette séance de TME; la comparaison de performances sera effectuée en utilisant : (i) une configuration de TSAR fixée à 4 clusters (16 cores), et (ii) un microbench que vous allez devoir écrire.

Lors de cette expérimentation, nous nous intéressons uniquement à la phase de l?exécution parallèle du microbench. L'indicateur de performance utilisé pour cette comparaison est la scalabilité (le passage à l'échelle en nombre de cores).

Description du microbech à développer

Le fonctionnement général du microbench est comme suit : après avoir analysé les arguments de la ligne de commande, le thread exécutant la fonction *main* du microbench doit allouer et initialiser avec des valeurs aléatoires un espace mémoire de *M* octets (*M* étant un argument de la ligne de commande). Ensuite, il doit lancer autant de threads de traitement (ou workers) qu'il y a de cores physiquement disponibles avant qu?il se synchronise sur leur terminaison. Enfin, il doit afficher le temps d?exécution de la phase parallèle (en nombre de cycles) avant de se terminer. Concernant le traitement parallèle, chaque worker doit calculer la signature MD5 d?une portion de l?espace mémoire alloué et initialisé par le thread initial. Cette portion est de taille *1/N * M* octets (*N* étant le nombre de workers).

Mise en place des trois stratégies

Pour cette expérimentation, nous utilisons une version d'Almix dont la notion de threads et la gestion de l?espace virtuel de processus sont tout à fait similaires à celles de Linux. Cette version d'Almix implémente les trois stratégies First-Touch, Interleave et Auto-Next-Touch. La stratégie Auto-Next-Touch y est appliquée par défaut, mais ce choix peut être désactivé et l?une des deux autres stratégies peut être activée dynamiquement (par une granularité de thread) à deux moments précis : lors de la création d?un thread et lors de la bifurcation d?un processus (appel système *fork*). Sans indication explicite lors de ces deux appels systèmes, la stratégie du thread créateur est héritée par le nouveau thread et elle est conservée lors de la transformation d?un processus (appel système *exec*).

Pour permettre au programmeur de choisir l'une des trois stratégies d'affinité mémoire proposées par Almix pour un nouveau processus, la bibliothèque de threads POSIX (libpthread) d'ALMOS a été étendue en ajoutant la fonction *pthread_attr_setforkinfo_np*. Cette fonction doit être appelée avant la bifurcation d'un nouveau processus. Elle prend en argument un mask représentant une association de plusieurs drapeaux permettant d'exprimer quelle stratégie d'affinité mémoire Almix doit appliquer pour le nouveau processus. En particulier, deux drapeaux sont à utiliser pour cette expérimentation :

- *PT_FORK_SET_AFFINITY* : ce drapeau permet d'activer le contrôle de l'affinité mémoire. Il doit être positionné quelques soit le choix de la stratégie d'affinité mémoire souhaitée.
- *PT_ATTR_MEM_PRIO, PT_ATTR_AUTO_NXTT* : l?association de ces deux drapeaux permet de choisir la stratégie Auto-Next-Touch.
- *PT_ATTR_INTERLEAVE_ALL* : ce drapeau permet de choisir la stratégie Interleave.

Pour pouvoir activer la stratégie First-Touch, il faut désactiver la stratégie par défaut Auto-Next-Touch et fournir à la fonction *pthread_attr_setforkinfo_np* un mask ne désignant aucune stratégie, c'est-à-dire, uniquement le drapeau *PT_FORK_SET_AFFINITY* doit être positionné.

Pour pouvoir appliquer une stratégie différente à chaque lancement du même programme, la commande *exec* du shell d?ALMOS doit être modifiée pour faire en sorte qu'il interprète la valeur d?une variable d?environnement : *ALMIX_MEMORY_AFFINITY*. Selon la valeur de cette variable, le shell informe le noyau Almix du choix de la stratégie à appliquer au nouveau processus.

Compte rendu

Il vous est demandé de rédiger un compte rendu reprenant les points suivants :

- Introduction présentant le contexte de l?expérimentation et précisant la question investiguée.
- Dispositif de l?expérimentation
- Méthodologie mise en place
- Résultats et analyses
- Conclusions
- Annexe contenant le code source commenté de votre microbench

Votre rapport doit être soumis au format .pdf par courriel à l'adresse suivante : ghassan.almaless (at) lip6.fr. La date limite est fixée pour le 23/10/2013 à 8h30. Toute soumission au-delà de cette date limite ne sera pas prise en compte.