TP1 - OSEM

Ilyas Toumlilt
Yassine Aroua

12/11/2015

Introduction :

Ce rapport propose une vérification expérimentale de l'efficacité de la distribution d'une structure de données en répartissant les traitements, dans une architecture Many-Cores à accès mémoire non uniforemes : cc-NUMA.

Cette étude a été menée sur TSAR, un simulateur de processeur many-cores précis au cycle près et au bit, pouvant être configuré jusqu'à 256 processeurs, et sur ALMOS, un système d'exploitation conçu par Ghassan Almaless dans le cadre de sa thèse, spécifiquement pour TSAR et qui repose sur le même paradigme de programmation en mémoire partagée que d'autres systèmes monolithiques tels que Linux ou BSD.

Pour pouvoir mesurer la répartition de traitements respectant la localité des accès mémoire, nous avons décidé, dans le cadre de ce TP, de réaliser une application calculant de manière équitable sur un nombre défini (pour chaque exécution) de processus, la taille en mots d'un texte donné. Une application qu'on a décidé de nommer : dwc (distributed words counter).

De manière générale, ce TP propose de vérifier expérimentalement s'il est possible que la conception du noyau d'ALMOS et les applications actuelles, reposant sur la notion de threads et de mémoire partagée, puissent passer à l'échelle en nombre de coeurs.

Mise en place de l'environement :

Avant de commencer l'implémentation de notre programme, il a d'abord fallut mettre en place l'environement cible pour notre exécution, en l'occurence ici la distribution ALMOS pour TSAR.

On commence donc par télécharger la dernière distribution stable d'ALMOS à l'adresse :
.

```
https://www-soc.lip6.fr/trac/almos/chrome/site/almos-tsar-
    mipsel-1.0.tbz2
```

Ensuite, il faudra décompresser l'archive :
.

```
$ tar jxf almos-tsar-mipsel-1.0.tbz2
```

Dans notre cas, on a eu quelques soucis liés à l'absence de packets dont dépendait l'utilisation du simulateur, on a donc eu à installer la liste de packets suivante :
.

```
libc6-i386 lib32stdc++6 lib32gcc1 lib32ncurses5 lib32z1
```

Les répertoires importants dans notre cas sont :

```
dans almos-tsar-mipsel-1.0
-> ./test/pf1 : // contient le makefile launcheur de TSAR (
    make sim<nb_clusters> )
-> ./apps : // repertoire contenant les sources de nos
    applications
```

Description de l'expérience :
Le programme à écrire pour cette expérience est le suivant :
Il faut créer un compteur de mot regroupé selon leur taille en nombre de caractères. Pour cela il nous faut un générateur de mots qui initialisera le contenu à analyser. Ensuite notre main lancera l'initialisation des différents threads et se chargera de repartir les taches entres les threads.
Nous aurons donc trois parties différentes lors de l'exécution de notre code : la première partie séquentielle qui initialise les threads et donne le "top départ" pour que tout les threads commencent la deuxièmes partie qui est

parallèles en même temps enfin la troisième partie qui est séquentielle et ou le thread du main réunis tout les résultats des autres threads.

Description du code implémenté :

Analyse des résultats :

Conclusion :
En conclusion ce TP nous a permis de prendre en main la plateforme TSAR et l'os ALMOS. Elle nous a aussi permis de nous initier à l'univers manycores. On a pu aussi grâce à ce TP comprendre le problème de localité spatial pour les données et le fait qu'il fallait les rapprocher le plus possible des processus qui travailleront dessus en jouant sur l'auto next-touch. On a pu aussi observer les limites de la répartition alors même pour des opérations aussi "simples" que le comptage de caractères. Ce TP nous a aussi permis de comprendre un peu mieux le principe de la loi d'Amdahl, notamment grâce aux différentes mesures effectués sur les différents sim possibles.