

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 uniformes : 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 Al-maless 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'environnement :

Avant de commencer l'implémentation de notre programme, il a d'abord fallu mettre en place l'environnement cible pour notre exécution, en l'occurrence 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 lanceur de TSAR (
    make sim<nb_clusters> )
-> ./apps : // repertoire contenant les sources de nos
    applications

```

Description de l'application :

DWC (Distributed Words Counter) est une application de comptage distribué de nombre de mots dans un texte. Le choix de cette application, est motivé par la nécessité d’avoir une application parallèle dont on évaluera la performance (speedup). Cette application sera écrite en C, et utilisera l’API POSIX Threads pour exprimer le traitement parallèle, elle suivra le schéma de traitement illustré par la figure :

T0000000000000000000D00

La phase d'initialisation permet de remplir la structure de calcul, représentée dans notre cas par un tableau de mots générés aléatoirement et dont la taille sera définie à l'avance (parce qu'on veut pouvoir manipuler la taille de mots, histoire de pouvoir évaluer les performances sur plusieurs plages de mots). On lancera ensuite autant de taches "workers" qu'il y a de processeurs dans la plateforme matérielle, chaque tache "worker" aura sa plage de travail dans le tableau de mots, et produira un tableau correspondant au nombre de mots d'indice i du tableau de lettres. (ainsi array[3] correspondra au nombre de mots de taille 3). Dès que les "workers" auront fini leur calcul, le main rentrera en phase de merge pour rassembler tous les tableaux de résultats et faire les sommes de nombres de mots.

Une exécution de l'application rend le temps de traitement en nombre de cycles par phase du programme. on exécutera le programme, sur les mêmes données et pour la même configuration séquentiellement et de manière distribuée (pour les benches).

Notre rendu :

L'archive rendue contient :

```
-> ./apps/dwc/* // sources du programme dwc
-> ./apps/dwc/makefile // makefile le compilant
-> ./execMe.sh // script compile le programme et lance une
    serie de simulations qu on decrite dans le paragraphe
    precedant
```

Cette archive est à copier à la source de la distribution, c'est à dire dans le dossier almos-tsar-mipsel-1.0.

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.