CUDAisation du code impératif

Université d'Angers

Vendredi 03 juin 2016

Tuteur pédagogique :Jean Michel RICHER

Alexis BRIARD
Jason JAMET
Guillaume GRANDJEAN

Plan

Objectif

- Transformer un code impératif en CUDA

```
# pragma cuda thread_loop ( i ) params (x ,y ,z , size )
void sum ( float *x , float *y , float *z , int size ) {
        for ( int i =0; i < size ; ++ i ) {
                z [ i ] = x [ i ] + y [ i ];
        }
}
```

```
__global__ void kernel ( float *x , float *y , float *z , int size ) {
        int gtid = ...
        if ( gtid < size ) {
                z [ i ] = x [ i ] + y [ i ];
        }
}
```

Analyse générale

Les transformations à réaliser :

- Initialisation de la variable récupérée dans le thread_loop
- Remplacement de la boucle for en condition if
- Modification de l'entête de la fonction

Analyse générale

Eléments qui peuvent poser problème :

- La variable size
- La taille des blocs
- Recherche de la boucle à paralléliser

Informations à extraire du code :

- Pragma
- Boucle for
- Déclaration de variables
- Fonction non void

Solutions envisagées

Les solutions :

- Analyseur syntaxique / lexical
- Compilateur gcc
- Regex

Solution retenue

Analyseur syntaxique / lexical

Simple d'utilisation, maitrisé par les membres de l'équipe.

Base de travail et correction

- 1ère version from scratch
- 2nd version basée sur l'implémentation yacc/lex de Jeff Lee
- 3ème version basée sur le projet "tc-parser" (github)

Analyse du pragma

- Définition du format
- Modification de l'arbre

#pragma cuda thread_loop(j) block_size(2,2) nbr_threads(16)

Recherche de la boucle à paralléliser

- Parcours de la fonction à CUDAiser
- Recherche du thread_loop dans les paramètre des boucles for

```
for(i=0; i<size; ++i) {
    ...
}
for(j=0; size>j; j++) {
    ...
}
```

Vérification de la portée des variables

- Accessibilité des variables

Transformation de la fonction

Wrapper de la fonction transformée

Améliorations

- Parse complet du langage c
- Gestion plus complète de la boucle for
- Automatisation des allocations

Conclusion