Rapport de PPN Session 2

YEHOSSOU Kakpo

1 Introduction

L'algèbre linéaire de par son importance, est à la base de beaucoup de resolutions de problème Mathématique. Elle est aussi impliqué dans plusieurs simulation notamment dans la simulation numerique en mécanique,mathématique,en SVT et d'autres domes encore. Elle permet de ressoudre les problèmes qui peuvent se ramener sous forme linéie ou qui sont déjà linéaire. Notre étude sera particulièrement axé sur l'étude de produit scalaire de deux vecteurs réels, le produit matrice vecteur réel et le produit matrice matrice réel. En premier lieu, nous présenteron chaque modèles, leurs algarithmes et leurs parallélisations, en suite nous ferons des testes et présenterons les resultats et enfin une étude sur la performancede ces modèles sur notre machine.

1.1 Motivation

Ce travaille nous permettra :

- la rapiditer de l'excécution du code dans son exécution,
- De résoudre les problémes se ramenant sous forme linéaire,
- De manipuler efficassement les matrice éfficassement en profondeur
- Permet d'utiliser les algorithme de bloc pour une performance élevée
- Garantir le bon équilibrage des charges,
- D'avoir un faible coût de la communication pour des gros blocs (diminution du nombre de messages).

1.2 Présentation Séquentielle

1.2.1 Algorithme produit scalaire

Le produit scalaire est la multiplication entre deux vecteur ou madtrices de taille N donc l'algorithme se présente sous la forme suivate :

Algorithm 1 Détermination du produit scalaire

y = 0 X $\in R^n, einR^n$
$Debut$
$Pouri \in [0, N-1] \quad faire$
$y(i) = x(i) * u(i) + y(i)$
FinPour
Fin

C'est un algoritme de complexité Linéaire.

1.2.2 Algorithme Produit matrice vecteur

Le produit matrice vecteur est la multiplication d'une matrice de taille N par par un vecteur colonne de m

ême taille. suivant l'algorithme :

Algorithm 2 détermination de y= Ax+b

A$\in R^n, y, x \in RDebutPouri \in [0, taille]$
$Pourj \in [0, taille]$
$y(i, j) = A(i, j) * x(j)$
$FinPour$
$FinPour$

Fin

Cet type d'algoritme est de complexité quadratique.

1.2.3 Produit matrice matrice

C'es d'une matrice de taille n à une autre matrice de même nombre de ligne :

beginalgorithm déterminati A$\in R^n, B \in R^n DebutPouri \in [0, taille]$
$Pourj \in [0, taille]$

$Pour k \in [0, taille]$

y(i,j) = A(i,k)*x(j)
FinPour
FinPour

Fin

Il est de complexité cubique.

2 Paralélisation

En architecture distibure, ici chaque module de mémoire avec son processeur les processeurs, pour assurer une gar ntie de la communication entre elles, on utilise une connection réseau. Nous allons pouvoir mettre en place une communication entre les processeus.dans ca nous allons utiliser la Bibliothèque MPI pour paraléliser nous pouvons égallement utiliser juste la Bivliothèque OpenMp pour paralléliser les boucle tout en faisant attention à la loi Lamdal et en veillant à l'équilibrage des charges.

2.1 produis scalaire

Comme nous somme en présence d'une réduction pour le produit scalaire, nous faisons appel à la fonction MPI-Allreduce, qui nous perme de diffuser le resulta de reduction. Elle fait la réduction et diffuse au autres. chaque processus fait sa reduction et envoie son resultaquiest sommé. Tout le monde aura le même resultat. Nous allons donc envoyer un element de type MPI-Double. nous aurons donc un ellement de type MPI-Double pas dans notre etude, la racine ici c'est root et nous somme le MPI-COMM-WORD.

2.2 Produit matrice vecteur

Ici chaque procesus doit faire son produit matrice vecteur et sa somme pour calla nous utilisons MPI-Gather, cette fonction permet de rassembler les donnée distribuée. c'est un algorithme de tous vers un qu'on va utiliser alors.

2.3 Produit matrice matrice

Ici chaque procesus doit faire son produit matrice matrice et sa somme pour calla nous utilisons MPI-Gather, cette fonction permet de rassembler les données et les distribuées. c'est un algorithme de tous vers un qu'on va utiliser alors.

3 Profilage

Resources complementaire prifilage voir github: https://github.com/finagnon/PPN2