Maestría en inteligencia artificial aplicada

Cómputo en la nube

Tarea 1. Programación de una solución paralela

Christopher Valdez Cantú A01793549

29 de enero de 2023

Introducción

En esta actividad se busca utilizar la librería OpenMP para poder profundizar en el tema de programación paralela al sumar dos arreglos utilizando dicha técnica para generar un tercer arreglo e imprimirlo en la consola.

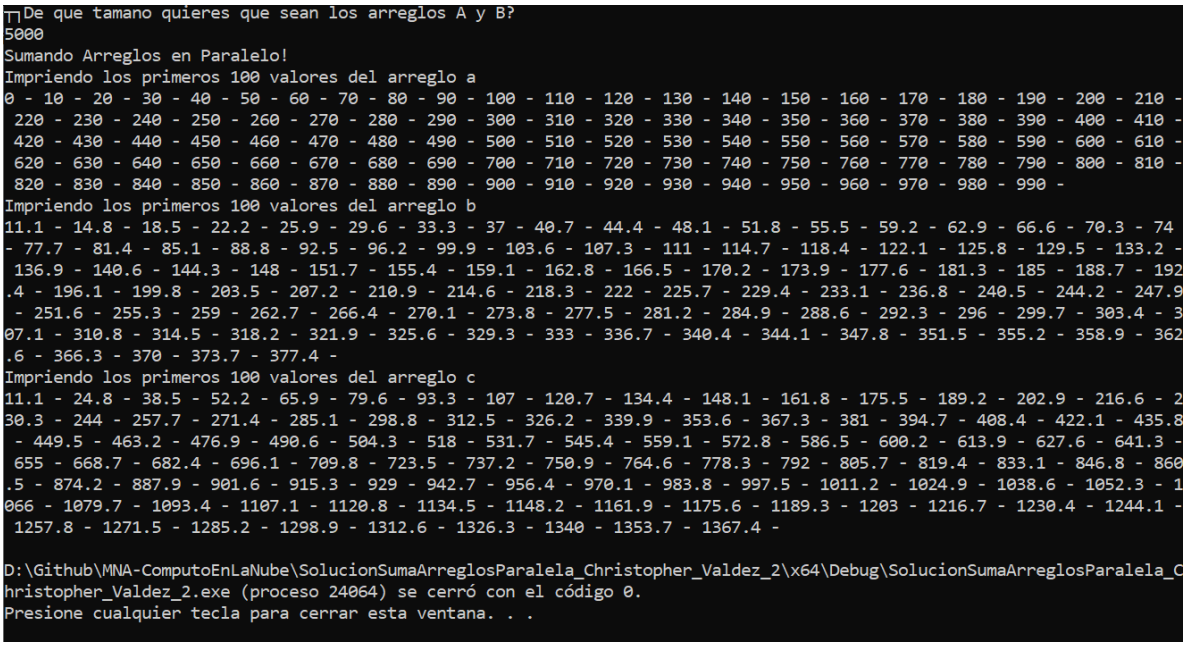
Esta actividad se realizará en Visual Studio y se subirá el código a GitHub

Liga de GitHub

https://github.com/ChrissValdez/MNA-ComputoEnLaNube

Capturas de pantalla

Ejecución con 5000 datos mostrando los primeros 100 valores.



Ejecución con 1000 datos mostrando los primeros 20.

```
De que tamano quieres que sean los arreglos A y B?
1000
Sumando Arreglos en Paralelo!
Impriendo los primeros 20 valores del arreglo a
0 - 10 - 20 - 30 - 40 - 50 - 60 - 70 - 80 - 90 - 100 - 110 - 120 - 130 - 140 - 150 - 160 - 170 - 180 - 190 -
Impriendo los primeros 20 valores del arreglo b
11.1 - 14.8 - 18.5 - 22.2 - 25.9 - 29.6 - 33.3 - 37 - 40.7 - 44.4 - 48.1 - 51.8 - 55.5 - 59.2 - 62.9 - 66.6 - 70.3 - 74
- 77.7 - 81.4 -
Impriendo los primeros 20 valores del arreglo c
11.1 - 24.8 - 38.5 - 52.2 - 65.9 - 79.6 - 93.3 - 107 - 120.7 - 134.4 - 148.1 - 161.8 - 175.5 - 189.2 - 202.9 - 216.6 - 2
30.3 - 244 - 257.7 - 271.4 -

D:\Github\MNA-ComputoEnLaNube\SolucionSumaArreglosParalela_Christopher_Valdez_2\x64\Debug\SolucionSumaArreglosParalela_C
hristopher_Valdez_2.exe (proceso 3492) se cerró con el código 0.
Presione cualquier tecla para cerrar esta ventana. . .
```

Código

```cpp
#include <iostream>
#include <omp.h>
#include "SolucionSumaArreglosParalela_Christopher_Valdez_2.h"

#define chunk 100
#define mostrar 20


void imprimeArreglo(float* d) {
    for (int x = 0; x < mostrar; x++) {
        std::cout << d[x] << " - ";
    }
    std::cout << std::endl;
}

int main()
{
    std::cout << "¿De que tamano quieres que sean los arreglos A y B?" << std::endl;
    int N;
    std::cin >> N;

    std::cout << "Sumando Arreglos en Paralelo!\n";

    float *a;
    a = new float[N];

    float *b;
    b = new float[N];

    float *c;
    c = new float[N];

    int i;

    for (i = 0; i < N; i++) {
        a[i] = i * 10;
        b[i] = (i + 3) * 3.7;
    }

    int pedazos = chunk;

    #pragma omp parallel for shared(a,b,c, pedazos) private(i) schedule(static, pedazos)
    for (i = 0; i < N; i++) {
        c[i] = a[i] + b[i];
     }

    std::cout << "Impriendo los primeros " << mostrar << " valores del arreglo a" << std::endl;
    imprimeArreglo(a);

    std::cout << "Impriendo los primeros " << mostrar << " valores del arreglo b" << std::endl;
    imprimeArreglo(b);

    std::cout << "Impriendo los primeros " << mostrar << " valores del arreglo c" << std::endl;
    imprimeArreglo(c);

}
```

Explicación del código y los resultados

El código primero define el tamaño de los arreglos, con la variable '**N'** la cual se le pregunta al usuario a través de la consola**,** y genera valores del arreglo '**A'** de 10 en 10 y del arreglo '**B'** multiplicando i + 3 * 3.7.

Después utilizando la librería OMP se hace la suma paralela la cual es más eficiente y rápida para sumar dichos valores y generar el arreglo '**C**'.

Después se imprimen los primeros "m" valores de los 3 arreglos en la consola, tamaño definido por la variable mostrar.

Reflexión

Esta actividad era muy simple y para fines prácticos utilizar programación paralela o hacer una suma de arreglos tradicional no hubiera hecho mayor diferencia en los tiempos de procesamiento de nuestra computadora, pero nos sirvió para probar el concepto y comprenderlo, al menos a mi me quedo mucho mas claro, porque se que si trabajamos, no con 1000 datos, sino con 1 millón, por ejemplo en Deep learning. O simplemente hubiera más pasos de procesamiento, no solo más cantidad de datos, entonces aun si es más difícil de programar sigue valiendo la pena ya que es mucho más rápido y puede ser la diferencia entre que el programa se ejecute por 3 días o 3 horas, especialmente trabajando en computo en la nube donde no solo se trabaja con una computadora, sino que se puede trabajar con 4 o 8 computadoras, dependiendo de la necesidad del proyecto o de la empresa.