Instituto Tecnológico y de Estudios Superiores de Monterrey

Maestría en Inteligencia Artificial Aplicada

Computo en la Nube

Felipe de Jesus Gastelum Lizarraga

A01114918

28/01/2023

Introducción

La programación párela hace referencia a solucionar un problema informático partiéndolo en pequeñas partes, las cuales serán resueltas de forma simultáneamente en diferentes recursos computacionales, posteriormente, las partes se unen para completar el problema. En el presente documento se describe la solución a un problema simple de suma de arreglos mediante la programación paralela.

Link de GitHub:

https://github.com/corvus18/T1-suma-de-Arreglos

Descripción de Problema:

Sumar dos arreglos llamados A y B de 100 elementos cada uno y generar un tercer arreglo llamado C, cuyos valores corresponderán a la sumatoria de A y B en los mismos subíndices.

Explicación de Código:

1. Se definen las librerías necesarias para imprimir texto en consola(iostream) y para realizar la programación paralela(omp.h)
2. Se definen 3 variables:
    a. **N:** Tamaño de arreglos
    b. **chunk:** tamaño de pedazos
    c. **mostrar:** Numero de valores a mostrar de arreglos
3. Se rellenan los arreglos A y B con valores arbitrarios
   A: valores de 10 en 10 hasta el 1000
   B: (índice + 3) * 3.7

```
4
5   #include <iostream>
6
7   #include <omp.h>
8
9
10  #define N 1000 // Tamaño de arreglos
11  #define chunk 100 //Tamaño de pedazos
12  #define mostrar 10 //Numero de valores a mostrar de arreglos
13
14  void imprimeArreglo(float* d);
15
16  int main()
17  {
18      std::cout << "Sumando Arreglos en Paralelo!\n";
19      float a[N], b[N], c[N];
20      int i;
21
22  //Generación de Información para rellenar arreglo
23      for (i = 0.0; i < N; i++) {
24          a[i] = i * 10.0;
25          b[i] = (i+3.0) * 3.7;
26      }
27
```

4. Se declara la variable “pedazos” con el valor de “chunk”
5. Se lleva a cabo un “For” paralelo a través de la librería “omp”, dividiendo el for en el tamaño de pedazos seleccionado. El For recorre los valores del arreglo A y B, sumándolos para generar el arreglo C.
6. Se imprimen los 10 primeros valores(de acuerdo a la variable “mostrar”) de los 3 arreglos: A, B y C

```
28
29      int pedazos = chunk;
30
31      //Suma de arreglos mediante FOR paralelo
32          #pragma omp parallel for \
33              shared(a,b,c,pedazos) private(i) \
34              shedule(static, pedazos)
35
36              for (int i = 0; i < N; i++)
37                  c[i] = a[i] + b[i];
38
39      //Impresión de Arreglos
40          std::cout << "Imprimiendo los primeros "<<mostrar<<" valores del arreglo a:"<< std::endl;
41          imprimeArreglo(a);
42
43          std::cout << "Imprimiendo los primeros " << mostrar << " valores del arreglo b:" << std::endl;
44          imprimeArreglo(b);
45
46          std::cout << "Imprimiendo los primeros " << mostrar << " valores del arreglo c:" << std::endl;
47          imprimeArreglo(c);
48  }
49
50  void imprimeArreglo(float * d) {
51
52      for (int x = 0; x < mostrar; x++) {
53          std::cout << d[x] << " - ";
54      }
55      std::cout << std::endl;
56  }
```

7. 1ra Ejecución de proyecto, bajo los siguientes supuestos:
    - A: valores de 10 en 10 hasta el 1000
    - B: (índice + 3) * 3.7

```
Sumando Arreglos en Paralelo!
Imprimiendo los primeros 10 valores del arreglo a:
0 - 10 - 20 - 30 - 40 - 50 - 60 - 70 - 80 - 90 -
Imprimiendo los primeros 10 valores del arreglo b:
11.1 - 14.8 - 18.5 - 22.2 - 25.9 - 29.6 - 33.3 - 37 - 40.7 - 44.4 -
Imprimiendo los primeros 10 valores del arreglo c:
11.1 - 24.8 - 38.5 - 52.2 - 65.9 - 79.6 - 93.3 - 107 - 120.7 - 134.4 -
```

1. 2da Ejecución de proyecto, bajo los siguientes supuestos:
    - A: valores de 5 en 5 hasta el 1000
    - B: (índice + 4) * 2

```
Sumando Arreglos en Paralelo!
Imprimiendo los primeros 10 valores del arreglo a:
0 - 15 - 30 - 45 - 60 - 75 - 90 - 105 - 120 - 135 -
Imprimiendo los primeros 10 valores del arreglo b:
8 - 10 - 12 - 14 - 16 - 18 - 20 - 22 - 24 - 26 -
Imprimiendo los primeros 10 valores del arreglo c:
8 - 25 - 42 - 59 - 76 - 93 - 110 - 127 - 144 - 161 -
```

Resultados:

Los valores de los arreglos A y B se sumaron, cuya sumatoria corresponde a los elementos del arreglo “C”. la sumatoria de los arreglos se realizo de forma paralela, es decir, la suma de los elementos se dividió en 100 pedazos (variable chunk) que fueron procesados de forma simultánea, para posteriormente unirse y resolver el problema.

Reflexión de programación paralela:

Uno de los grandes retos a los cuales se enfrenta el análisis de grandes volúmenes de información es la demanda del poder de procesamiento. Existen problemas en donde se requieren realizar operaciones con millones o miles de millones de datos, lo cual puede afectar de forma negativa el rendimiento o tiempo de respuesta durante el procesamiento.

La programación paralela es un claro ejemplo de la frase: “divide y vencerás”, ya que propone el no resolver un gran problema a la vez, sino resolver múltiples y pequeños problemas al mismo tiempo. Destinar diferentes recursos computacionales para resolver una parte del problema y después unir todo, puede mejorar significativamente el tiempo de respuesta, puesto que la carga de procesamiento se descentraliza.

Es importante tener en cuenta, que no todos los problemas se pueden resolver de forma paralela, y que cuenta con algunas desventajas como la complejidad de la sincronización y el costo de energía, sin embargo, hoy en día la programación paralela en una de las mejores herramientas que tenemos para hacer frente al procesamiento masivo de información.