Universidad Católica
San Pablo

CIENCIA DE LA COMPUTACIÓN

Computación Paralela y Distribuida

Laboratorio - 1

Rodrigo Alonso Torres Sotomayor

CCOMP 8-1

"El alumno declara haber realizado el presente trabajo de acuerdo a las normas de la Universidad Católica San Pablo"

Introducción

En este trabajo se analizarán dos bucles FOR utilizando Replit. Para este análisis se realizarán diferentes pruebas con distintos casos y finalmente realizar una comparación dando a conocer que bucle es el más rápido y el por qué de dichos resultados.

Requisitos de la PC

En este trabajo se utilizó Replit y las caracteristicas de dicha plataforma son:

- 0.5 vCPUs: Medio núcleo de CPU virtual.
- RAM: 512 MB
- Espacio del código: 35 MB

Implementaciones

0.1 Implementación 1

Para la primera implementación de estos bucles, se definió la variable MAX con el valor de 1000, de igual manera dentro del main se definieron las variables i y j antes de los bucles.

0.2 Implementación 2

Para la segunda implementación al vector x se le asignó un valor a cada elemento que compone dicho vector.

```
for(int aux = 0; aux < MAX; aux++)
  {
    x[aux] = aux;
  }
```

0.3 Implementación 3

Para la tercera implementación a los vectores x y y se les asignó valores a los elementos que conforman dichos vectores.

```
for(int aux = 0; aux < MAX; aux++)
    {
      x[aux] = aux;
      y[aux] = aux;
    }
```

0.4 Implementación 4

Para la cuarta implementación los vectores x y y al igual que la matriz A se es asignó valores para no tener solo los constructores y valores por default. Es con esta cuarta implementación que se realizarán las pruebas de tiempo variando el valor asignado de la variable MAX, se toma está implementación para las pruebas ya que es de buena práctica inicializar los valores para evitar comportamientos indefinidos y mejorar la legibilidad del código.

```
for(int aux = 0; aux < MAX; aux++)
  {
    x[aux] = aux;
    y[aux] = aux;
  }
```

```
for(i = 0; i < MAX; i++)
  for (j = 0; j < MAX; j++)
    A[i][j] = 1;
```

Para los tiempos se utilizó la libreria chrono de c++, y así realizar las comparaciones de los resultados obtenidos para cada bucle.

Resultados

	Bucle 1	Bucle 2
Implementación 1	0.0132253	0.00939757
Implementación 2	0.0747999	0.0688111
Implementación 3	0.00471284	0.00297172
Implementación 4	0.00334244	0.00991031

Utilizando la Implementación 4 se realizaron las pruebas con distintos tamaños de datos y los resultados son:

	Bucle 1	Bucle 2
100 elementos 1	1.3823e-05	7e-06
1000 elementos	0.00334244	0.00991031
10000 elementos	0.407602	12.7432

Análisis

Se observá a través de los resultados que el Bucle 1 es más rápido que el Bucle 2 y mientras mayor sea el número de elementos se refleja en el tiempo una mayor diferencia.

Haciendo un análisis más preciso a los dos bucles que realizan la misma función de recorrer la matriz A aplicandole ciertas operaciones y ambos bucles obtienen los mismos resultados se observó que:

En el primer caso, es decir el primer FOR anidado, se observa que el primer bucle itera a través de las filas de la matriz y el segundo bucle itera a través de las columnas. Esto significa que se está accediendo a los elementos de la matriz A de manera consecutiva en la dirección de las columnas lo que refleja la localidad espacial. Además, los elementos de la matriz A y los elementos del vector x se acceden varias veces en el segundo bucle lo que refleja la localidad temporal. Esta disposición puede ser más eficiente en términos de localidad espacial, ya que aprovechas la caché para acceder a elementos cercanos en memoria.

En el segundo caso, es decir el segundo FOR anidado, se observa que el primer bucle itera a través de las columnas y el segundo bucle itera a través de las filas. Esto significa que se está accediendo a los elementos de la matriz A en la dirección de las filas lo que hace referencia a la localidad espacial. Sin embargo, como los elementos del vector x se utilizan en todos los cálculos dentro del segundo bucle, hay un menor grado de localidad temporal. Esto puede llevar a un mayor número de accesos a memoria en comparación con el primer FOR anidado, ya que los mismos elementos de x se acceden repetidamente a lo largo del segundo bucle.

Link del repositorio

https://github.com/RodATS/ComputacionParalela.git