Actividad: Uso eficiente de la memoria cache

Instituto Politécnico Nacional
Escuela Superior de Computo
Desarrollo de Sistemas Distribuidos
Adrian González Pardo
4CV1
21/01

7 de octubre de 2020

1. Código fuente:

```
class MultiplicaMatriz{
  static int N = 1000;
  static int[][] A = new int[N][N];
  static int[][] B = new int[N][N];
  static int[][] C = new int[N][N];

  public static void main(String[] args){
    long t1 = System.currentTimeMillis();
    // inicializa las matrices A y B
    for (int i = 0; i < N; i++)
      for (int j = 0; j < N; j++){
        A[i][j] = 2 * i - j;
        B[i][j] = i + 2 * j;
        C[i][j] = 0;
      }
    // multiplica la matriz A y la matriz B, el resultado queda en la matriz C
    for (int i = 0; i < N; i++)
      for (int j = 0; j < N; j++)
        for (int k = 0; k < N; k++)
           C[i][j] += A[i][k] * B[k][j];
    long t2 = System.currentTimeMillis();
    System.out.println("Tiempo: " + (t2 - t1) + "ms");
  }
}
```

En este código fuente podemos ver que las operacionel calculo de una matriz de NxN es o puede ser cálculada por las propiedades matemáticas las cuales son $\Sigma(fila\ x\ columna)$ la cual da lugar al cálculo de la matriz el cual puede tener una complejiad $O(N^3)$

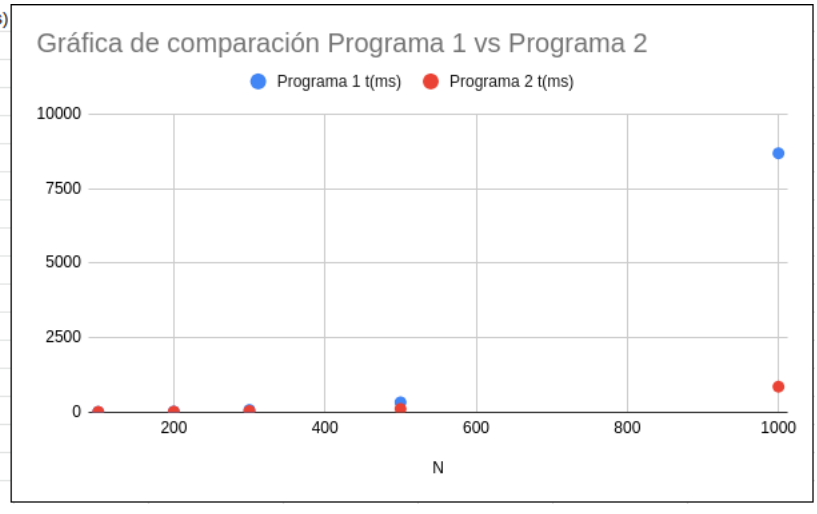
```
class MultiplicaMatriz2{
  static int N = 1000;
  static int[][] A = new int[N][N];
  static int[][] B = new int[N][N];
  static int[][] C = new int[N][N];

  public static void main(String[] args){
    long t1 = System.currentTimeMillis();
    // inicializa las matrices A y B
    for (int i = 0; i < N; i++)
      for (int j = 0; j < N; j++){
        A[i][j] = 2 * i - j;
        B[i][j] = i + 2 * j;
        C[i][j] = 0;
```

```
      }
    // transpone la matriz B, la matriz traspuesta queda en B
    for (int i = 0; i < N; i++)
      for (int j = 0; j < i; j++){
        int x = B[i][j];
        B[i][j] = B[j][i];
        B[j][i] = x;
      }
    // multiplica la matriz A y la matriz B, el resultado queda en la matriz C
    // notar que los indices de la matriz B se han intercambiado
    for (int i = 0; i < N; i++)
      for (int j = 0; j < N; j++)
        for (int k = 0; k < N; k++)
           C[i][j] += A[i][k] * B[j][k];
    long t2 = System.currentTimeMillis();
    System.out.println("Tiempo: " + (t2 - t1) + "ms");
  }
}
```

En este código fuente podemos ver que las operacionel calculo de una matriz de NxN es o puede ser cálculada por las propiedades matemáticas las cuales son $\Sigma(fila\ x\ columna)$ pero con una ligera modificación al código donde podemos ver que se genera una matriz transpuesta M^T de la matriz B por el cual es muy sencillo el acceso de la variable ya que no hay un movimiento $fila\ x\ columna$ sino que su acceso es $fila\ x\ fila$, por ello esta implementación es más eficiente en tiempo.

2. Gráfica de ejecución

N	Programa 1 t(ms)	Programa 2 t(ms)
100	16	16
200	26	23
300	81	45
500	336	114
1000	8683	861



Gráfica de descripción de la ejecución del programa.

3. Especificaciones del equipo que lo ejecuto

- Procesador Intel Core I5-3210M CPU 2.50GHz x 2 nucleos físicos (1 nucleo virtual por nucleo)
- 3 MB Intel Smart Cache
- 12 GB de RAM