UNIVERSIDAD DEL VALLE DE GUATEMALA

CC3069 - Computación Paralela y Distribuida

Sección 21

Ing. Miguel Novella Linares

Proyecto #2

Programación paralela con MPI

Davis Alvarez, 15842

Juan Solorzano, 18151

Mario Perdomo, 18029

GUATEMALA, 13 de abril de 2022

Introducción

La computación paralela permite realizar cálculos o procesos de manera simultánea, con ayuda de MPI es posible interconectar máquinas para crear conjuntos computacionales homogéneos y heterogéneos. MPI (Message Passing Interface) es un estándar para el modelo de memoria distribuida, que nos da la habilidad de interconectar máquinas en área local sin tener que depender de la topología de la red de interconexión.

DES es un algoritmo de cifrado de texto plano que utiliza una llave privada para cifrar y descifrar. Con ayuda de MPI se realizó un versión paralela un programa que encuentra la llave privada con la que fue cifrado un texto plano con ayuda del método de "fuerza bruta", el cual consiste en probar cada una de las posibles combinaciones de llaves que existen.
Para comprobar que se ha descifrado correctamente el texto se validó si el texto contiene una palabra clave, la cual se conoce a priori, dentro del texto resultante.

Objetivos

- Implementa y diseña programas para la paralelización de procesos con memoria distribuida usando OpenMPI.
- Optimizar el uso de recursos distribuidos y mejorar el speedup de un programa paralelo.
- Descubrir la llave privada usada para cifrar un texto, usando el método de fuerza bruta (brute force).

DES

Es un algoritmo de cifrado de bloques que toma texto plano en bloques de 64 bits y los convierte a texto cifrado utilizando claves de 48 bits. DES da como resultado una permutación entre los 2 ^ 64 posibles arreglos de 64 bits, cada uno de los cuales puede ser 0 o 1. Cada bloque de 64 bits se divide en dos bloques de 32 bits cada uno, un medio bloque izquierdo L y un medio derecho R.

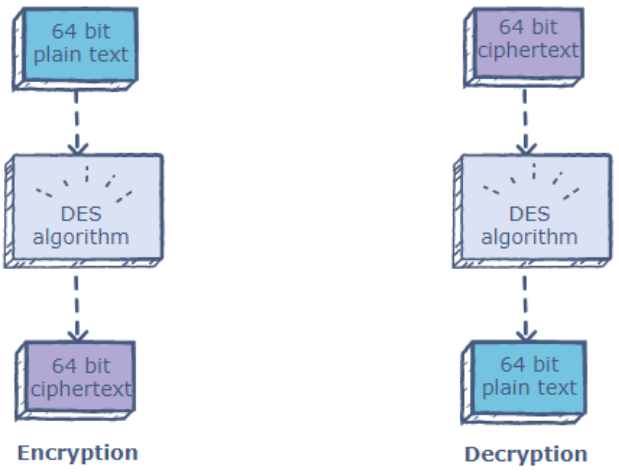


Fig. 1: Diagrama explicando la lógica de DES. (edpresso, 2020).

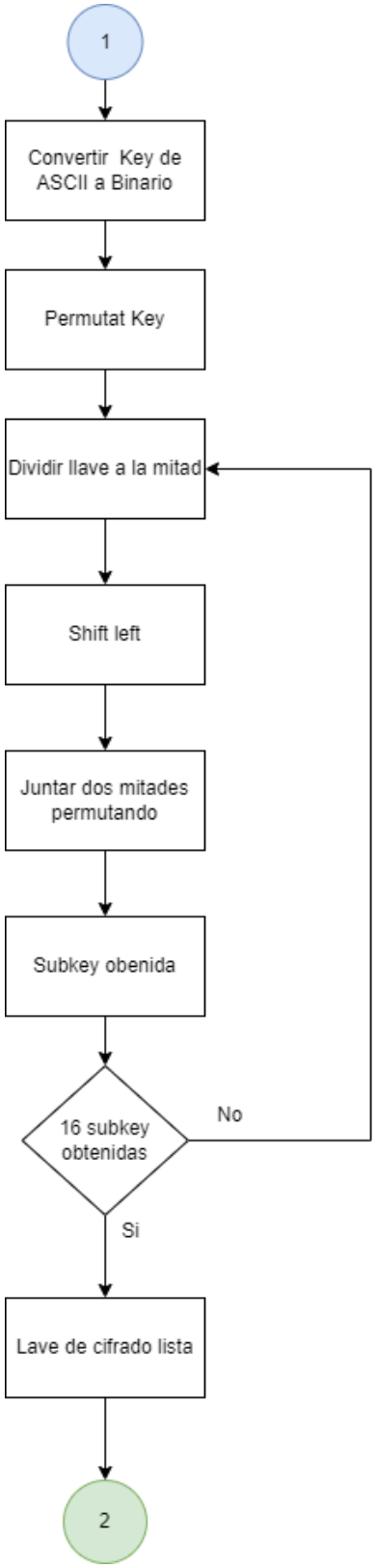
Es un algoritmo de clave simétrica, lo que significa que se utiliza la misma clave para cifrar y descifrar datos. Las keys se almacenan en realidad con una longitud de 64 bits, pero no se utiliza cada octavo bit de la clave (es decir, bits numerados 8, 16, 24, 32, 40, 48, 56 y 64). La longitud real es de 56 bits. (edpresso, 2020).

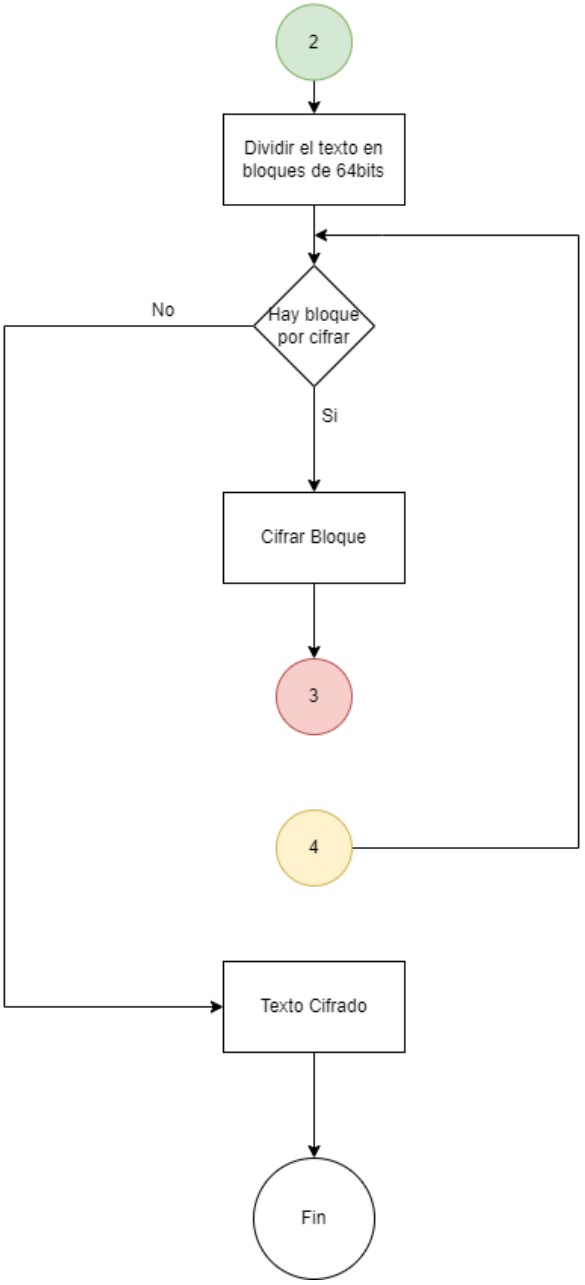
Pseudocódigo de este modelo:

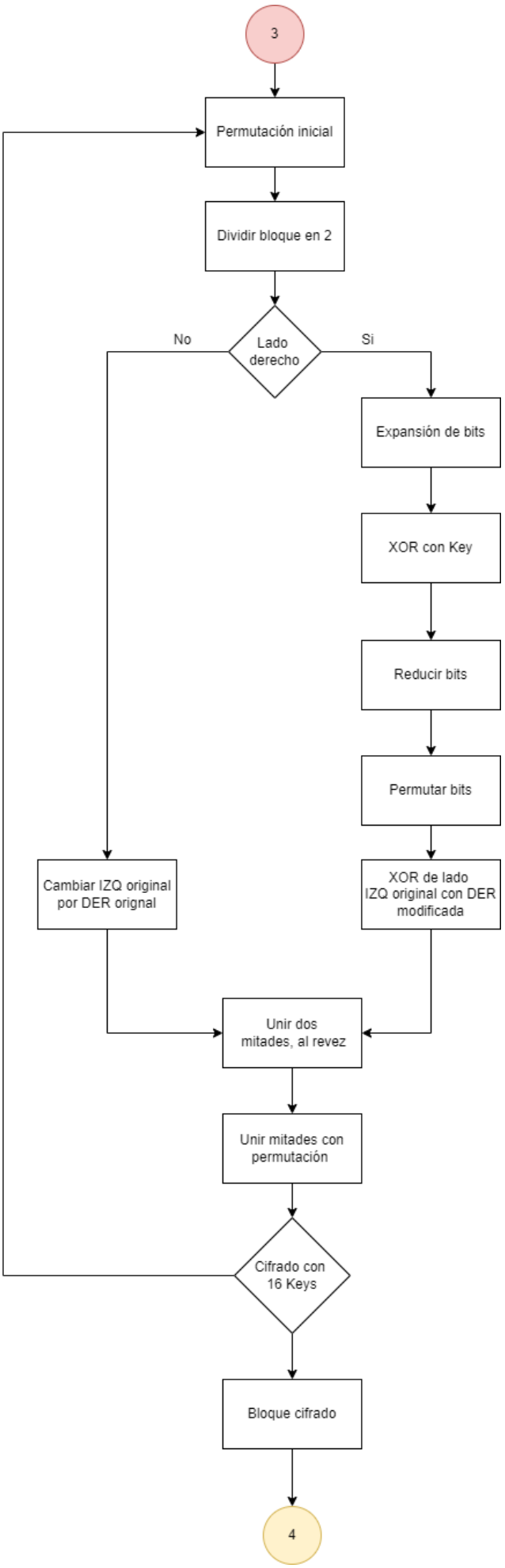
1. En el primer paso, el bloque de texto sin formato de 64 bits se transfiere a una función de Permutación (IP) inicial.
2. La permutación inicial realizada en texto plano.
3. A continuación, la permutación inicial (IP) produce dos mitades del bloque permutado; dice Texto sin formato izquierdo (LPT) y Texto sin formato derecho (RPT).
4. Ahora cada LPT y RPT pasan por 16 rondas de proceso de encriptación.
5. Al final, LPT y RPT se vuelven a unir y se realiza una permutación final (FP) en el bloque combinado
6. El resultado de este proceso produce texto cifrado de 64 bits.

(GeeksforGeeks, 2018).

Diagrama de flujo




2
Dividir el texto en bloques de 64bits
Hay bloque por cifrar
No
Si
Cifrar Bloque
3
4
Texto Cifrado
Fin


3
Permutación inicial
Dividir bloque en 2
Lado derecho
No
Si
Expansión de bits
XOR con Key
Reducir bits
Permutar bits
XOR de lado IZQ original con DER modificada
Cambiar IZQ original por DER orignal
Unir dos mitades, al revez
Unir mitades con permutación
Cifrado con 16 Keys
Bloque cifrado
4

Descripción de rutinas

decrypt()

Función que descifra utilizando la llave, bloques y su tamaño. Este itera en la llave haciendo shift los bits y luego utiliza librerías para setear el parity y descifrar.

tryKey()

Esta función busca si la palabra que se está buscando se encuentra en el texto a través de llamadas a mmcpy, luego decrypt para descifrar la llave y últimamente strstr para determinar si hay una ocurrencia del string.

memcpy()

Esta función recibe como parámetros a dest, src y n. Al ser llamada copia n caracteres del área de memoria src hacia la memoria dest y retorna un puntero al destino.

strstr()

Esta función recibe como parámetros a haystack y needle. Al ser llamada, la función busca la primera ocurrencia del string needle en el string haystack y retorna un puntero a dicha ocurrencia o null si no se encontró.

Uso y Flujo de comunicación de las primitivas de MPI:

MPI_Irecv

MPI_Irecv significa MPI (Receive with Immediate return); no se bloquea hasta que se recibe el mensaje. Para saber si el mensaje ha sido recibido, debe utilizar MPI_Wait o MPI_Test en el MPI_Request lleno.

MPI_Send

MPI_Send es el envío estándar en MPI. Entre bastidores, emitirá un envío con buffer MPI_Bsend o un envío síncrono MPI_Ssend. Esta decisión se basará en si el buffer adjunto para los envíos con buffer contiene suficiente espacio libre para el mensaje a enviar.

MPI_Wait

MPI_Wait espera a que se complete una operación no bloqueante. Es decir, a diferencia de MPI_Test, MPI_Wait se bloqueará hasta que la operación no bloqueante subyacente se complete. Dado que una operación no bloqueante retorna inmediatamente, lo hace antes de que la rutina MPI subyacente se complete. Esperar a que esa rutina se complete es para lo que está diseñado MPI_Wait.

(Rookiehpc, 2022).

Resultados

Ejecución del programa

```
diego@LAPTOP-SUTIIHV4:/mnt/c/Users/jdieg/Documents/UVG/A5S1/Comp Paralela/P2-Paralela$ mpirun -np 4
 ./brute
Found is: 36028797018963968
Work took 1.273155 s
```

```
diego@LAPTOP-SUTIIHV4:/mnt/c/Users/jdieg/Documents/UVG/A5S1/Comp Paralela/P2-Paralela$ mpirun -np 4
 ./brute
Found is: 36028797018963968
Work took 1.090765 s
```

Conclusiones/Recomendaciones

Nuestro grupo se enfrentó a bastantes retos en este proyecto, la mayoría siendo de la configuración del mismo. Mucho de nuestro tiempo se invirtió en la búsqueda de una librería que reemplazará la que se dio para la implementación de DES. Sin embargo, no se encontró ninguna opción viable además de crear nuestras propias funciones lo cual tardará aún más. Luego de una exhaustiva búsqueda y bastantes pruebas, se logró utilizar la librería inicial en Ubuntu 18 con WSL y las instalaciones adecuadas.

Para la solución del proyecto se implementaron varias funciones MPI vistas en clase, las cuales se encuentran anteriormente en la sección de descripción de rutinas. Luego de generar la lógica necesaria y colocar los parámetros adecuados, se logró desarrollar un programa que satisfactoriamente encuentre una cadena deseada a través de fuerza bruta.

Anexo

Anexo 1 - Recursos

- Fig. 1: edpresso. (2020). What is the DES algorithm. Extraído de: https://www.educative.io/edpresso/what-is-the-des-algorithm

Referencias Bibliográficas

- edpresso. (2020). What is the DES algorithm. Extraído de: https://www.educative.io/edpresso/what-is-the-des-algorithm
- GeeksforGeeks. (2018). Data encryption standard (DES) | Set 1. Extraído de: https://www.geeksforgeeks.org/data-encryption-standard-des-set-1/

- Rookiehpc. (2022). MPI documentation. Extraído de: https://www.rookiehpc.com/mpi/docs/index.php