**Taller 6**

Para instalar OpenMPI en CentOS 7, sigue estos pasos:

Primero, abre una terminal en tu sistema CentOS 7.

Asegúrate de que tu sistema esté actualizado ejecutando el siguiente comando:

|  |
| --- |
| sudo yum update -y |

Instala los paquetes de desarrollo necesarios para compilar OpenMPI:

|  |
| --- |
| sudo yum groupinstall -y "Development Tools" |

Instala otros paquetes requeridos para la compilación de OpenMPI:

|  |
| --- |
| sudo yum install -y wget numactl-devel |

Descarga la última versión de OpenMPI desde su sitio web oficial. En este ejemplo, estamos utilizando la versión 4.1.3, pero asegúrate de verificar si hay una versión más reciente disponible:

|  |
| --- |
| wget https://download.open-mpi.org/release/open-mpi/v4.1/openmpi-4.1.3.tar.gz |

Extrae el archivo comprimido:

|  |
| --- |
| tar xvf openmpi-4.1.3.tar.gz |

Cambia al directorio extraído:

|  |
| --- |
| cd openmpi-4.1.3 |

Configura y compila OpenMPI:

|  |
| --- |
| ./configure --prefix=/usr/local/openmpi  make -j $(nproc) |

Instala OpenMPI en el sistema:

|  |
| --- |
| sudo make install |

Configura las variables de entorno para usar OpenMPI. Edita el archivo .bashrc (o .bash\_profile) en tu directorio personal y añade las siguientes líneas al final:

|  |
| --- |
| export PATH=$PATH:/usr/local/openmpi/bin  export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/local/openmpi/lib |

Guarda el archivo y carga las nuevas variables de entorno ejecutando:

|  |
| --- |
| source ~/.bashrc |

¡Eso es todo! Ahora deberías tener OpenMPI instalado en tu sistema CentOS 7. Puedes verificar la instalación ejecutando:

|  |
| --- |
| mpicc --version |

**Ejemplo MPI**

MPI (Message Passing Interface) es una biblioteca utilizada para realizar computación paralela en C++ y otros lenguajes. Aquí tienes un ejemplo básico de código fuente en C++ que utiliza MPI para sincronizar procesos y calcular la suma de números enteros de 1 a N, donde cada proceso calcula una parte de la suma.

|  |
| --- |
| #include <iostream>  #include <mpi.h>  int main(int argc, char \*argv[]) {  int rank, size, N = 100;  int local\_sum = 0, global\_sum = 0;  int start, end;  // Inicializar MPI  MPI\_Init(&argc, &argv);  MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);  MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);  // Dividir el trabajo entre los procesos  int range = N / size;  start = rank \* range + 1;  end = (rank == size - 1) ? N : (rank + 1) \* range;  // Cada proceso calcula su suma local  for (int i = start; i <= end; ++i) {  local\_sum += i;  }  // Reducir las sumas locales a la suma global  MPI\_Reduce(&local\_sum, &global\_sum, 1, MPI\_INT, MPI\_SUM, 0, MPI\_COMM\_WORLD);  // Imprimir la suma global en el proceso maestro (rank 0)  if (rank == 0) {  std::cout << "La suma de los números enteros de 1 a " << N << " es " << global\_sum << std::endl;  }  // Finalizar MPI  MPI\_Finalize();  return 0;  } |

Este código crea una aplicación MPI que divide el problema de calcular la suma de los números enteros de 1 a N entre varios procesos. Cada proceso calcula su suma local y luego usa la función MPI\_Reduce para sumar todas las sumas locales en el proceso maestro (rank 0), que imprime la suma global.

Para compilar y ejecutar este programa, necesitarás tener instalado un compilador compatible con MPI, como mpic++. Puedes compilar y ejecutar el programa de la siguiente manera:

|  |
| --- |
| mpic++ main.cpp -o main  mpirun -np 4 ./main |

Esto compilará el programa y lo ejecutará con 4 procesos en paralelo.

**Solicitud**

Utilizar el proyecto de taller 5 (Backtracking) y realizar procesos paralelos utilizando mpi.