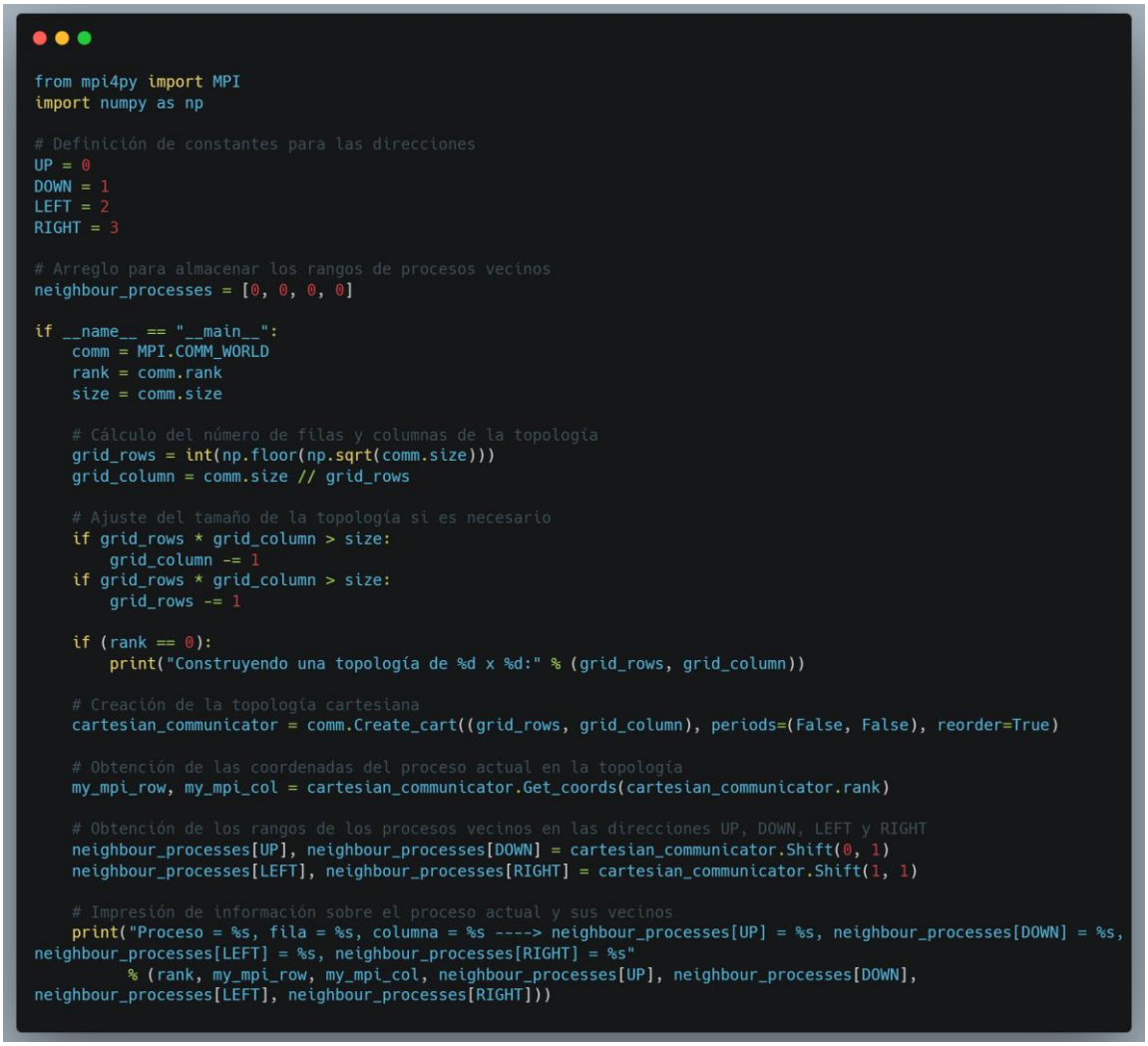
OPTIMIZACIÓN DE LA COMUNICACIÓN ENTRE PROCESOS

IRVYN XICALE CABRE
201963582

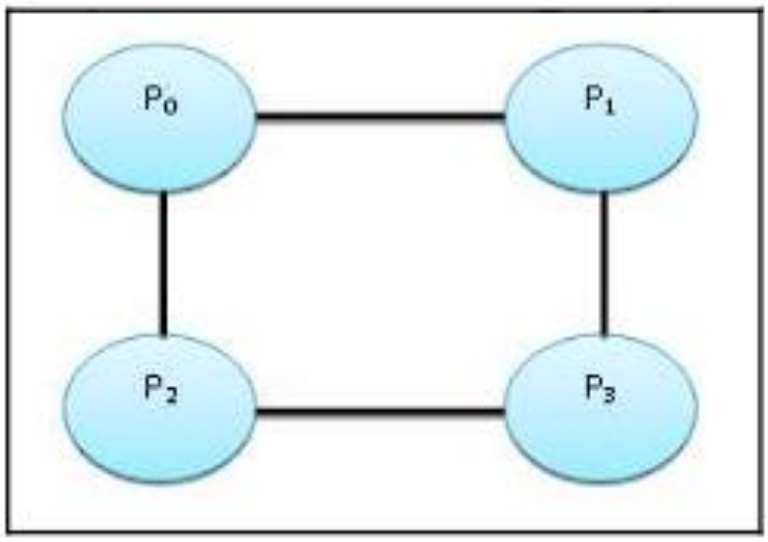
En este código creamos una topología virtual de tipo malla además de poder identificarlos procesos vecinos.



```python
from mpi4py import MPI
import numpy as np

# Definición de constantes para las direcciones
UP = 0
DOWN = 1
LEFT = 2
RIGHT = 3

# Arreglo para almacenar los rangos de procesos vecinos
neighbour_processes = [0, 0, 0, 0]

if __name__ == "__main__":
    comm = MPI.COMM_WORLD
    rank = comm.rank
    size = comm.size

    # Cálculo del número de filas y columnas de la topología
    grid_rows = int(np.floor(np.sqrt(comm.size)))
    grid_column = comm.size // grid_rows

    # Ajuste del tamaño de la topología si es necesario
    if grid_rows * grid_column > size:
        grid_column -= 1
    if grid_rows * grid_column > size:
        grid_rows -= 1

    if (rank == 0):
        print("Construyendo una topología de %d x %d:" % (grid_rows, grid_column))

    # Creación de la topología cartesiana
    cartesian_communicator = comm.Create_cart((grid_rows, grid_column), periods=(False, False), reorder=True)

    # Obtención de las coordenadas del proceso actual en la topología
    my_mpi_row, my_mpi_col = cartesian_communicator.Get_coords(cartesian_communicator.rank)

    # Obtención de los rangos de los procesos vecinos en las direcciones UP, DOWN, LEFT y RIGHT
    neighbour_processes[UP], neighbour_processes[DOWN] = cartesian_communicator.Shift(0, 1)
    neighbour_processes[LEFT], neighbour_processes[RIGHT] = cartesian_communicator.Shift(1, 1)

    # Impresión de información sobre el proceso actual y sus vecinos
    print("Proceso = %s, fila = %s, columna = %s ----> neighbour_processes[UP] = %s, neighbour_processes[DOWN] = %s,
neighbour_processes[LEFT] = %s, neighbour_processes[RIGHT] = %s"
          % (rank, my_mpi_row, my_mpi_col, neighbour_processes[UP], neighbour_processes[DOWN],
neighbour_processes[LEFT], neighbour_processes[RIGHT]))
```

```
(escuela) C:\Users\irvyn\OneDrive\Documents\programDistribuida\ComunicacionMPI>mpiexec -n 4 python ma
lla.py
Building a 2 x 2 grid topology:
Process = 0, row = 0, column = 0 ----> neighbour_processes[UP] = -1, neighbour_processes[DOWN] = 2, neighbour_processes[LEFT]
 = -1, neighbour_processes[RIGHT] = 1
Process = 1, row = 0, column = 1 ----> neighbour_processes[UP] = -1, neighbour_processes[DOWN] = 3, neighbour_processes[LEFT]
 = 0, neighbour_processes[RIGHT] = -1
Process = 3, row = 1, column = 1 ----> neighbour_processes[UP] = 1, neighbour_processes[DOWN] = -1, neighbour_processes[LEFT]
 = 2, neighbour_processes[RIGHT] = -1
Process = 2, row = 1, column = 0 ----> neighbour_processes[UP] = 0, neighbour_processes[DOWN] = -1, neighbour_processes[LEFT]
 = -1, neighbour_processes[RIGHT] = 3
```

Gracias a la salida podemos identificar que la topología es de la siguiente manera:

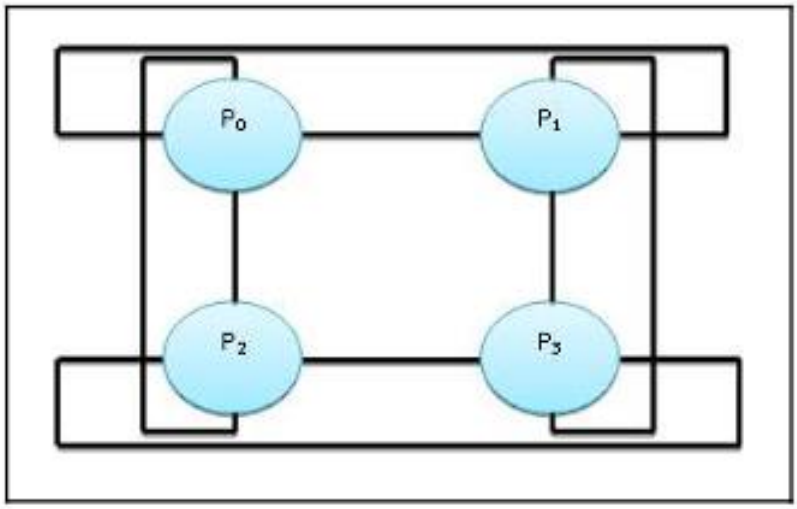


Además de la topología malla tenemos la topología toroidal que para crearla lo único que tenemos que hacer es cambiar comm.Create_cart() el parámetro de periods=(True, True)

```
# Creación de la topología
cartesian_communicator = comm.Create_cart((grid_rows, grid_column), periods=(True, True), reorder=True)
```

```
(escuela) C:\Users\irvyn\OneDrive\Documents\programDistribuida\ComunicacionMPI>mpiexec -n 4 python toroidal.py
Process = 1, row = 0, column = 1 ----> neighbour_processes[UP] = 3, neighbour_processes[DOWN] = 3, neighbour_processes[LEFT]
= 0, neighbour_processes[RIGHT] = 0
Building a 2 x 2 toroidal topology:
Process = 0, row = 0, column = 0 ----> neighbour_processes[UP] = 2, neighbour_processes[DOWN] = 2, neighbour_processes[LEFT]
= 1, neighbour_processes[RIGHT] = 1
Process = 2, row = 1, column = 0 ----> neighbour_processes[UP] = 0, neighbour_processes[DOWN] = 0, neighbour_processes[LEFT]
= 3, neighbour_processes[RIGHT] = 3
Process = 3, row = 1, column = 1 ----> neighbour_processes[UP] = 1, neighbour_processes[DOWN] = 1, neighbour_processes[LEFT]
= 2, neighbour_processes[RIGHT] = 2
```

Como podemos observar la salida nos daría una topología como se muestra en la siguiente imagen:



Algunas diferencias que logre notar en estas topologías son:

- Malla:
    - los procesos se organizan en una estructura similar a una cuadrícula o matriz.
    - La comunicación entre procesos vecinos se realiza directamente a lo largo de las direcciones de la cuadrícula.
    - La topología de malla es efectiva para aplicaciones en las que los procesos se comunican principalmente con sus vecinos inmediatos.
- Toroidal:
    - los procesos se organizan de manera que los bordes se conectan. Esto crea un ciclo continuo de procesos.
    - es útil cuando los procesos necesitan comunicarse a través de los bordes