

Universidad Nacional Autónoma de México  
Facultad de Ciencias  
Computación Distribuida  
**Reporte de Práctica 1**

## 1. Equipo

### Integrantes del equipo:

- Bonilla Reyes Dafne
- García Ponce José Camilo
- Ortega Venegas Rodrigo Aldair
- Rivera Mora Jesús Alberto

## 2. Desarrollo

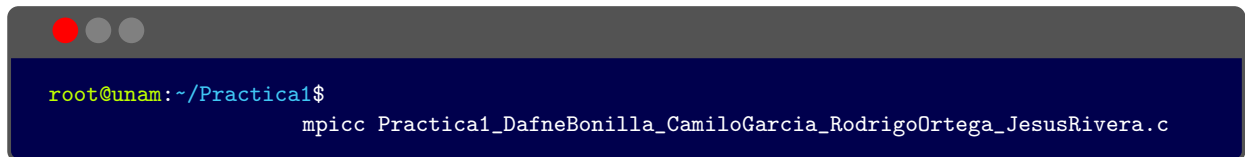
### 2.1 Descripción del desarrollo de la práctica y su funcionamiento:

- *Configuración:*
  - Importamos la biblioteca MPI (`#include <mpi.h>`) para poder realizar la programación distribuida.
  - Se definen las etiquetas (tags) para las operaciones de comunicación entre nodos como: `TAG_SOLICITUD`, `TAG_RESPUESTA`, `TAG_DISTANCIA`, etc.
- *Inicialización:*
  - Se inicializa el entorno MPI con `MPI_Init`.
  - Obtenemos el número de nodos y el rango del nodo actual.
- *Generación de retrasos aleatorios:*
  - Se utiliza `srand` para inicializar un generador de números aleatorios con una semilla única basada en la hora actual y el nodo actual.
  - Cada nodo genera números aleatorios entre 1 y 1000 para simular los retrasos de comunicación con otros nodos.
- *Distancias iniciales:*
  - Se obtiene la distancia inicial de un nodo respecto al resto, por medio de solicitudes y respuestas de los nodos.
- *Almacenamiento de distancia:*
  - Se crea un arreglo de arreglos para almacenar la ruta más corta de cada nodo.
  - Se crea un arreglo `distancia_cero[size]` para que cada nodo almacene el arreglo de distancias del nodo 0.
  - El nodo 0 copia los valores del arreglo `distancia[i]` al arreglo `distancia_cero[i]`
- *Actualización de distancias mínimas:*
  - Se inicia un ciclo que iterará hasta que no haya cambios en las distancias mínimas.
  - El nodo 0 envía su arreglo de distancias mínimas a todos los demás nodos. Los demás nodos actualizan sus distancias mínimas si encuentran una ruta más corta a través del nodo 0.
  - Los nodos envían su arreglo de distancias actualizadas al nodo 0.
- *Actualización de rutas más cortas:*
  - El nodo 0 verifica las distancias mínimas actualizadas y, si encuentra una ruta más corta, entonces actualiza el arreglo de arreglos de rutas.
  - Después, el nodo 0 envía un indicador de cambio o no cambio a todos los nodos.

- *Iteración y terminación:*
  - El proceso se repite hasta que no haya cambios en las rutas más cortas.
  - Una vez que no se detectan cambios, se sale del ciclo.
- *Resultados:*
  - Se muestra el retraso total de la ruta del nodo 0 a todos los demás nodos y el arreglo *distancia* de cada nodo.
  - Las rutas más cortas se construyen siguiendo el paso de actualización más corta.
- *Finalización:*
  - El programa MPI finaliza utilizando `MPI.Finalize()`.

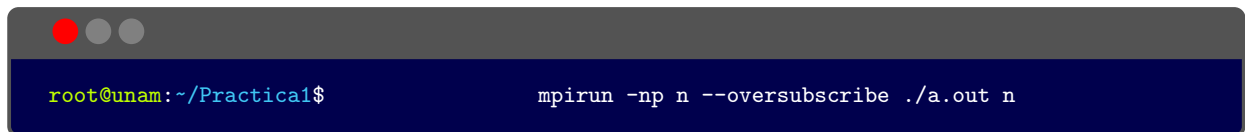
## 2.2 Compilación y salidas:

- Compilar desde `/Practica1`:



```
root@unam:~/Practica1$
mpicc Practica1_DafneBonilla_CamiloGarcia_RodrigoOrtega_JesusRivera.c
```

- Para ejecutar desde `/Practica1`:

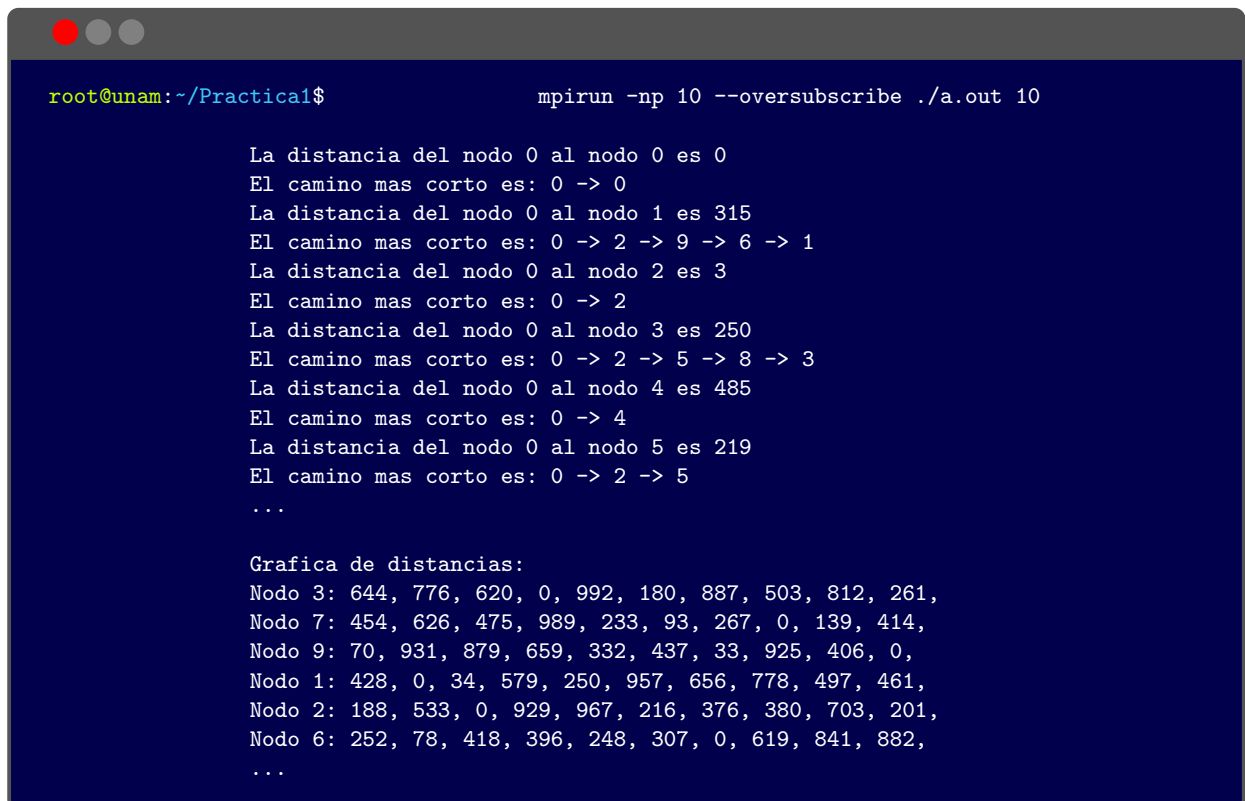


```
root@unam:~/Practica1$
mpirun -np n --oversubscribe ./a.out n
```

Donde `n` es el número de nodos deseados.

## 2.3 Ejemplos de salidas:

- Ejemplo con 10 nodos:



```
root@unam:~/Practica1$
mpirun -np 10 --oversubscribe ./a.out 10

La distancia del nodo 0 al nodo 0 es 0
El camino mas corto es: 0 -> 0
La distancia del nodo 0 al nodo 1 es 315
El camino mas corto es: 0 -> 2 -> 9 -> 6 -> 1
La distancia del nodo 0 al nodo 2 es 3
El camino mas corto es: 0 -> 2
La distancia del nodo 0 al nodo 3 es 250
El camino mas corto es: 0 -> 2 -> 5 -> 8 -> 3
La distancia del nodo 0 al nodo 4 es 485
El camino mas corto es: 0 -> 4
La distancia del nodo 0 al nodo 5 es 219
El camino mas corto es: 0 -> 2 -> 5
...

Grafica de distancias:
Nodo 3: 644, 776, 620, 0, 992, 180, 887, 503, 812, 261,
Nodo 7: 454, 626, 475, 989, 233, 93, 267, 0, 139, 414,
Nodo 9: 70, 931, 879, 659, 332, 437, 33, 925, 406, 0,
Nodo 1: 428, 0, 34, 579, 250, 957, 656, 778, 497, 461,
Nodo 2: 188, 533, 0, 929, 967, 216, 376, 380, 703, 201,
Nodo 6: 252, 78, 418, 396, 248, 307, 0, 619, 841, 882,
...
```

■ Ejemplo con 15 nodos:

```

root@unam:~/Practical1$ mpirun -np 15 --oversubscribe ./a.out 15

La distancia del nodo 0 al nodo 0 es 0
El camino mas corto es: 0 -> 0
La distancia del nodo 0 al nodo 1 es 194
El camino mas corto es: 0 -> 12 -> 13 -> 3 -> 1
La distancia del nodo 0 al nodo 2 es 110
El camino mas corto es: 0 -> 2
La distancia del nodo 0 al nodo 3 es 193
El camino mas corto es: 0 -> 12 -> 13 -> 3
La distancia del nodo 0 al nodo 4 es 172
El camino mas corto es: 0 -> 10 -> 5 -> 4
La distancia del nodo 0 al nodo 5 es 131
El camino mas corto es: 0 -> 10 -> 5
La distancia del nodo 0 al nodo 6 es 133
El camino mas corto es: 0 -> 2 -> 6
La distancia del nodo 0 al nodo 7 es 181
El camino mas corto es: 0 -> 2 -> 6 -> 7
La distancia del nodo 0 al nodo 8 es 204
El camino mas corto es: 0 -> 12 -> 8
La distancia del nodo 0 al nodo 9 es 249
El camino mas corto es: 0 -> 2 -> 6 -> 7 -> 9
...

Grafica de distancias:
Nodo 14: 589, 891, 448, 839, 411, 322, 750, 59, 461, 286, 589, 175, 119,
392, 0,
101,
Nodo 9: 442, 96, 557, 288, 726, 995, 941, 569, 273, 0, 250, 209, 853, 415,
201,
Nodo 11: 735, 29, 849, 369, 779, 936, 490, 107, 473, 196, 823, 0, 506, 857,
681,
Nodo 1: 441, 0, 458, 150, 400, 464, 920, 606, 26, 932, 669, 987, 760, 418,
254,
Nodo 3: 741, 1, 569, 0, 246, 295, 878, 921, 241, 276, 649, 893, 130, 477,
398,
Nodo 6: 891, 426, 910, 506, 224, 553, 0, 48, 659, 761, 426, 110, 726, 314,
142,
Nodo 8: 698, 716, 795, 665, 850, 238, 429, 674, 0, 791, 532, 495, 1, 756,
873,
Nodo 13: 342, 290, 747, 18, 81, 378, 403, 521, 238, 541, 666, 761, 238, 0,
526,
Nodo 5: 741, 737, 979, 767, 41, 0, 938, 64, 955, 154, 261, 999, 693, 870,
585, 441,
...
```

**Nota:** Los ... (tres puntos) en los ejemplos de salidas de ejecución indican que hay más texto, sin embargo, con el fin de no saturar el documento decidimos acortar el texto en estos ejemplos, pero en una ejecución normal en consola se muestra todo el resultado.

## 2.4 Comentarios:

Si el número de nodos es muy grande (por ejemplo más de 100 nodos), podría suceder que al intentar mostrar el camino más corto exista la posibilidad de que no se muestre todo, ya que usamos una cadena de tamaño 1100.

Además, es importante mencionar que usamos un algoritmo similar al de *Bellman-Ford*, debido a que consideramos que es más adecuado para esta práctica. Gracias a este algoritmo encontramos una forma no tan complicada de obtener los nodos que conforman la ruta más corta, de una manera algo recursiva.