MEMORIA BINGO

Guillermo Marquínez

En este proyecto se pide que construyamos un bingo utilizando el módulo MPI. En esta memoria voy a dar una explicación de cómo he hecho la práctica y cómo funciona el programa. Después haré un repaso sobre los problemas que me han surgido y el modo con el que los he intentado solucionar.

El primer paso es inicializar el bingo, el proceso 0 es el encargado de preparar el juego, él va a hacer de mesa, y por lo tanto no va a jugar. Para que este proceso cero pueda crear las plantillas que va a entregar a los jugadores, se ha creado la siguiente función.

```
def planilla(n):
    A=[]
    for i in range(0,n):
        x=random.randint(0,99)
        while x in A:
            x=random.randint(0,99)
        A.append(x)
        A.sort()
    return A
```

En esta función, la entrada n es la cantidad de números que queremos que tenga cada plantilla. En mi programa, en el desarrollo del juego he puesto que n=10. Como en la mayoría de los bingos, los números que pueden salir del bombo son los números de 0 al 99, ambos incluídos.

Además de hacer que se cree una plantilla para jugador, el proceso 0 tiene que comprobar que no entrega dos plantillas iguales a dos jugadores distintos. Por lo tanto, para comprobar eso, he optado por crear una lista de tantas plantillas como jugadores haya, teniendo cuidado de que cada plantilla nueva que se añade a la lista sea una plantilla nueva. Por esta razón, para poder compararlas, he ordenado cada plantilla con A.sort().

```
def planillas(num_jug, n):
    A=[0]
    for i in range (1,num_jug):
        x=planilla(n)
        while x in A:
            x=planilla(x)
        A.append(x)
    return(A)
```

Otra función que se utilizará durante el juego es bombo(x). En ella, el proceso cero saca un número aleatorio entre 0 y 99 que no haya salido anteriormente.

```
def bombo(A):
    x=random.randint(0,99)
    while x in A:
        x=random.randint(0,99)
    A.append(x)
    return(x)
```

Con estas funciones estamos en condiciones de iniciar el juego.
El proceso 0 va a repartir la lista de plantillas utilizando la función scatter(), entregando así a cada jugador i, la plantilla que está en la posición i. Por esta razón el primer valor de la lista es un 0, pues con scatter() el primer valor es para el proceso 0.

El siguiente paso que he hecho es el de que cada proceso imprima cuál es su plantilla, para comprobar que el juego se ha inicializado bien. Para que cada jugador imprima su plantilla siguiendo un orden, he hecho que cada jugador le envíe al proceso 0 el mensaje que quiere imprimir y el proceso 0 lo imprima, así me aseguro que siguen el orden que quiero. Todo ello está en los siguientes códigos.

```
if rank==0:
    A=planillas(size, 10)
    Bombo=[]
    contador=0
    Bingo=0
    victoria=0
     texto_plantilla="Soy la mesa y he dado a cada jugador su
plantilla para jugar, el Bingo empezará pronto"
    com.Barrier()

for i in range(1,size):
    if rank==i:
        contador=0
        A=None
        x=None
        Bingo=0
        com.Barrier()
plantilla=com.scatter(A,root=0)

for i in range(1,size):
    if rank==i:
            texto_plantilla="Soy el jugador "+str(rank)+ " y tengo la
planilla "+str(plantilla)

texto_plan=com.gather(texto_plantilla,root=0)

if rank==0:
    for i in range(len(texto_plan)):
        print(texto_plan[i])
```

Ahora ya estamos en condiciones de comenzar el bingo. Cada jugador tiene un contador inicializado en 0. El proceso cero comenzará a sacar bolas aleatoriamente y de una en una, utilizando la función bombo() y guardando los números que ya han salido en el vector Bombo. Con cada bola que saca, cada jugador comprobará si el nuevo número que ha salido está en su plantilla, en caso de que esté sumará uno a su contador particular. Todos los jugadores tienen una variable llamada Bingo. Esta variable comienza siendo 0 y pasa a ser 1 cuando el contador del jugador llega a 10. Después de que todos los jugadores hayan comprobado si el número que ha salido del bombo está en su plantilla, y después de haber ajustado su contador y su variable Bingo, todos los jugadores envían la variable Bingo a cero, utilizando el máximo de todas las variables con la función reduce(). Es decir, en el caso de que alguien tenga bingo, el máximo de las variables bingo es 1. Así el proceso cero se entera que el bingo ha acabado.

A partir de aquí he tenido el mayor problema, pues al recibir el dato de que bingo=1, el proceso 0 sabe que el juego ha terminado y que hay un ganador (todavía no sabe quién es). En este momento he intentado que el bucle que está sacando bolas del bombo se pare. Sin embargo, el break del bucle for no funcionaba y el bucle se quedaba atascado infinitamente. Como ya he explicado en Github, no he comprendido cuál es la razón.

De modo que he tenido que acabar este proceso de una forma un poco más fea. La idea es que, cuando el proceso cero reciba la noticia de que alguien tiene bingo, continúe sacando bolas hasta que el bucle termine, pero para que los jugadores no puedan seguir jugando he hecho que a partir de este momento el número que sale no es aleatorio, sino que siempre es el número 100. Como ningún jugador tiene este número en su planilla, nadie aumenta su contador. De esta manera, cuando el bucle acabe sabremos que el ganador es aquel que tiene el contador=10.
El código de este trozo es el siguiente (la funcionbingo comprueba si el número está en la plantilla):

```
for i in range(0,100):

    if rank==0:
        if Bingo==0:
            x=bombo(Bombo)
        else:
            x=100

    x=com.bcast(x,root=0)
    for i in range(1,size):
        if rank==i:
            if x in plantilla:
                contador+=1
            Bingo=funcionbingo(contador)

    Bingo=com.reduce(Bingo, op=MPI.MAX, root=0)
```

Por último, una vez acabado el bucle, cada jugador revisa su contador, en el caso de que el contador sea 10, el jugador crea una variable que se llama victoria y le da el valor 1. Además escribe el texto en el que dice que ha ganado. Todas estas variables se envían en un gather() al proceso 0. Este identifica en qué posición de la lista está el valor 1 y escribe quién es el ganador del bingo. Además va imprimiendo en orden todos los textos escritos de cada jugador, imprimiendo en primer lugar el texto del jugador campeón.

```
for i in range(1,size):
    if rank==i:
        if contador==10:
            text="Soy el jugador"+ str(rank)+ "y he ganado."
            victoria=1

        else:
            text="Soy el jugador"+str(rank)+ "y he perdido."
            victoria=0

result=com.gather(victoria, root=0)

if rank==0:
    print("El bombo ha sacado los números :" +str(Bombo))
    for i in range(1,size):
        if result[i]==1:
            text="Soy la mesa y ha ganado el jugador "+str(i)
texto=com.gather(text,root=0)
if rank==0:
    print(texto[0])
    for i in range(1,size):
        if result[i]==1:
            print(texto[i])
    for i in range(1,size):
        if result[i]==0:
            print(texto[i])




MPI.Finalize()
```