**Programación de Sistemas Distribuidos**

Curso 2014/2015

**Práctica 2**

**Message Passing Interface**

**(MPI)**

Autores:

* Borja Salazar Rey
* Juan Carlos Banda Carrillo

# Tabla de tiempo de los distintos ejercicios. (Pendiente)

|  |  |  |  |
| --- | --- | --- | --- |
| con Nº de procesos igual a… | Numero de cores físicos | Numero de cores lógicos | Mayor que Nº cores lógicos |
| Ejercicio 1 |  | | |
| Ejercicio 2 |  |  |  |
| Ejercicio 3 |  |  |  |
| Ejercicio 4 |  |  |  |
| Ejercicio 6 |  | | |
| Ejercicio 7 |  |  |  |

# Llamadas MPI utilizadas

## Ejercicio 2

* MPI\_Bcast: Esta función nos permite enviar un conjunto de datos a todos los demás procesos. Todos los demás procesos reciben los mismos datos.
* MPI\_Send: Esta función envía un único dato a un único proceso que se identificara mediante los parámetros de entrada. Para que ese proceso reciba el dato es necesario realizar la función explicada a continuación.
* MPI\_Recv: Esta función se encarga de recibir datos que hayan enviado otros procesos. Es necesario pasarla como parámetros los identificadores del proceso del que se espera el dato, además que buffer de entrada y demás datos.

## Ejercicio 3

* MPI\_Bcast: Explicada en el ejercicio 2
* MPI\_Isend: Esta función es similar a la función MPI\_Send, solo que envía los datos asíncronamente.
* MPI\_Irecv: Esta función es similar a MPI\_Recv solo que envía los datos de forma asíncrona.
* MPI\_Waitall: Una función que sirve para llevar un control del flujo de los procesos que usan Isend o Irecv. Esta función comprueba la request de cada Isend o Irecv que se le pase como parámetros (en una lista), y solo continuara el proceso cuando todos los request le indiquen que han terminado los Isend o Irecv.

## Ejercicio 4

* MPI\_Bcast: Explicado en el ejercicio 2.
* MPI\_Scatter: Esta función envía a cada proceso una parte de los datos que se le envíen como parámetro. Se le deben pasar los buffer de entrada (datos a enviar) y de salida así como en proceso que realiza el Scatter.
* MPI\_Gather: Esta función se encarga de agrupar los datos de todos los procesos en uno solo. Es la función contraria al scatter. Se le deben pasar como parámetros los buffer de entrada y salida, así como su tamaño, el Rank del proceso que realiza el gather y la variable de entorno MPI\_COMM\_WORLD.
* MPI\_Barrier: Para sincronizar adecuadamente todos los procesos utilizamos esta función. Lo que hace es esperar a que todos los procesos lleguen hasta ella para poder continuar.

## Ejercicio 7

* MPI\_Scatter: Explicada en el ejercicio 4.
* MPI\_Reduce: Esta función realiza una recolección de los datos de todos los procesos y los junta en uno de ellos que viene determinado en los parámetros de la función, y además realiza una operación sobre ellos y la guarda en algún parámetro de ese proceso.