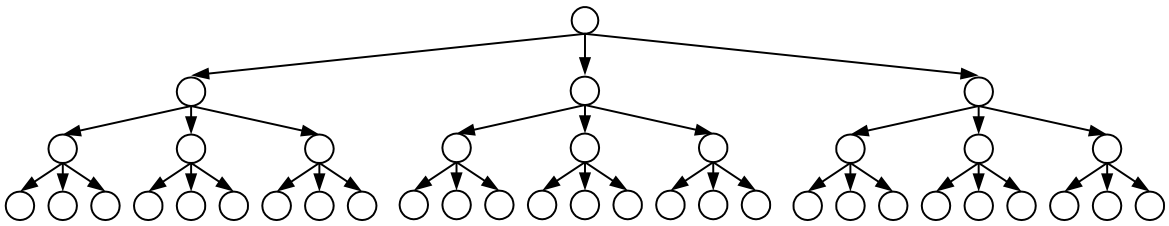
PROGRAMACIÓN PARALELA Y DISTRIBUIDA

PRACTICA Nº 3: PROGRAMACIÓN Y RENDIMIENTO

TEMAS: Ejercicios de programación en MPI. Rendimiento de aplicaciones paralelas.
OBJETIVOS: Que el alumno resuelva problemas a través del paralelismo, y desarrolle soluciones en lenguaje C sobre un cluster de computadoras mediante la librería de comunicación MPI. Asimismo, que evalúe y valore el comportamiento paralelo frente al secuencial.

Ejercicio Nº1 – En la siguiente figura se presenta un grafo de dependencias. Se asume que la ejecución de cada tarea insume una unidad de tiempo y que la comunicación entre procesos es instantánea (es decir que no consume tiempo).



Se proponen las siguientes actividades:

a) Calcular el grado de concurrencia.
b) Calcular el máximo Speedup posible si se dispone de una cantidad ilimitada de nodos computacionales.
c) Calcular Speedup y Eficiencia si el número de nodos computacionales es igual al grado de concurrencia.
d) Calcular Speedup y Eficiencia si el número de nodos computacionales es igual a la mitad del grado de concurrencia.
e) Calcular Speedup y Eficiencia si el número de nodos computacionales es igual a la tercera parte del grado de concurrencia.
f) Graficar el factor de Speedup y explicar los resultados obtenidos.

Ejercicio N°2 – Suponer que se desea calcular el valor de una variable llamada ***s***, cuyo valor se obtiene al sumar los primeros 16 números naturales. Resolver los siguientes incisos:

a) Suponer que la ejecución de cada suma requiere 1 unidad de tiempo (UT). Determinar cuántas operaciones (sumas) deben realizarse, y determinar cuánto tiempo se requiere si las cuentas se pudieran organizar en 1, 2, 4, 8, y 16 nodos.
b) Calcular el Speedup y la Eficiencia en cada caso.
c) Construir la correspondiente tabla de valores.
d) Graficar y analizar los resultados.

Ejercicio Nº3 – Escribir un programa en MPI que implemente el clásico “Hola Mundo!” y además de saludar indique en qué nodo se está ejecutando.

Ejercicio Nº4 – Escribir un programa en MPI que implemente un “ping-pong”, esto es, un programa en el que un proceso envía un mensaje a otro y éste último lo devuelve inmediatamente al primero. Utilizar la función MPI_Wtime para calcular cuánto tiempo se invierte en esta operación.

Ejercicio Nº5 – Dada la malla de procesos que se muestra a continuación, en la que cada proceso aparece con su rango en el comunicador MPI_COMM_WORLD. Escribir una única llamada a MPI_Comm_Split que permita obtener un nuevo comunicador (new_comm) que incluya a los procesos que aparecen con fondo gris en la figura pero cuyos rangos se ordenen de forma inversamente proporcional al rango que tenían en el comunicador original. Utilizar el menor número de órdenes posible.

a)

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

b)

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Ejercicio Nº6 – Re-implementar el juego de Siete y Medio realizado en la práctica Nº2, pero ahora siguiendo el esquema Master/Worker y comunicación mediante MPI.

Ejercicio Nº7 – Utilizando las operaciones punto-a-punto de MPI, implementar la función `Bcast(int source, char *msg, int tam)` que emulará a la operación de broadcast. ¿Cuántas iteraciones invierte la función?