ARQUITECTURAS DE SISTEMAS PARALELOS
Computación Paralela

Problemas Tema 2: Sistemas Multicomputador : Modelo de Paso de Mensajes

2.1.- Se utiliza la suma de n números para evaluar un sistema multicomputador con las siguientes características:

- Cada procesador dispone en memoria local de uno de los números a sumar.
- El programa disponible en cada procesador es una tarea para sumar dos números y por tanto si se desea sumar mas de dos hay que repetir tantas veces como sea necesaria dicha tarea.
- El tiempo de cálculo de una suma de dos elementos en un procesador es T
- El tiempo de comunicación para enviar un numero a otro procesador es C.

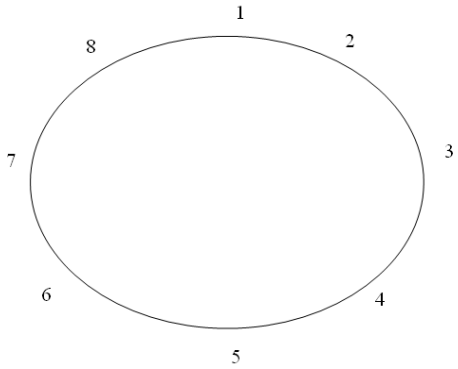
Se pide expresar los resultados en función del tamaño del problema, n, el número de procesadores y calcule en términos del tiempo de ejecución, el Speed-up respecto al algoritmo serie, la eficiencia del procesador y la función de isoeficiencia.

Se deben analizar los siguientes algoritmos para la suma de n números:

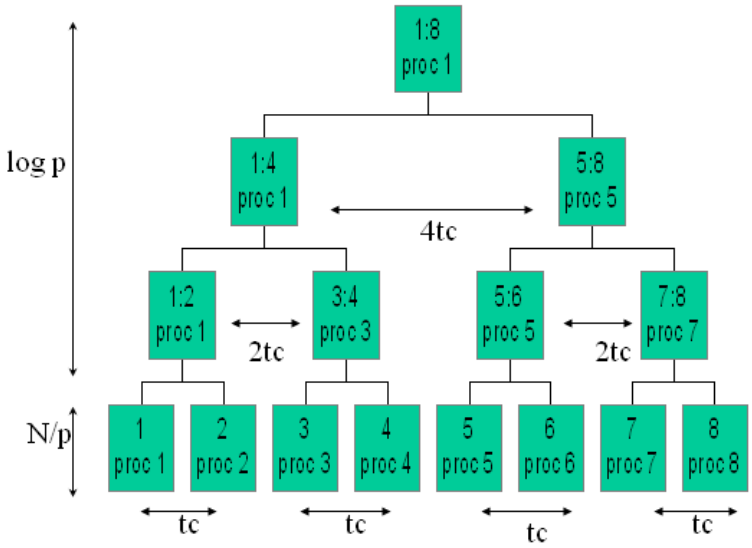
- Caso n = p (suponga que son potencias de 2) con un algoritmo que realiza los cálculos en etapas, cada etapa con una fase de comunicación y otra de computación, comunicando los datos entre procesadores con una estructura de árbol binario.
- Caso n >> p suponiendo que cada procesador tiene n/p elementos y utilizando el mismo algoritmo del caso anterior. Es decir, sobre un árbol con p procesadores, se repite n/p veces y el último procesador al final sumará el resultado de las n/p repeticiones.
- Caso n >> p suponiendo que cada procesador tiene n/p elementos, pero primero realiza localmente la suma de estos números y con las n/p sumas parciales utiliza un árbol con p procesadores.

2,2.- Un sistema con p=8 procesadores conectados en anillo se utiliza para calcular la suma de n números. Cada procesador almacena n/p números, el tiempo de comunicación entre vecinos próximos es t_c y el tiempo de una operación de computación elemental de suma es t_s. Analice el impacto de la comunicación al calcular el Speed-up si el algoritmo se realiza:

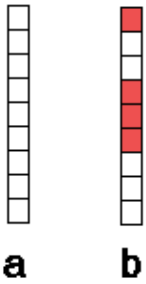
- Siguiendo el anillo de comunicación.
- Implementando sobre el anillo el algoritmo de comunicación basado en un árbol binario, como se indica en la figura, de tal manera que si por ejemplo la comunicación se realiza entre el procesador 1 y el 5 se tarda 4 t_c.



Exprese también el resultado en función de la relación r = t_c /t_s indicando para que valores de r se consiguen mejoras en el caso n=p=8.



2.3 En un sistema multicomputador se deben distribuir la carga de trabajo representada en la figura, donde cada cuadrado sin rellenar representa una tarea que se resuelve en una unidad de tiempo y los cuadrados rellenos son tareas que tardan 10 unidades de tiempo. Las comunicaciones no están representadas en la figura.



Se propone un esquema maestro y tres esclavos donde el maestro solo se encarga de la distribución de trabajo.
Se pide analizar en términos de Speed-Up para ambos casos (fig a y fig b)

1) Reparto estático. El master reparte inicialmente el trabajo entre los esclavos, dando a cada esclavo (num tareas/ num esclavos).
2) Reparto dinámico. El master reparte una tarea a cada esclavo y cuando termine le asigna una nueva.

Repita los apartados anteriores suponiendo que se añade el efecto de comunicaciones de tal manera que cada envío de trabajo implica una unidad de tiempo (tanto si se envía una tarea como si son mas).

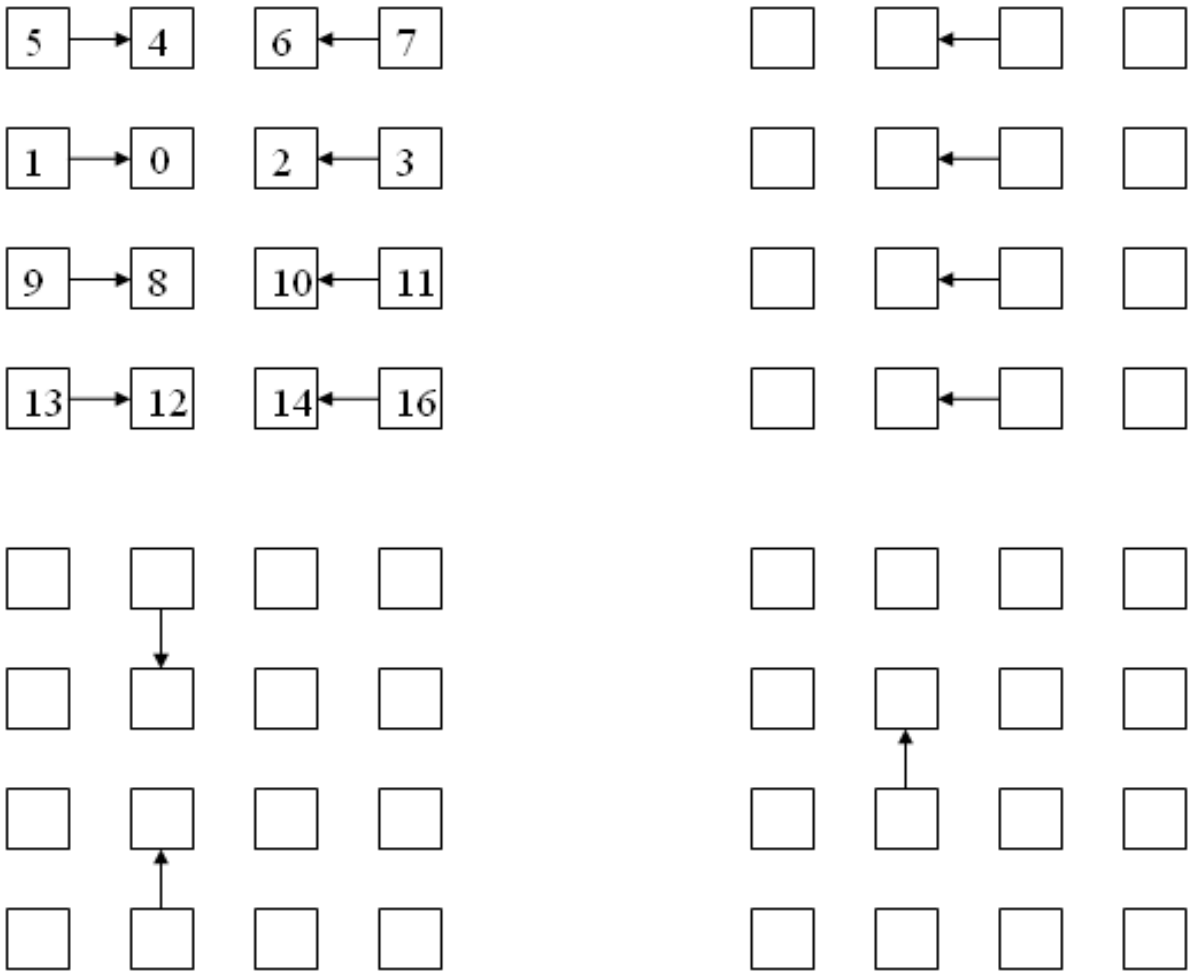
3) Para minimizar el efecto de comunicaciones se introduce un buffer en los esclavos capáz de mantener k = 2 tareas y solapar computación con comunicación. Indique dependiendo del funcionamiento del buffer, como se ve afectado el Speed-Up.

2.4.- Un sistema se modeliza como una máquina multiprocesador con p procesadores actuando sobre n números (p<=n). Se implementa un algoritmo que calcula si el valor x se encuentra entre los n números. El valor de x está en el procesador P0 y también debe almacenar el resultado de la búsqueda.

Pasos del algoritmo:
1.- Distribuye el valor de X a todos los procesadores.
2.- Cada procesador comprueba n/p números y pone un flag local .
3.- Comprueba si algún flag está activado.

- Indique la complejidad de cada paso del algoritmo en terminos de n (carga de trabajo) y p numero de procesadores.

2.5.- Un sistema con las comunicaciones en malla indicadas en la figura se utiliza para calcular la suma de n números. Cada procesador almacena uno de los números, el tiempo de comunicación entre vecinos próximos es t_c y el tiempo de una operación de computación elemental de suma es t_s. Analice el impacto de la comunicación al calcular el Speed-up



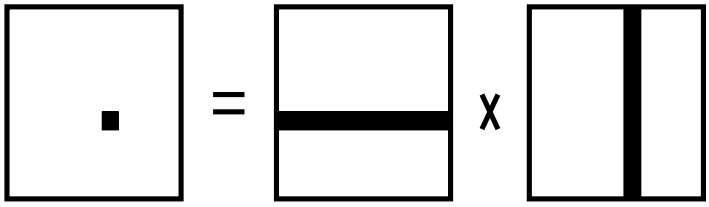
2.6.- Se pretende analizar el tiempo de ejecución de un programa que realiza el producto de dos matrices n x n en función de los siguientes parámetros: Número de procesadores, elementos de memoria, tamaño y número de mensajes, tiempo para las comunicaciones y tiempo de ejecución. Calcule el Speedup y la tendencia cuando n tiende a infinito.

Suponga el siguiente modelo de tiempos:

- El tiempo de una operación elemental entre dos números de la matriz es t_c
- El tiempo para el envío de un mensaje de x elementos se caracteriza por un tiempo de preparación de la comunicación independiente del tamaño (λ) y un tiempo asociado a la transferencia de cada elemento β, tal que el tiempo de comunicación es $\lambda + x\beta$.

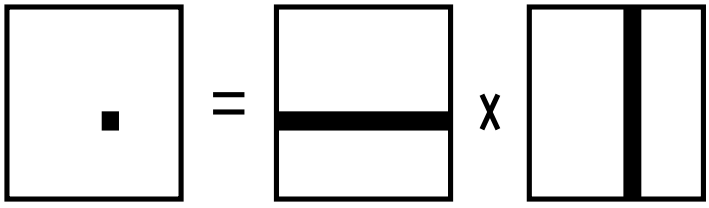
Evalúe la ejecución de los siguientes algoritmos:

a) algoritmo secuencial en un procesador:



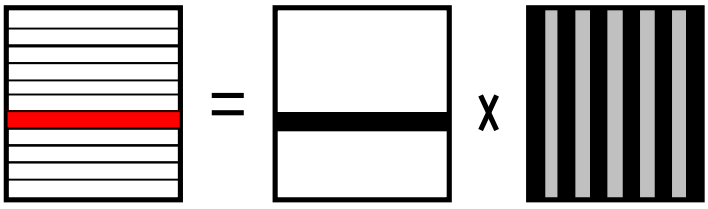
```
for ( i = 0;i < n, i++)
   for (j=0;j<n;j++)
         {
         t=0;
         for (k=0;k<n;k++)
               t=t+A[i][k]*B[k][j];
         C[i][j]= t;
         }
```

b) algoritmo MM1 con tantos procesadores como elementos tiene la matriz y cada procesador debe recibir uno de ellos trabaja con la fila y la columna que necesita. El algoritmo se describe con el pseudocodigo:



```
for all Pi con (i=0;i<n,i++)
   for all Pj con (j=0;j<n;j++)
         {
         t=0;
         for (k=0;k<n;k++)
               t=t+A[i][k]*B[k][j];
         C[i][j]= t;
         }
```

c) algoritmo MM2 con tantos procesadores como filas tiene la matriz y cada procesador debe recibir los datos necesarios para calcular esa fila. El algoritmo se describe con el pseudocodigo:



```
for all Pi con (i=0;i<n,i++)
   for (j=0;j<n;j++)
         {
         t=0;
         for (k=0;k<n;k++)
               t=t+A[i][k]*B[k][j];
         C[i][j]= t;
         }
```

2.7.- Estime el tiempo de comunicaciones necesarios para una comunicación de difusión uno-a-todos (broadcast one-to-all) de un mensaje de B bytes, suponiendo que el tiempo para el envío de un mensaje de x bytes es independiente del número de salto y se caracteriza por un tiempo de preparación de la comunicación independiente del tamaño (t_s) y un tiempo asociado a la transferencia de cada elemento t_w, tal que el tiempo de comunicación es $t_s + t_w \cdot x$.

a) En un anillo de P procesadores.
b) En una malla cuadrada con P procesadores tal que cada lado de la malla esta formado por sqrt (P) procesadores.

Represente gráficamente las etapas de la difusión de los mensajes.

2.8.- Considere el siguiente código MPI

```
int a[10], b[10], myrank;
MPI_Status status;
...
MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
if (myrank == 0) {
        MPI_Send(a, 10, MPI_INT, 1, 1, MPI_COMM_WORLD);
        MPI_Send(b, 10, MPI_INT, 1, 2, MPI_COMM_WORLD);
}
else if (myrank == 1) {
        MPI_Recv(b, 10, MPI_INT, 0, 2, &status, MPI_COMM_WORLD);
        MPI_Recv(a, 10, MPI_INT, 0, 1, &status, MPI_COMM_WORLD);
}
...
```

- Explique si el código puede producir un bloqueo.
- En su caso indique como se podría resolver:
    - Cambiando el orden de las sentencias.
    - Sin cambiar el orden cambiando los parámetros.
    - Sin cambiar el orden ni los parámetros.

2.9.- Suponga que el comunicador MPI COMM WORLD consta de tres procesos 0,1, y 2, y que se ejecuta el siguiente código

```
int x, y, z;
switch(my_rank) {
        case 0: x=0; y=1; z=2;
                MPI_Bcast(&x, 1, MPI_INT, 0, MPI_COMM_WORLD);
                MPI_Send(&y, 1, MPI_INT, 2, 43, MPI_COMM_WORLD);
                MPI_Bcast(&z, 1, MPI_INT, 1, MPI_COMM_WORLD);
                break;
        case 1: x=3; y=8; z=5;
                MPI_Bcast(&x, 1, MPI_INT, 0, MPI_COMM_WORLD);
                MPI_Bcast(&y, 1, MPI_INT, 1, MPI_COMM_WORLD);
                break;
        case 2: x=6; y=7; z=8;
                MPI_Bcast(&z, 1, MPI_INT, 0, MPI_COMM_WORLD);
                MPI_Recv(&x, 1, MPI_INT, 0, 43, MPI_COMM_WORLD, &status);
                MPI_Bcast(&y, 1, MPI_INT, 1, MPI_COMM_WORLD);
                break;
        }
```

¿ Cuál será el valores de x, y, z al finalizar la ejecución de cada proceso?

2.10.- En la descripción de MPI se usa el término buffer.

a) Indique las diferencias entre las siguientes tipos de buffers:
- Buffer utilizado por variables del programa.
- Buffer utilizado a nivel de usuario pero asignado a MPI.
- Buffer utilizado a nivel de sistema para MPI.

b) MPI dispone de las funciones send y receive con comportamiento tanto bloqueante como no-bloqueante.¿Qué supone con respecto a una variable de programa usada como buffer?

c) ¿ Cuál es la ventaja de combinar operaciones bloqueantes con operaciones no-bloqueantes?

2.11.- Dado el siguiente código ejecutandose en un sistema con tres procesadores:

```
...
double a,b;
int myRank,noOfProcessors;
MPI_Status status;

MPI_Init(...);
MPI_Comm_size( MPI_COMM_WORLD, &noOfProcessors );
MPI_Comm_rank( MPI_COMM_WORLD, &myRank );
a = myRank+1;
if (myRank==0) {
        for (int i=1; i<noOfProcessors; i++) {
                MPI_Recv(&b,1,MPI_DOUBLE,i,0,MPI_COMM_WORLD,&status);
                a+=b;
                }
        }
else {
MPI_Send(&a,1,MPI_DOUBLE,0,0,MPI_COMM_WORLD);
}
...
```

a) Indique que tipo de comunicación implementa este fragment de código.

b) Qué puede suceder si se utilizan comunicaciones no-bloqueantes en el fragment anterios sin añadir lineas adicionales?

2.12.- Considere el siguiente fragmento de código.

```
int a=0;
MPI_Send(&a, 1, MPI_INT, 2, 0, MPI_COMM_WORLD);
a=1;
```

```
MPI_Send(&a, 1, MPI_INT, 2, 0, MPI_COMM_WORLD);
```

Ejecutado en el nodo 0, y

```
Int a=2;
MPI_Send(&a, 1, MPI_INT, 2, 0, MPI_COMM_WORLD);
a=3;
MPI_Send(&a, 1, MPI_INT, 2, 0, MPI_COMM_WORLD);
```

Ejecutado en el nodo 1

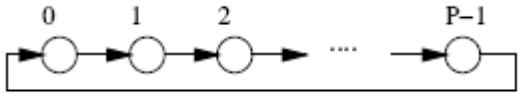
a) Cuál son los valores posibles para b0,b1,b2 y b3 cuando se ejecuten las cuatro sentencias MPI_Recv operations desde el node 2 ?
b) ¿ La solución es única?
c) ¿ Pueden adelantarse los mensajes enviados?

2.13.- Considere el siguiente fragmento incomplete de codigo (suponga que los valores de los arrays están correctamente inicializados) .

```
20: int a[10], b[10], npes, myrank;
21: ...
22: MPI_Comm_size(MPI_COMM_WORLD, &npes);
23: MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
24: MPI_Send(a, 10, MPI_INT, (myrank+1)%npes,1,MPI_COMM_WORLD);
25: MPI_Recv(b, 10, MPI_INT, (myrank-1+npes)%npes,1,MPI_COMM_WORLD);
26: ...
```

a) ¿Qué topología es la más adecuada para el código anterior?
b) Describe brevemente que hace el código cuando se ejecuta en un sistema paralelo con mas de un procesador?
c) Indique si el código funcionará correctamente (es seguro) si MPI usa un buffer para la operación send.
d) Suponiendo que MPI no usa buffer para la operación send pero que el programa es ejecutado en solo 2 procesadores. Indique si el fragmento de código es seguro para esta configuración.
e) Modifique el fragmento de código anterior para que sea seguro. Use solo funciones bloqueantes ,i.e. MPI Send() y MPI Recv(), partiendo el proceso en dos partes
f) Modifique el código tal que el programa funcione de manera segura tanto si el send usa buffer como si no lo utilize.

2.14.- Un programa realiza el producto de una **matriz m x m** números por un **vector de m** números en coma flotante. El sistema consta de p elementos de proceso (PE) o procesadores con memoria local comunicados en anillo como se indica en la figura.

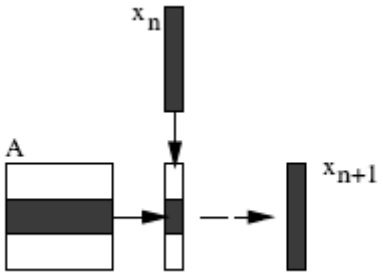


Las únicas comunicaciones son las indicadas y para un procesador determinado se permite recibir y enviar simultáneamente un mensaje. Suponga el siguiente modelo de tiempos:

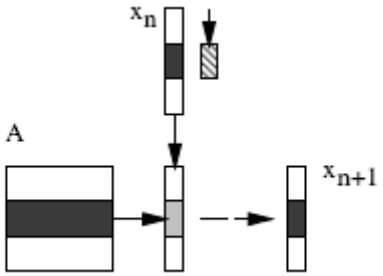
- El tiempo de una operación elemental entre dos números: $t_{suma} = t_{producto} = t_{flop}$
- El tiempo para el envío de un mensaje de q elementos se caracteriza por un tiempo de preparación de la comunicación independiente del tamaño (λ) y un tiempo asociado a la transferencia de cada elemento β, tal que el tiempo de comunicación es $\lambda + q\beta$.

Se consideran los siguientes particionamientos:

a) Cada PE inicialmente dispone en su memoria local de m/p filas de A y una copia de X. Indique el algoritmo realizado por cada PE para que al final el vector resultado quede guardado en la memoria local de cada procesador.



b) Cada PE inicialmente dispone en su memoria local de m/p filas de A y un trozo de X con m/p elementos. Los trozos de X se hacen circular por el anillo de PE, de tal manera que después de P fases de computo y comunicación X se ha enviado a todos los PE y cada uno ha realizado por partes el correspondiente calculo con el trozo recibido hasta que al final el vector resultado ha quedado guardado en la memoria local de cada procesador.



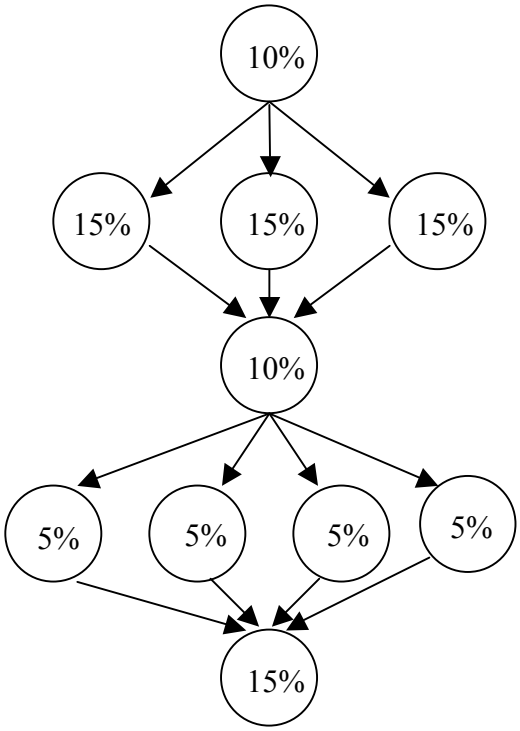
Se pide calcular en ambos casos y comparar los resultados obtenidos para:

- Los elementos de memoria local necesarios en cada PE.
- El tamaño del buffer de comunicaciones de cada PE y el número de mensajes utilizado.
- El tiempo para las comunicaciones y el tiempo de ejecución
- El Speedup.

2.15.- En la figura se representa el grafo de dependencias entre las tareas de una aplicación. Para cada tarea se indica la fracción de tiempo de ejecución secuencial que tarda. Si se supone que el tiempo de ejecución secuencial en un procesador es de 60 segundos y que las tareas no pueden dividirse en subtareas de menor granularidad y no hay penalización por las comunicaciones. Se pide obtener el tiempo de ejecución en paralelo y la aceleración obtenida con:

- 4 procesadores.
- 2 procesadores.
-

Represente el perfil de paralelismo del programa y el paralelismo real en función del tiempo que se consigue en cada caso.

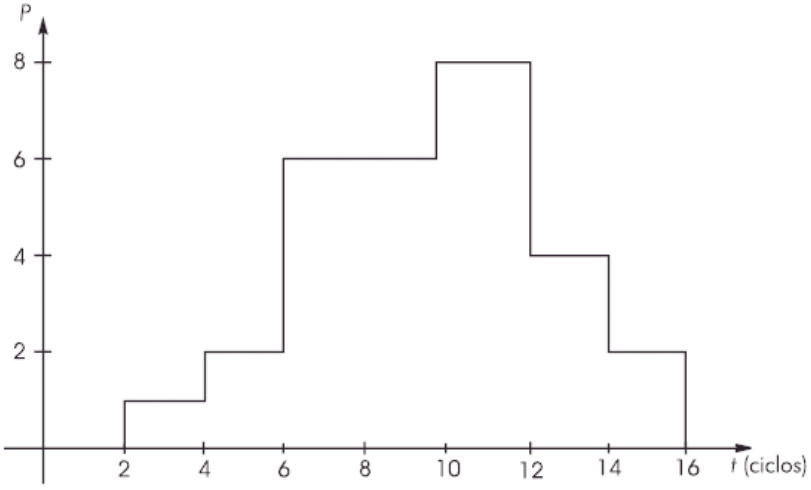


2.16.- Un 25% de un programa no se puede paralelizar, el resto se puede distribuir por igual entre cualquier número de procesadores sin considerar sobrecarga.

a) ¿ Cuál es el máximo valor de aceleración que se podría conseguir al paralelizalo?

b) ¿ A partir de qué número de procesadores se podría conseguir aceleraciones mayores o igual que 2?

2.17.- Supongamos que un programa tiene el perfil de paralelismo que se muestra en la figura:



a) Calcular el paralelismo medio del citado programa.
b) Calcular la carga de trabajo de ese proceso si la gráfica de la figura se ha obtenido con procesadores que ejecutan 2 instrucciones por ciclo.
c) ¿Cuál será la ganancia máxima de velocidad que podrá conseguirse cuando se ejecute ese programa en una máquina con 4 procesadores de la misma velocidad?
d) En las condiciones de los apartados anteriores: ¿Cuál sería la ganancia máxima de velocidad que se obtendría, en ese sistema, con un programa que tuviera un grado de paralelismo constante de 8?

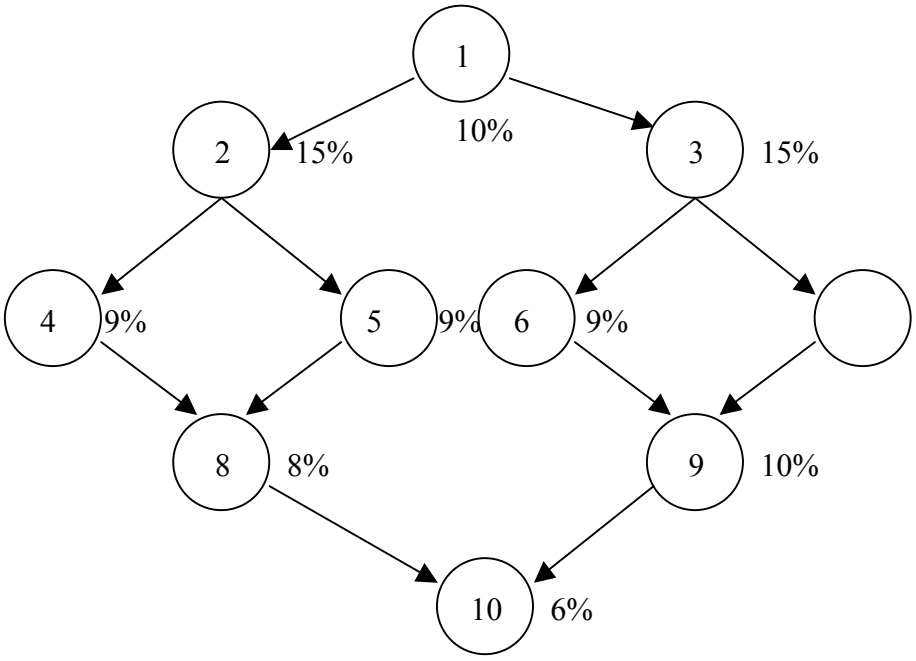
2.18.- Un programa, que tardaba 2 segundos en ejecutarse en un procesador, se ha dividido en 10 tareas. El orden de precedencia entre las tareas se representa en la siguiente figura, indicándose también el porcentaje de tiempo de cada una. Considerando que el tiempo de comunicación entre tareas se puede despreciar.

Represente el perfil de paralelismo del programa y calcule:

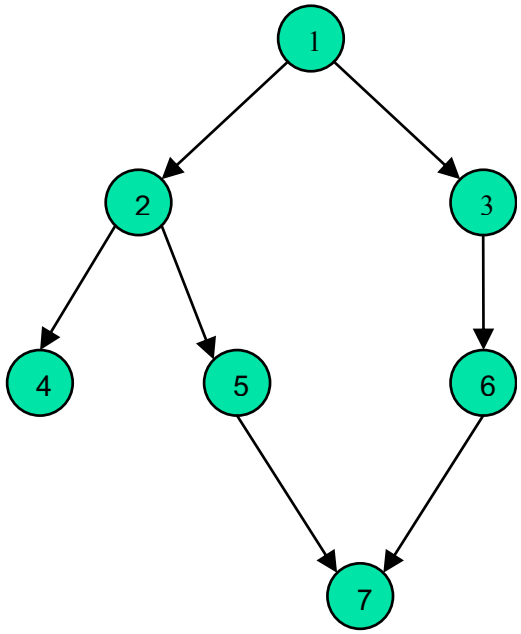
a) ¿Qué tiempo tarda en ejecutarse el programa en paralelo?

7 9%

b) ¿Qué aceleración se obtiene con respecto a la ejecución secuencial?



2.19.- La ejecución de un programa se distribuye en siete tareas con idéntico coste computacional y con las dependencias que se indican en el siguiente grafo.



En un sistema con 2 procesadores se planifican las tareas con el siguiente reparto;

P0	1	2	4	6	7
P1		3	5		

Calcule

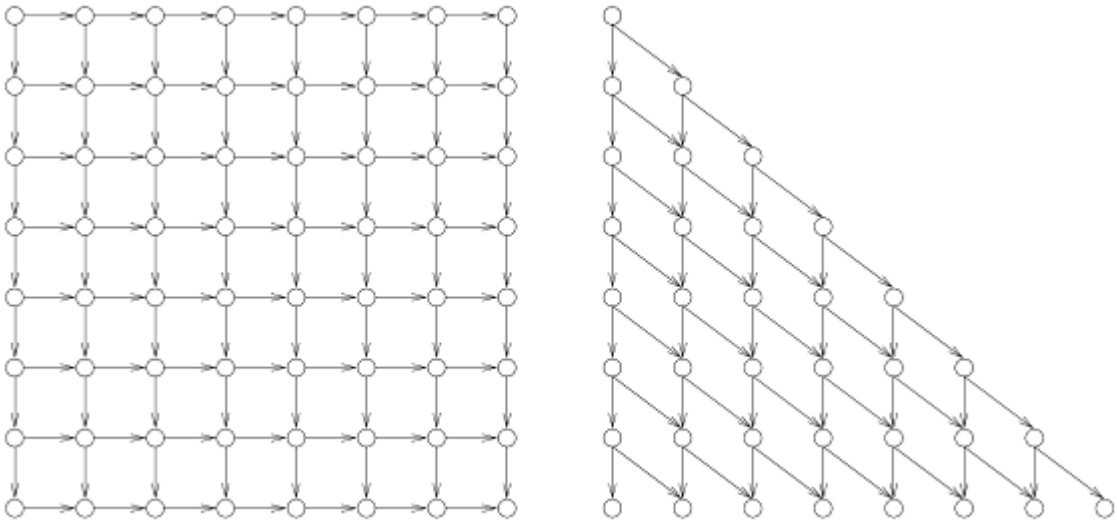
- Represente el perfil de parelismo del programa
- Indique el tiempo de ejecución en un sistema con un solo procesador.
- La aceleración conseguida si se dipone de dos procesadores y se utiliza la planificación de la tabla.
- ¿ Se puede obtener mayor aceleración para dos procesadores variando la planificación?. ¿Qué criterio mejora la planificación?
- Cuantos procesadores son necesarios para alcanzar la máxima aceleración posible.

2.20.- La eficiencia de un sistema con n procesadores es de un 89%, resolviendo un problema que inicialmente se caracteriza por tener una 2% no paralelizable. Calcule el número de procesadores utilizados en el sistema bajo los siguientes supuestos.

a) El trabajo computacional es de tamaño fijo y el sistema de n procesadores ha permitido reducir el tiempo de ejecución.

b) El trabajo computacional se ha escalado con en numero de procesadores para mejorar la precisión de la solución.

2.21.- La figura representan el grafo de dependencias de una aplicación que se va a ejecutar en un sistema multiprocesador con p procesadores. Si se generaliza denominando N al número de nodos en un grafo y n es un entero, de tal manera que en el grafo hay N= n x n nodos en la figura de la izquierda y N= n x (n+1)/2 nodos en la de la derecha.



Se supone que no hay penalización por las comunicaciones y que cada nodo representa una tarea que tarda una unidad de tiempo y no puede dividirse en subtareas de menor granularidad.

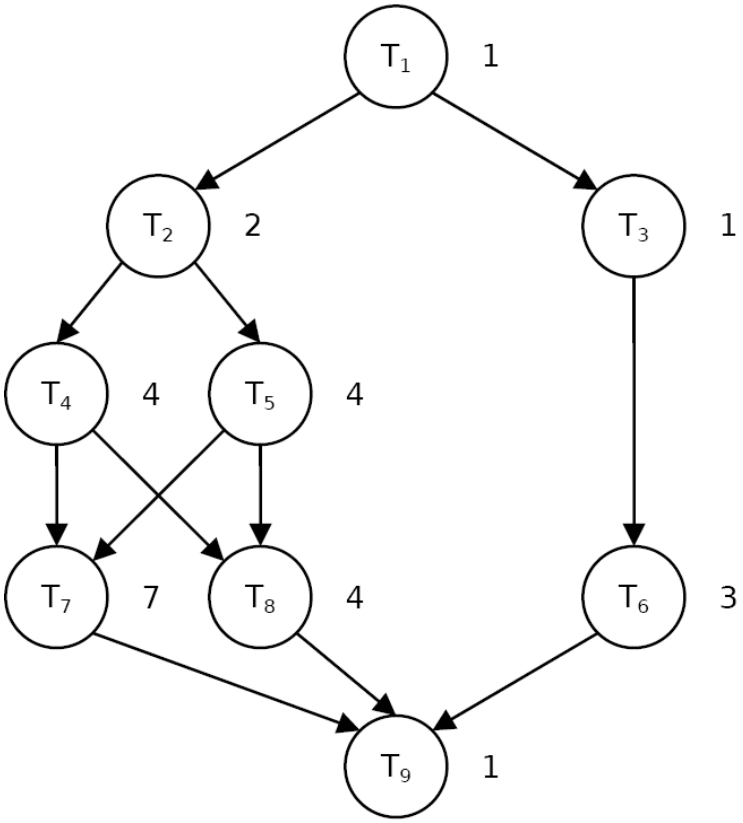
a) Represente el perfil de paralelismo durante la ejecución de la aplicación para n=8

b) El grado de paralelismo máximo y el trabajo realizado con grado de paralelimo tres .

c) Calcule el speed-up (S) y la eficiencia (E) en función del número de procesadores S(p) y compare para este caso los valores S(∞), S(p=4) y S(p=8)

d) Exprese el grado de paralelismo máximo y el Speed-up máximo en función de n.

2.22.- La figura representa el grafo de dependencias de una aplicación que se va a ejecutar en un sistema con p procesadores. Cada círculo representa una tarea independiente y el valor a su derecha representa el número de ciclos que tarda (i.e: T5 tarda 4 ciclos, T6 tarda 3 ciclos,…).



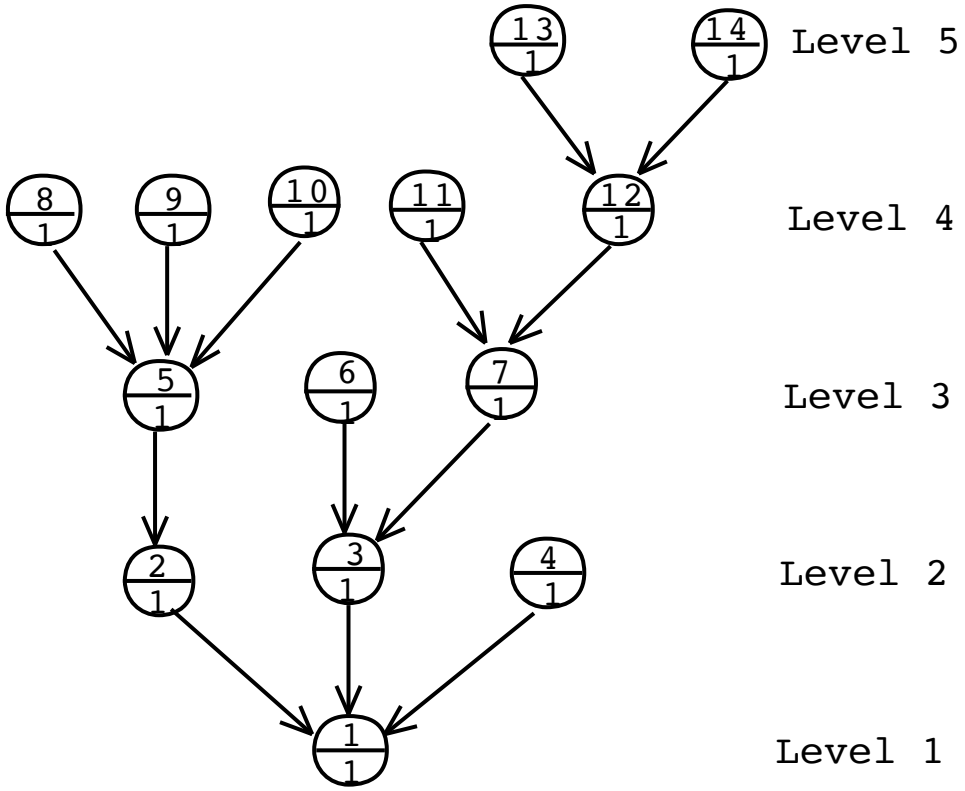
e) Represente el perfil de paralelismo de esta aplicación.

f) Calcule el grado de paralelismo máximo y el trabajo realizado con grado de paralelismo tres.

g) Calcule el speed-up (S) o aceleración en función del número de procesadores S(p) y compare para este caso los valores S(∞), S(p=2) y S(p=4).

h) Calcule el número de procesadores para que la eficiencia del sistema sea máxima y en ese caso represente en un cronograma la distribución de tareas en función del tiempo para cada procesador.

2.23.- El grafo de dependencias de un algoritmo es el indicado en la figura. Cada tarea se ejecuta en un ciclo.



El esquema de planificación asigna a cada tarea una prioridad definida como el número de sucesores de cada nodo y a cada procesador disponible le asigna la tarea lista sin ejecutar con la más alta prioridad .

a) Indique cual es el camino crítico que define la mayor longitud de planificación en unidades de tiempo.
b) Estime el número de procesadores del sistema que optimiza la ejecución del algoritmo.
c) Represente gráficamente el resultado de planificación en un sistema con 3 procesadores, indicando el tiempo final de ejecución.
d) Repita el apartado anterior con 2 procesadores