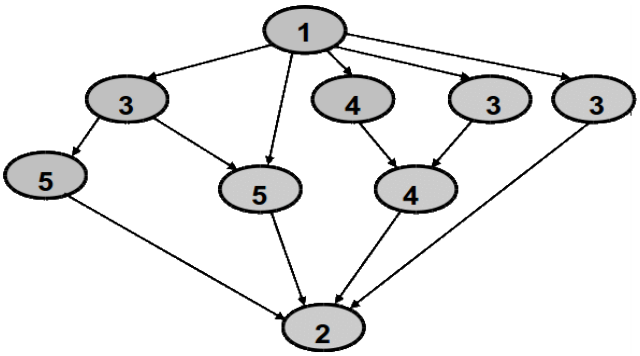
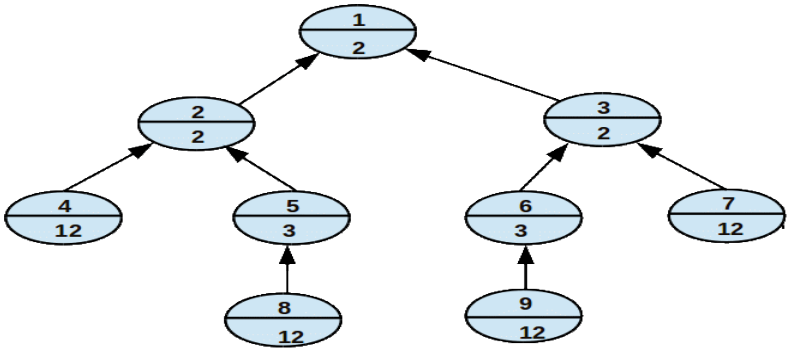
Programación Paralela
4º de Grado en Ingeniería Informática.

Ejercicios del Tema 3. Metodología de Diseño de Algoritmos Paralelos

1. ¿Cuál es el grado medio de concurrencia del siguiente grafo de dependencias? Asumiendo que el costo de comunicación fuese despreciable, cuál sería la asignación óptima de las tareas del grafo a dos procesos.

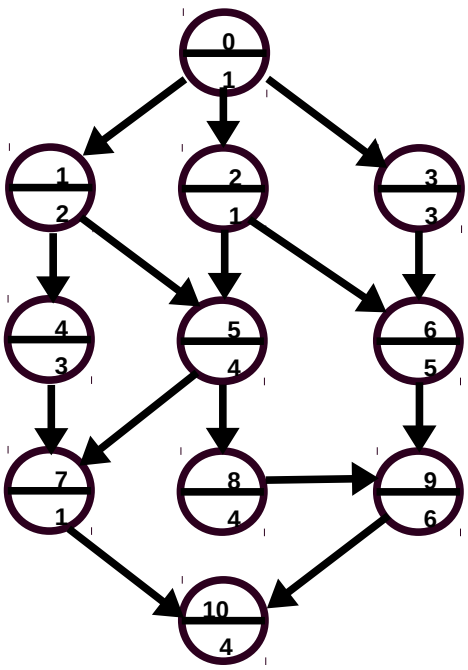


2. Cuál es el grado medio de concurrencia de los siguientes grafos de dependencias entre tareas en el que cada nodo está etiquetado con su identificación en la parte superior y con su coste en la parte inferior. Asumiendo que el costo de comunicación entre tareas fuese despreciable, establecer cuál sería la asignación óptima de las tareas del grafo a cuatro procesos en el caso a) y a tres procesos en el caso b). Dibujar un diagrama de ejecución.



a)

b)



3. ¿Se desea paralelizar un cálculo iterativo que tiene como entrada un vector de reales **y** de dimensión **N** (**y**=(**y1**,**y2**,…,**yN**)) y devuelve como salida otro vector real **dy** también de dimensión **N** que se calcula en como:

$$dy_i^{(k+1)} = \frac{y_{i-1}^{(k)} + y_i^{(k)} * y_{i+1}^{(k)} - y_{i+2}^{(k)}}{8}, \qquad k = 0,\ldots M.$$

donde los valores del lado derecho que entran fuera del rango se determinan cíclicamente como:

$$y_0^{(k)} = y_N^{(k)}, \quad y_{N+1}^{(k)} = y_1^{(k)}, \quad y_{N+2}^{(k)} = y_2^{(k)}$$

Se supone que la función se evaluará repetidamente durante un cierto número de iteraciones y en cada iteración el vector de salida de la iteración anterior será el vector de entrada en la iteración actual.

a) Establecer la descomposición de tareas así como la estructura de comunicación y las operaciones de comunicación necesarias para coordinar las tareas. Se puede asumir que el reparto inicial del vector de entrada entre las tareas está ya hecho y que no es necesario recolectar el vector de salida en la última iteración.

b) Definir una estrategia de asignación eficiente sobre 4 y 8 procesadores. Cómo se distribuyen el vector de entrada y el vector solución entre los procesadores con la solución que se propone.

Programación Paralela
4º de Grado en Ingeniería Informática.

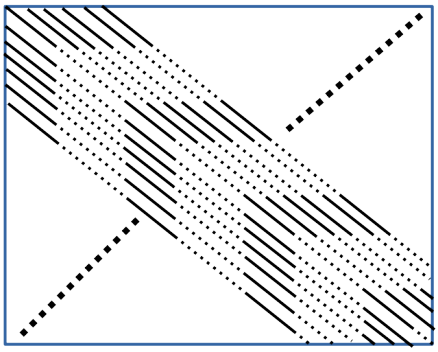
4. Se desea paralelizar un programa secuencial que procesa una malla bidimensional **X** de 1000x1000 puntos (una matriz), actualizándola de acuerdo al siguiente esquema iterativo:

$$X_{i,j}^{t+1} = \frac{4X_{i,j}^{t} + X_{i+1,j+1}^{t} + X_{i+1,j-1}^{t} + X_{i-1,j+1}^{t} + X_{i-1,j-1}^{t}}{20}$$

Se asume que el valor de $X_{i,j}$ cuando los índices i, j están fuera de rango en la fórmula es el siguiente:

$X_{i,0}=X_{i,N}$, $X_{i,N+1}=X_{i,1}$, $X_{0,j}=X_{N,j}$, $X_{N+1,j}=X_{1,j}$, para i,j=1,…,N.

Además, en cada iteración se calcula el máximo de los puntos situados en cada diagonal para chequear convergencia. Se asume que las diagonales se consideran de izquierda a derecha de la malla, como se muestra en la siguiente figura:



Inicialmente, el problema se descompone en una tarea por punto de la malla.

- Establecer una estrategia de asignación adecuadas para resolver el problema sobre 5 procesadores, justificando las decisiones tomadas.

5. Se desea resolver en paralelo un problema de exploración exhaustiva de un árbol de búsqueda. Se supone la existencia de las siguientes funciones:

- Función *Ramifica(nodo, hijo1,hijo2)* para generar dos nuevos nodos del árbol, *hijo1* e *hijo2* que cuelgan de *nodo* en el árbol.

- Función lógica *Solucion(nodo,peso)* que devuelve *Verdadero* si un nodo del árbol es una solución y, *Falso* en caso contrario. Si el nodo es solución, también devuelve un valor entero peso.

El objetivo es obtener la suma de los pesos de todos los nodos solución del árbol.
Estudiar las opciones más apropiadas para aglomeración y asignación del algoritmo dependiendo del número de procesadores *P* y del costo de evaluación de las funciones Ramifica y Solucion.

6. Supongamos que deseamos derivar un programa paralelo para ordenar ***N*** enteros $N=2^k$, $k>20$ sobre una máquina de memoria distribuida con ***P=8*** procesadores, teniendo en cuenta que el resultado de la ordenación debe replicarse en todos los procesadores. Supongamos también que en la máquina destino se dispone de un procedimiento secuencial ***mezcla(lista1,lista2, listamezclada)*** que permite mezclar eficientemente dos secuencias ordenadas de enteros de cualquier longitud y de un procedimiento secuencial ***ordena(lista)*** que permite ordenar una secuencia de enteros arbitraria.

Diseñar el programa paralelo de ordenación, justificando de forma concisa las decisiones tomadas. Suponer que la descomposición del problema en tareas ya ha sido realizado dando lugar a una tarea por cada par de elementos disjuntos de la lista a ordenar. Cada una de las tareas se encarga de ordenar el par que se le ha asignado.

- Establecer la estructura de comunicación y las operaciones de comunicación necesarias para coordinar las tareas.
- Definir una estrategia de asignación eficiente sobre los 8 procesadores.

7. Disponemos de **9** tareas, **T1, T2, ..., T9** que deben ser ejecutadas sobre **3** procesadores idénticos (**P0, P1 y P2**). Los tiempos de ejecución de cada tarea vienen dados en la siguiente tabla:

Tarea	T1	T2	T3	T4	T5	T6	T7	T8	T9
Coste	3	4	5	6	7	7	8	8	10

Se ha diseñado un algoritmo paralelo que utiliza un esquema de asignación dinámica centralizada tipo maestro-esclavo para planificar la ejecución de las tareas en los tres procesadores, donde el maestro va asignando una tarea a cada procesador como respuesta a la petición de un procesador. Adicionalmente se asume lo siguiente:

a) El procesador P0 es el maestro, pero también hace el papel de esclavo, en el sentido de que también ejecuta tareas.

b) El coste recibir peticiones de procesadores en P0, así como de seleccionar y encargar la realización de una tarea a un procesador (ya sea P0 mismo, P1 ó P2) no conlleva ningún coste en P0. Por tanto, P0 trabaja como si fuera un esclavo más pero, al mismo tiempo y sin que afecte para nada a su labor de esclavo, asigna tareas a todos los procesadores (incluyéndose a sí mismo) según el enfoque maestro-esclavo.

c) La selección de la siguiente tarea a asignar a un procesador que demanda trabajo se hace de forma aleatoria por parte de P0.

Calcular la ganacia en velocidad (speedup) que se obtendría en el peor caso y en el mejor caso bajo estas suposiciones. Mostrar uno de los mejores casos y uno de los peores casos indicando qué tareas ejecutaría cada proceso y en qué orden.

Programación Paralela
4º de Grado en Ingeniería Informática.

8. Se pretende paralelizar un algoritmo iterativo que actúa sobre una matriz cuadrada **A=(aij)** con 1000 x 1000 enteros usando un modelo programación basado en paso de mensajes (distribuido). Inicialmente se asume que la matriz A tiene un valor inicial $\mathbf{A^{(0)}}$ (sería el valor de A para la iteración inicial, t=0) y se desea realizar la actualización de A en cada iteración (es decir, pasar del valor en la iteración t al valor en t+1). La actualización de cada elemento aij de A en la iteración t+1 se lleva acabo a partir de los valores de A en la iteración anterior t, usando la siguiente expresión:

Si (t+1 es par)	Si (t+1 es impar)
Si (i+j es par) $a_{i,j}^{(t+1)} = a_{i,j}^{(t)} - 2\,a_{i+1,j}^{(t)} - a_{i,j+1}^{(t)} - a_{i,j-1}^{(t)} + 2\,a_{i-1,j}^{(t)}$ en caso contrario $a_{i,j}^{(t+1)} = a_{i,j}^{(t)}$	Si (i+j es impar) $a_{i,j}^{(t+1)} = a_{i,j}^{(t)} - 2\,a_{i+1,j}^{(t)} - a_{i,j+1}^{(t)} - a_{i,j-1}^{(t)} + 2\,a_{i-1,j}^{(t)}$ en caso contrario $a_{i,j}^{(t+1)} = a_{i,j}^{(t)}$

En definitiva, mientras en una iteración se utiliza una fórmula de diferencias finitas que sólo se aplica a los elementos que ocupan posiciones pares (a_{ij}, con i+j par), en la siguiente iteración, la fórmula se utiliza sólo para actualizar aquellas entradas con posición impar (i+j impar). Sería algo similar a un tablero de ajedrez en el que primero se actualizan las casillas negras, después las blancas, y así sucesivamente. La fórmula de diferencias finitas se aplica a cada elemento a_{ij} de A mientras a_{ij} cumpla una condición que tiene un costo computacional constante.

Se asume que en la fase de descomposición se asigna una tarea a la gestión de cada elemento de A.

a) Establecer qué distribución de la matriz A sobre 4 procesos resulta más apropiada cuando se conoce de antemano que todos los elementos de A requieren el mismo número de iteraciones para converger al resultado. Describir el código de cada proceso en estas circunstancias.

b) Establecer la distribución más apropiada sobre 4 procesos cuando se sabe que el número de iteraciones que requiere cada elemento de A no es el mismo para todos los elementos de A, sino que es proporcional al número de columna que ocupa. Justificar la respuesta con concisión.

9. Se pretende derivar un programa paralelo para encontrar de forma eficiente el primer registro que cumpla una determinada condición en una lista de registros de pequeño tamaño comenzando por el principio. Como resultado de realizar la descomposición en tareas del problema, se ha obtenido el algoritmo abstracto que se esboza a continuación:

- Se sigue un esquema centralizado, con una tarea maestro y N tareas trabajadoras.

- Una tarea maestro gestiona la lista y va asignando registros individuales a un conjunto de N tareas trabajadoras, comenzando por el principio de la lista. Inicialmente, se asignan N registros a las N tareas trabajadoras (uno por tarea). El resto de registros se van asignando conforme cada tarea trabajadora solicita un nuevo registro, pero siempre se asignan registros partiendo del extremo inferior de la lista que queda por procesar. Cuando una tarea trabajadora encuentra un registro que cumple la condición, el maestro envía un mensaje de terminación al resto de trabajadores, conforme le van llegando las nuevas solicitudes de los mismos. Naturalmente, si alguno de los trabajadores restantes envía un mensaje indicando que el registro procesado cumple la condición, no se le envía ningún mensaje de terminación a dicha tarea trabajadora y se actualiza el resultado si este registro estuviera en una posición anterior al previamente encontrado.

- Las tareas trabajadoras chequean continuamente registros de la lista hasta que reciben de la tarea maestro un mensaje de terminación. Concretamente realizan las siguientes acciones:

```
Recibe mensaje de maestro
    Mientras mensaje ≠ terminación
        Registro=mensaje
        Chequear condición
        Si el registro cumple la condición, envía mensaje al maestro y termina
    Recibe mensaje de maestro
```

a) Estudiar cómo se podría realizar la asignación sobre P=8 procesadores, asumiendo que la condición a chequear es muy costosa computacionalmente. Describir el código de los procesos resultantes incluyendo mecanismos para equilibrado de carga y terminación elegante de las tareas.

b) Describir brevemente cómo se podría mejorar la estrategia de asignación si se dispusiera de P=100 procesadores y la condición a chequear no fuera demasiado costosa.

10. Se desea encontrar todas las ocurrencias de una subcadena particular con m caracteres, llamado **patrón**, dentro de otra cadena, llamada **texto** con N caracteres (N>>m y N es múltiplo de m). Tanto el patrón como el texto se almacenan internamente como arrays de caracteres. Describe un algoritmo paralelo para resolver el problema.

- Establecer la estructura de comunicación y las operaciones de comunicación necesarias para coordinar las tareas, suponiendo suponiendo que el texto se reparte inicialmente por bloques de caracteres consecutivos entre las tareas.

- Establecer una estrategia de aglomeración y asignación eficiente sobre 8 procesadores para dos casos diferentes:
  - Todos los procesadores son del mismo tipo y potencia .
  - Los procesadores son de tipos y potencias muy dispares pero desconocidos a priori.

11. Se plantea el problema de determinar el número de palabras que contienen letras únicas en una lista de 10000 palabras, es decir, determinar el número de palabras en las que ninguna letra aparece más de una vez.
El acceso a una palabra se hace mediante un enorme array C de N caracteres (N>>10000) donde las palabras se almacenan de forma consecutiva y cada palabra se separa de la anterior por el carácter especial @. El acceso a la lista es mucho más rápido en término medio si se accede a palabras en posiciones consecutivas de la lista. La longitud de cada palabra se conoce después de explorarla mediante una función C que determina si la siguiente palabra de un array de caracteres (donde las palabras se separan con @) contiene letras únicas y cuál es su tamaño.

Describir un algoritmo paralelo de paso de mensajes para este problema usando un esquema centralizado de equilibrado de carga (maestro-esclavo). Al final del ćomputo, el proceso maestro debería imprimir el número de palabras que contienen letras únicas y también aquellas palabras con letras únicas que tienen la mayor longitud encontrada. Describir de forma resumida pero clara, qué acciones debería realizar el proceso maestro y cada proceso esclavo asumiendo que la lista de palabras se encuentra inicialmente en la memoria del proceso maestro.

12. Considérese una matriz bidimensional cuadrada con 1000 **x** 1000 enteros que se reparte entre 10 procesadores siguiendo una distribución cíclica por bloques. Especifica los 4 parámetros (P_x, P_y, mb_x, mb_y) que definen la distribución particular de la matriz entre los 10 procesadores en cada uno de estos casos**:**

a) Distribución cíclica por columnas.
b) Distribución cíclica por filas.
c) Distribución cíclica por bloques de columnas con tamaño de bloque 5.
d) Distribución cíclica por bloques de filas con tamaño de bloque 5.