ExamenFeb19final.pdf

TTronc0

Análisis y diseño de algoritmos

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

E. T. S. de Ingeniería Informática | Curso: 2018–19

Análisis y Diseño de Algoritmos | 31 de enero

Alumno: ______________________________

Grupo: ______________________________

Bloque I: Complejidad, Especificación y Divide y Vencerás

1. **[3p]** Proporcionar una especificación pre/post del siguiente procedimiento:

   func $generatePairs$ ($\downarrow A[1 \ldots i], B[1 \ldots j]$: $\mathbb{N}$): Set$\langle \mathbb{N} \times \mathbb{N} \rangle$

   Este procedimiento toma como argumentos dos arrays de números naturales (sin elementos en común), y devuelve un conjunto con todos los pares de elementos formados por un número de un array y un número de otro. Por ejemplo, $generatePairs([1,2],[4]) = \{(1,4),(2,4),(4,1),(4,2)\}$.

2. **[7p]** Supongamos que disponemos de una implementación del procedimiento anterior.

   a) **[5p]** Diseñar un procedimiento
   $\{\forall i,j \in \{1 \ldots n\}, i \neq j : L_i \neq L_j\}$
   func $generateAllPairs$ ($\downarrow L[1 \ldots n]$: $\mathbb{N}$): Set$\langle \mathbb{N} \times \mathbb{N} \rangle$
   $\{\forall i,j \in \{1 \ldots n\}, i \neq j : (L_i, L_j) \in generateAllPairs(L)\}$
   mediante la estrategia de Divide y Vencerás, que genere un conjunto con todos los pares de elementos distintos de L.

   b) **[2p]** Utilizar el teorema maestro para estimar la complejidad del anterior algoritmo, suponiendo que las llamadas a la función $generatePairs(\cdot)$ tienen complejidad $\Theta(1)$.

Bloque II: Programación Dinámica

Dado un vector de enteros $[a_1, \cdots, a_n]$, definimos su *sumaducto* como el mayor valor que puede conseguirse sumando elementos y/o productos de dos elementos adyacentes. Cada elemento sólo puede en una suma o en un producto. Por ejemplo, si el vector es $\langle 1,3,4,3,2,2,5,3 \rangle$, su sumaducto sería $36 = 1 + (3 \cdot 4) + (3 \cdot 2) + 2 + (5 \cdot 3)$.

1. **[2p]** Mostrar que el problema exhibe la propiedad de subestructura óptima.
2. **[4p]** Proporcionar la ecuación de Bellman para el sumaducto.
3. **[3p]** Implementar un algoritmo de programación dinámica *bottom-up* según la ecuación anterior.
4. **[1p]** Indicar la complejidad en tiempo y espacio del algoritmo anterior.

Bloque III: Ramificación y Poda

Dado un tablero de 3×3 con 8 fichas (numeradas del 1 al 8) y un espacio vacío ya situados; el objetivo es colocar los números en las casillas para que coincidan con la configuración final utilizando el espacio vacío.

1	2	3
5	6	
7	8	4

conf. inicial

1	2	3
5		6
7	8	4

conf. final

En cada movimiento podemos deslizar una de las cuatro casillas adyacentes (izquierda, derecha, arriba y abajo) en el espacio vacío. El objetivo de nuestro problema es alcanzar un estado final dado con el menor número de movimientos. En el ejemplo mostrado, los tres movimientos posibles inicialmente son mover la ficha 3, la ficha 6 o la ficha 4 al espacio vacío produciéndose una nueva configuración.

1. **[6p]** Hacer una descripción detallada de los elementos de la solución. Establecer cuál es la estructura solución, restricciones, proceso de ramificación, y definir claramente la función de estimación para cada nodo y su complejidad (indicar al menos un ejemplo del cálculo de la cota).
2. **[4p]** Expandir el árbol con la configuración inicial del ejemplo, indicando, claramente, el orden de generación de los nodos y las cotas utilizadas y/o calculadas para llegar a la configuración final.