Preguntas de Conceptos

Jeremy Barbay

17 October 2011

Contents

1 PREGUNTAS

1.1 Cota superior de (la complejidad de) Max ord

Dado un arreglo ordenado de n enteros, en cuanto accessos al arreglo pueden calcular su valor maximal?

1. □ 0
2. □ 1
3. □ $n-1$
4. □ n
5. □ otra

1.2 Definicion de la mediana

Dado un arreglo de n enteros, cual es la definicion correcta de la mediana?

1. □ El promedio de las valores minima y maxima del arreglo.
2. □ La valor en el centro del arreglo.
3. □ La valor en el centro del arreglo ordenado.
4. □ La valor superior a $\lceil (n-1)/2 \rceil$ valores y inferior a $\lfloor (n-1)/2 \rfloor$ valores.
5. □ otra respuesta.

1.3 Dificultad de problemas en arreglos

Dado un arreglo de n enteros, cual problema requiere mas accessos al arreglo? Mas computacion?

1. □ Calcular la valor minima
2. □ Calcular la valor maxima
3. □ Calcular la valor mediana
4. □ Calcular la valor promedia
5. □ Son todos iguales

1.4 Cota Inferior para Max

Dado un arreglo de n enteros, cuanto comparaciones entre los elementos del arreglo se necessitan para calcular su valor maximal?

1. ☐ 0
2. ☐ 1
3. ☐ $n-1$
4. ☐ n
5. ☐ otra respuesta

1.5 Definicion del problema de MinMax

Dado un arreglo A de n enteros, cual es la definicion del problema de "minmax"?

1. ☐ calcular $\min_{i\in[1..n],j\in[i..n]} A[i]$
2. ☐ calcular $\min_{i\in[1..n]} \max j \in [i..n] A[i]$
3. ☐ calcular $(\min_{i\in[1..n]} A[i], \max_{i\in[1..n]} A[i])$
4. ☐ calcular $(\min_{i\in[1..n]} A[i], \max_{j\in[1..n]} A[j])$
5. ☐ otra respuesta

1.6 Cotas de (la complejidad de) problemas combinados

Dado dos problemas A y B (e.g. min y max), cada uno con un algoritmo que le resuelve optimalemente con complejidad $f_A(n)$ y $f_B(n)$, cual es la complejidad del problema AB (e.g. min max)?

1. ☐ min{$f_A(n)$, f$_B$(n)}$
2. ☐ $f_A(n)$ + f$_B$(n)$
3. ☐ ($f_A(n)$ + f$_B$(n))/2$
4. ☐ max{$f_A(n)$, f$_B$(n)}$
5. ☐ otra respuesta

1.7 Cota superior de (la complejidad de) Min Max

Dado un arreglo de n enteros, en cuanto comparaciones (cantidad exacta, no asimptotica) entre los elementos del arreglo pueden calcular su valor maximal y minimal?

1. □ $n-1$
2. □ $3n/2-2$ si n es par, $3n/2+1/2$ si n es impar.
3. □ $(n-1)+(n-2)$
4. □ $2(n-1)$
5. □ otra respuesta

1.8 Cota inferior de (la complejidad de) Min Max

Dado un arreglo de n enteros, cuanto comparaciones (cantidad exacta, no asimptotica) entre los elementos del arreglo se necessitan para calcular su valor maximal y minimal?

1. □ $n-1$
2. □ $\lceil 3n/2 \rceil - 2$
3. □ $(n-1)+(n-2)$
4. □ $2(n-1)$
5. □ otra respuesta

1.9 Juego de las preguntas, $n=4$

Cuanta preguntas (e.g. "$x<4$?", "x=2"?) se necesitan para adivinar un entero entre 1 y $4 (i.e. $x \in [1..4]$)?

1. □ 1
2. □ 2
3. □ 3
4. □ 4
5. □ otra

1.10 Juego de las preguntas, $n = 1024$

Cuanta preguntas (e.g. "$x < 10$?", "x=10"?) se necesitan para adivinar un entero entre 1 y 1024?

1. □ 8
2. □ 9
3. □ 10
4. □ 11
5. □ otra

1.11 Codificacion de un simbolo

Dado 1 simbolo elegido a dentro de $[1..\sigma]$

1. □ no se puede codificar **nunca** en $o(\lg \sigma)$ bits
2. □ no se puede codificar **siempre** en $o(\lg \sigma)$ bits
3. □ no se sabe **como codificar siempre** en $o(\lg \sigma)$ bits
4. □ no se sabe **si nunca se puede codificar** en $o(\lg \sigma)$ bits
5. □ otra

1.12 Definicion de un arbol de decision

Un arbol de decision es definido como un arbol

1. □ modelisando algoritmos en el modelo de comparacion.
2. □ binario donde cada hoja identifica una instancia.
3. □ binario donde cada nodo prueba una caracteristica de la instancia.
4. □ un arbol de grado finito donde cada hoja indica una decision sobre la instancia.
5. □ otra.

1.13 Codificacion de n simbolos

Dado n simbolos elegido a dentro de un alfabeto de tamaño σ

1. □ no se puede codificar **nunca** en $o(n \lg \sigma)$ bits
2. □ no se puede codificar **siempre** en $o(n \lg \sigma)$ bits
3. □ no se sabe **como codificar siempre** en $o(n \lg \sigma)$ bits
4. □ no se sabe **si nunca se puede codificar** en $o(n \lg \sigma)$ bits
5. □ otra

1.14 Definicion de "InsertionRank"

Dado un arreglo ordenado $A[1..n]$ de n valores y una valor x, cual(es) de estas definiciones del *Posicion de Insercion* ("Insertion Rank") de x en A son incorectas? ($A[1] = -\infty$ y $A[n+1] = +\infty$)

1. □ la posicion en cual x deberia ser insertado por dejar A ordenado
2. □ el entero $p \in [1..n+1]$ tal que $A[p-1] < x \leq A[p]$
3. □ el entero $p \in [0..n]$ tal que $A[p] \leq x < A[p+1]$
4. □ el entero $p \in [1..n]$ tal que $x = A[p]$
5. □ ningunos o mas que dos

1.15 Dos tipos de busqueda ordenada

Dado el codigo siguente, cual es la mejor manera de completarlo para minimizar la complejidad (non asymptotica) en el peor caso? El el caso promedio?
insertionRank(x,A,l,r) { if($r - l < 2$) return l else { m=(l+r)/2; ... } }

1. □ if($x < A[m]$) return insertionRank(x,A,l,m) else if($x > A[m]$) return insertionRank(x,A,m,r) else if($x = A[m]$) return m endif
2. □ if($x = A[m]$) return m else if($x < A[m]$) return insertionRank(x,A,l,m) else if($x > A[m]$) return insertionRank(x,A,m,r) endif
3. □ if($x = A[m]$) return m else if($x < A[m]$) return insertionRank(x,A,l,m) else return insertionRank(x,A,m,r) endif
4. □ if($x < A[m]$) return insertionRank(x,A,l,m) else return insertionRank(x,A,m,r) endif
5. □ performan iguales todos en el peor caso.

1.16 Cota inferior por busqueda ordenada $n = 1024$.

Dado un arreglo ordenado A de 1024 enteros y un entero x, cuanto comparaciones con elementos del arreglo son necesarias para decidir si x pertenece a A (en el peor caso)?

1. □ 9
2. □ 10
3. □ 11
4. □ 1024
5. □ otra

1.17 Cota inferior por busqueda ordenada general n.

Dado un arreglo ordenado A de n enteros y un entero x, cuanto comparaciones con elementos del arreglo son necesarias para decidir si x pertenece a A (en el peor caso)?

1. ☐ $\lceil \lg n \rceil$
2. ☐ $1 + \lceil \lg n \rceil$
3. ☐ $n - 1$
4. ☐ n
5. ☐ otra

1.18 Definicion del modelo de comparacion

Cuales de estos algoritmos simples son en el modelo de comparacion?

1. ☐ c=0; for(int i=1; i<n; i++) { if(A[i]>A[i+1]) c++;}
2. ☐ for(int i=1; i<n; i++) { if(A[i]>A[i+1]) print i;}

; 3. [ ] for(int i=1; i<n; i++) { if(A[i]>A[i+1]) print i;} ; 4. [ ] for(int i=1; i<n; i++) { if(A[i]>A[i+1]) print i;}

1. ☐ ningunos

1.19 Relacion entre codificacion y busqueda

Cual de estas aserciones es falsa en el modelo de comparacion?

1. ☐ A cada algoritmo de busqueda corresponde una codificacion de enteros.
2. ☐ A cada codificacion de enteros corresponde un algoritmo de busqueda.
3. ☐ A algunos algoritmos de busqueda corresponde una codificacion de enteros
4. ☐ A algunas codificaciones de enteros corresponde un algoritmo de busqueda.
5. ☐ otra

1.20 Busqueda Doblada

:PROOF:

1. □ $\lg(1+n)$ comparaciones -> Busqueda binaria
2. □ $p+1$ comparaciones -> Busqueda secuencial
3. □ $2\lg p$ comparaciones -> Busqueda dublada
4. □ $2\lg(n-p)$ comparaciones -> Busqueda dublada, iniciando a la derecha

END: Cual de las asercions siguentes son falsas? Dado una valor x y un arreglo ordenado A de n valores, existe un algoritmo calculando la posicion de inserción p de x en A en

1. □ $\lg(1+n)$ comparaciones
2. □ $p+1$ comparaciones
3. □ $2\lg p$ comparaciones
4. □ $2\lg(n-p)$ comparaciones
5. □ ningunas o mas que dos.

1.21 Compression de enteros

Dado un entero $x \in [1..n]$, existe un esquema de codificacion representando x con

1. □ $\lg n$ bits,
2. □ $2\lg p$ bits,
3. □ p bits,
4. □ $2\lg(n-p)$ bits,
5. □ ningunas o mas que dos.

1.22 Cota inferior ordenamiento (en el modelo de comparacion)

Decir que "Ordenar es en $\Omega(n\lg n)$ (en el modelo de comparacion) significa que

1. □ no se puede ordenar en $o(n\lg n)$ comparaciones
2. □ ninguno algoritmo conocido (del modelo de comparacion) ordena en $o(n\lg n)$ comparaciones
3. □ no se puede ordenar en tiempo $o(n\lg n)$
4. □ ninguno algoritmo conocido (del modelo de comparacion) ordena en tiempo $o(n\lg n)$
5. □ otra respuesta

1.23 Tecnicas de cotas inferiores

Cual(es) de las tecnicas siguentes permitten de mostrar cotas inferiores para la complejidad en promedio?

1. ☐ lemma del ave
2. ☐ Estrategia de Adversario
3. ☐ Arbol Binario de Decision
4. ☐ lemma del minimax
5. ☐ ningunas o mas de dos.

[1] FOOTNOTE DEFINITION NOT FOUND: 0