Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC2283 - Diseño y Análisis de Algoritmos

Ayudantía 12 - Algoritmos Aleatorizados

Septiembre de 2025

Nicolás Buzeta, Manuel Villablanca

---

1. Pregunta 1

Para un entero impact $n \geq 1$, considere el siguiente algoritmo Las Vegas estudiado en clases para calcular la mediana de una lista $L[n.,1]$:

CalcularMediana$(L[1..n])$
  if $n < 2001$ **then**
    Mergesort(L)
    return $L[\lceil \frac{n}{2} \rceil]$
  else
    sea R una lista de $\lceil n^{\frac{3}{4}} \rceil$ números enteros escogidos con distribución uniforme y de manera independiente desde L
  Mergesort(R)
  $d := R\left[\left\lfloor \frac{1}{2} \cdot n^{\frac{3}{4}} - n^{\frac{1}{2}} \right\rfloor\right]$
  $u := R\left[\left\lceil \frac{1}{2} \cdot n^{\frac{3}{4}} + n^{\frac{1}{2}} \right\rceil\right]$
  $S := \emptyset$
  $m_d := 0$
  $m_u := 0$
  for $i := 1$ **to** n **do**
    if $d \leq L[i]$ **and** $L[i] \leq u$ **then Append**$(S, L[i])$
    else if $L[i] < d$ **then** $m_d := m_d + 1$
    else $m_u := m_u + 1$
  if $m_d \geq \lceil \frac{n}{2} \rceil$ **or** $m_u \geq \lceil \frac{n}{2} \rceil$ **or** **Length**$(S) > 4 \cdot \lfloor n^{\frac{3}{4}} \rfloor$ **then return** *sin_resultado*
  else
    Mergesort(S)
    return $S[\lceil \frac{n}{2} \rceil - m_d]$

a) Explique por qué las variables d y u son inicializadas de esa manera.

b) ¿Por qué si se cumple $m_d \geq \lceil \frac{n}{2} \rceil$ or $m_u \geq \lceil \frac{n}{2} \rceil$ el algoritmo responde sin resultado?

c) ¿Por qué el algoritmo es considerado de tipo Las Vegas, si en ciertas ocasiones puede fallar y no encontrar la mediana de L?

1.1. Solución

a) Cuando creamos R, lo hicimos haciendo un sampleo uniforme de $n^{\frac{3}{4}}$ de los elementos de n. Ahora si nos fueramos a imaginar que L estuviera ordenado, nuestro elementos de R igual deberia estar distribuidos uniformemente en L.

Como R tiene $n^{\frac{3}{4}}$ elementos, por cada elementos que avanzamos en R, debemos avanzar $\frac{n}{n^{\frac{3}{4}}} = n^{\frac{1}{4}}$ elementos.

Despues de esto, sabemos que la mediana de R esta en posición $\frac{1}{2} \cdot n^{\frac{3}{4}}$.

Ahora necesitamos un margen de error para la mediana verdadera. Esto se define como movernos $n^{\frac{1}{2}}$ posiciones dentro de nuestro arreglo R. Como nos movimos $n^{\frac{1}{2}}$ posiciones en R y sabemos que cada paso corresponde a $n^{\frac{1}{4}}$ pasos en L, podemos decir que la distancia esperada de nuestro margen en L es de $n^{\frac{1}{2}} \cdot n^{\frac{1}{4}} = n^{\frac{3}{4}}$. Por lo que esperamos que hayan $2 \cdot n^{\frac{3}{4}}$ elementos entre d/u en L.

b) Si mas de la mitad de los elementos de L están arriba o abajo de nuestro margen, sabemos que la mediana no puede estar dentro de ella.

c) Es considerado de tipo Las Vegas porque si bien puede fallar, siempre que este falla el algoritmo dice que fallo, por lo que se puede ejecutar varias veces en serie el algoritmo convirtiendose en uno de Las Vegas.

2. Pregunta 2

Dado un arreglo no necesariamente ordenado $A[1..n]$, se dice que éste tiene una mayoría absoluta si al menos $\lfloor n/2 \rfloor + 1$ de los elementos son iguales.

a) Diseñe un algoritmo aleatorizado de tipo Monte Carlo que permita determinar si el arreglo tiene una mayoría absoluta o no (retornando Sí o No según corresponda). El tiempo de ejecución de su algoritmo debería ser $O(n)$ comparaciones de elementos.

b) Haga un análisis de error de su algoritmo, indicando en qué casos falla en entregar una respuesta correcta, y cuál es la probabilidad de fallar.

2.1. Solución

Algorithm 1 Mayoría($A[1..n]$)

```
1: Elegir un índice m de forma uniforme en [1..n]
2: c ← 0
3: for i ← 1 to n do
4:     if A[i] = A[m] then
5:         c ← c + 1
6:     end if
7: end for
8: if c ≥ ⌊n/2⌋ + 1 then
9:     return Sí
10: else
11:     return No
12: end if
```

1) Si el algoritmo retorna **Sí**, siempre es correcto: la única manera de que responda de esta manera es que exista una mayoría.

2) Si el algoritmo retorna **No**:

(i) Si el arreglo A no tiene una mayoría, entonces la respuesta del algoritmo es correcta.

(ii) Si el arreglo A tiene una mayoría, la respuesta del algoritmo es incorrecta.

En conclusión, el algoritmo puede equivocarse cuando responde **No**. Respecto a la probabilidad de error:

Si A tiene una mayoría, la probabilidad de que $A[m]$ no sea la mayoría es

$$\leq \frac{n - \left\lfloor \frac{n}{2} \right\rfloor - 1}{n} \approx \frac{1}{2} - \frac{1}{n}.$$