# COMPLEJIDAD MEDIA DE ALGORITMOS : EJEMPLOS

Estudiamos la complejidad media por considerar que en ocasiones la complejidad en el peor de los casos no resulta demasiado representativa al aparecer unos pocos casos muy malos al lado de una mayoría mucho mejores. Como veremos, ello es particularmente cierto en problemas en los que aparece alguna estructura arborescente cuyo comportamiento sólo es negativo cuando deviene en una estructura lineal degenerada, lo que sólo sucede en contadas ocasiones.

Sin embargo, para comenzar nuestro estudio presentaremos un par de ejemplos sencillos en los que la complejidad media coincide con la del peor de los casos.

## Ordenación por inserción

Consideramos el algoritmo de ordenación de una secuencia de enteros distintos : $a_1, \dots, a_n$ que se basa en ir manteniendo la permutación ordenada $a_{i_1}, \dots, a_{i_j}$ de los $j$ primeros datos. Para lograrlo, tras recibir cada $a_{j+1}$ se inserta ordenadamente en dicha permutación, mediante una búsqueda secuencial de la posición adecuada.

Caso peor : corresponde, curiosamente, al caso en el que los datos nos llegan ordenados. En tal caso la inserción de $a_j$ requiere $j-1$ comparaciones, que hacen un total de $\sum_{j=1}^{n} j-1 = \frac{n(n+1)}{2} \in O(n^2)$

Complejidad media : Supondremos que las $n!$ permutaciones de los datos son equiprobables. Ello implica que la posición de $a_{j+1}$ al insertarlo en la permutación ordenada de los datos anteriores es una cualquiera entre la 1 y la $j+1$, con probabilidad uniforme $1/j+1$. Exactamente se requieren en media $\sum_{k=1}^{j-1} \frac{1}{j} k + \frac{1}{j}(j-1) = \frac{1}{j}\left(\frac{j(j-1)}{2} + j-1\right) = \frac{1}{j}\,\frac{(j+2)(j-1)}{2} = m_j$ comparaciones para insertar cada $a_j$ con $j \geq 2$.

Entonces $\frac{j-1}{2} < m_j < \frac{j+2}{2}$, de modo que al sumar el coste de todas las inserciones obtenemos $\sum_{j=2}^{n} \frac{j-2}{2} < \text{coste\_medio} < \sum_{j=2}^{n} \frac{j+2}{2}$, de donde concluimos inmediatamente que el coste medio es exactamente del orden de $n^2$ : $\Theta(n^2)$.

Para concluir, observemos que el resultado no es trivial, ya que el coste en el mejor de los casos del algoritmo, que se alcanza cuando los datos vienen completamente invertidos respecto a la ordenación deseada, es lineal. En cambio, si utilizamos un vector, más exactamente su comienzo, para mantener las permutaciones ordenadas, el coste de insertar $a_j$ es lineal en $j$, por lo que el coste del algoritmo es <u>siempre</u> del orden de $n^2$.

## Búsqueda binaria

Consideramos el algoritmo de búsqueda binaria, o por bipartición, en un vector ordenado sin elementos repetidos. Suponemos que buscamos uno de los elementos del mismo.

Caso peor: corresponde al caso en el que no encontramos el elemento hasta que hemos reducido el subvector sobre el que se busca a un subvector trivial con un único elemento. Como en cada iteración se divide por dos el tamaño del subvector, obtenemos un máximo de $\log_2(n)$ iteraciones.

Caso mejor: corresponde a la búsqueda del elemento central del vector, que se encuentre en la primera iteración, o sea con coste constante $O(1)$.

Complejidad media: Observamos que el proceso de búsqueda conjunto de la totalidad de los elementos del vector se corresponde con la estructura del árbol de búsqueda de mínima profundidad que alberga sus elementos. Resulta inmediato que la profundidad de dicho árbol es $\log_2(n)$ y que más de la mitad de sus elementos se encuentran en sus dos últimos niveles. En consecuencia, con probabilidad mayor que $1/2$ tendremos un coste $\log_2(n) - 1$, por lo que el coste medio es al menos $O(\log_2(n))$, coincidiendo pues con el coste en el caso peor.

Observación: A pesar de que el tipo de datos vector es un tipo lineal, al tratarse de un vector ordenado el acceso aleatorio permite que se comporte como si de una estructura arborescente. Se trata además de un árbol de mínima profundidad que hace que el caso peor sea ya logarítmico, haciendo ya muy difícil que el comportamiento en media sea mejor, lo que en efecto no sucede.

Abordamos ahora el estudio de los árboles binarios de búsqueda. Comenzaremos viendo que el coste medio de su construcción con inserciones sucesivas es $O(n \log n)$, siendo $n$ el nº de elementos insertados.

### Construcción de un árbol de búsqueda

Si contabilizamos sólo las comparaciones entre elementos, el número de ellas que necesitamos para insertar un elemento en un árbol de búsqueda coincide con la profundidad en la que finalmente lo insertamos, cuando comenzamos considerando que la profundidad del nodo raíz es 0.

El valor total de dichas profundidades se corresponde con la longitud de caminos internos $I(T)$ definida mediante

$$I(\text{crearbol}) = 0$$

$$I(T = \text{montarbol}(r, hi, hd)) = |T| - 1 + I(hi) + I(hd)$$

La media de dichos valores sobre árboles construidos a partir de una permutación aleatoria de $n$ elementos, la denotaremos por $I_{av}(n)$ que cumple $I_{av}(n) = n - 1 + \frac{1}{n} \sum_{i=0}^{n-1} I_{av}(i) + I_{av}(n-1-i)$, ya que $|hi|$ variará uniformente entre $0$ y $n-1$.

Partimos de $I_{av}(0) = 0$ y si computamos $n\, I_{av}(n) - (n-1) I_{av}(n-1)$ obtenemos $2n - 2 + 2\, I_{av}(n-1)$, con lo que, despejando obtenemos

$$\frac{I_{av}(n)}{n+1} - \frac{I_{av}(n-1)}{n} = \frac{2(n-1)}{n(n+1)}$$

Si ahora tomamos $g(n) = \frac{I_{av}(n)}{n+1}$, tenemos $g(n) - g(n-1) = \frac{2(n-1)}{n(n+1)}$ lo que nos conduce a $g(n) = \sum_{i=1}^{n} 2\frac{i-1}{i(i+1)}$.

Pero como $\frac{1}{i+3} < \frac{i-1}{i(i+1)} < \frac{1}{i+2}$, concluimos que asintóticamente $g(n)$ tiene exactamente el mismo comportamiento que la serie armónica $\sum \frac{1}{i}$. Luego $g(n) \in \textcircled{H}(\log n)$, de modo que $I_{av}(n) \in \textcircled{H}(n \log n)$.

Si ahora nos preguntamos por el tiempo medio que tardaría una ulterior búsqueda de un elemento en el árbol, tenemos que coincidiría con el valor medio de la profundidad de un nodo del mismo. O sea es $I_{av}(n)/n$, y por tanto del $\textcircled{H}(\log n)$.

Si en cambio buscamos un elemento que podría no estar en el árbol, suponiendo equiprobables los distintos intervalos inducidos por los elementos en el árbol, tenemos que el coste medio se correspondería con la longitud media de los caminos externos hasta los nodos fallo o hojas del árbol expandido asociado a $T$.

Por inducción se demuestra fácilmente que $E(T) = I(T) + 2|T|$ con lo que $E_{av}(n) = I_{av}(n) + 2$ y así las búsquedas infructuosas también tendrían coste medio $\textcircled{H}(\log n)$.

Y otro tanto sucedería con el coste medio de una ulterior inserción, pues su coste coincide con el de la localización del nodo expandido al que sustituirá el nodo creado por la inserción.

Como colofón estudiamos un problema más complicado: el cálculo de la profundidad media de un árbol de búsqueda.

## Profundidad media de un árbol de búsqueda

Curiosamente este problema que podría parecer similar a los últimos estudiados en el punto anterior, pues se refiere a una propiedad del árbol construido tras una serie de inserciones aleatorias, es sin embargo más complejo. Incluso ello parece contradictorio pues las propiedades que estudiamos antes se refieren al coste de una operación ulterior que hay que considerar aleatoria, mientras que la ahora considerada es una simple propiedad estática del árbol aleatorio construido. Lo que justifica esta aparente contradicción es que la profundidad es una propiedad local que no depende directamente de la totalidad de los nodos del árbol, sino sólo de aquéllos que se encuentren a máxima profundidad. Y como quiera que evaluamos complejidades medias, aquellas propiedades en las que participen de forma uniforme todos los elementos de la estructura evaluada resultan más fáciles de evaluar que aquéllas en las que las contribuciones no son uniformes.

Veremos dos formas distintas de demostrar que la profundidad media de un árbol de búsqueda aleatorio es logarítmica. En la primera conseguiremos reducir nuestro estudio al de una propiedad en la que sí contribuyen de forma análoga todos los nodos del árbol; en la segunda conseguiremos realizar una evaluación directa en términos recurrentes gracias al manejo de unos ingeniosos, aunque sencillos, resultados combinatorios.

### 1ª demostración

La propiedad uniforme que evaluaremos es la profundidad en el árbol de cada elemento $k_\ell$ insertado en ese lugar en el mismo. En concreto, veremos que es muy baja la probabilidad de que ningún nodo esté a gran profundidad, lo que claramente es condición necesaria para que la profundidad del árbol construido sea grande, en concreto más que logarítmica.

Estudiamos pues la profundidad $d(k_j, T)$ a la que se encontrará el $k$-ésimo elemento insertado en un árbol de búsqueda aleatorio. Al efecto estudiamos el conjunto de elementos sobre el camino desde la raíz del árbol hasta la posición donde se inserte $k_j$. Desglosamos dicho

conjunto en dos subconjuntos : $G_j$, los mayores que $k_j$ en dicho camino y $L_j$, formado por los elementos menores que $k_j$.

A continuación estudiamos la propiedad que debe cumplir cada $k_i > k_j$ para formar parte de $G_j$. Ante todo, por supuesto, ha de insertarse antes que $k_j$, o sea tendremos $i<j$. Además, si $k_i \in G_j$, tendremos que si tras insertar $k_i$ insertaramos inmediatamente $k_j$, éste se ubicaría como hijo izquierdo del nodo que ocupa $k_i$. Ello implica que en el recorrido en orden simetrico del árbol resultante $k_j$ aparecería justo tras $k_j$, lo que supone que $k_i$ sería el menor de los elementos $k_\ell$ con $\ell \le i$ tales que $k_\ell > k_j$. En definitiva :

$$G_j = \{ k_i : i<j \wedge k_i > k_j \wedge \forall \ell<i \ ( k_\ell > k_j \Rightarrow k_\ell > k_i ) \}$$

Análogamente,

$$L_j = \{ k_i : i<j \wedge k_i < k_j \wedge \forall \ell<i \ ( k_\ell < k_j \Rightarrow k_\ell < k_i ) \}$$

Ahora, de cara a evaluar $|G_j|$, $|L_j|$, y por tanto $d(k_j,T) = |G_j| + |L_j|$, consideramos un conjunto más sencillo que estos, generado tambien a partir de permutaciones aleatorias. En concreto, evaluaremos el cardinal del conjunto

$$S = \{ i \mid k_i = \min_{\ell \le i} \{ k_\ell \} \}$$

O sea, el numero de veces que cambia el mínimo del prefijo de una tal permutación aleatoria.

Como quiera que $k_1, \dots, k_i$ forman a su vez una permutación aleatoria, la probabilidad de que $k_i$ sea el mínimo de todos ellos es $1/i$. Esta es por tanto la probabilidad de que $i \in S$, con lo que

$$\text{media}(|S_n|) = \sum_{i=1}^{n} 1/i = H_n$$

Se trata de la serie armónica de la que sabemos que $H_n = \ln n + O(1)$, de modo que $H_n \in O(\log n)$ y $H_n > \ln n$.

Si reexaminamos la forma en que se genera $S$ y con ello $|S|$, podemos ver cada aportación a $S$ como el resultado de la etapa i-esima de una distribución de Bernoulli con $n$ etapas en cada una de las cuales la probabilidad de éxito es $1/i$. Esta distribución es muy conocida, dada su importancia y relativa simplicidad. En particular, sabemos de ella que está fuertemente concentrada entorno a su media, siendo muy pequeña la probabilidad de que se alcancen valores mucho mayores que ella. En concreto, $\forall \beta > 1$ $\text{prob}(|S| - H_n \ge \beta H_n) < \left( \frac{e H_n}{\beta H_n} \right)^{\beta H_n}$. Si en particular tomamos $\beta$ con $\beta = \frac{2}{\ln \beta - 1}$, lo cual nos da $\beta \approx 4.32$, obtenemos $\left( \frac{e H_n}{\beta H_n} \right)^{\beta H_n} = e^{(1-\ln\beta)\beta H_n} \le e^{-(\ln\beta - 1)\beta \ln n} = n^{-(\ln\beta - 1)\beta} =$

$= n^{-2} = 1/n^2$ .

Reescribimos el resultado obtenido en la forma $\text{prob}\,(|S| \geq (\beta+1)\,H_n) < 1/n^2$. Veamos que la cota obtenida es transferible de $S$ a $G_j$ y $L_j$. En efecto, si en la definición de $G_j$ eliminamos la condición $k_i > k_j$ y relajamos la condición $i<j$ convirtiéndola en $i<n$, obtenemos la definición de un conjunto al que es más fácil pertenecer, que tendrá por tanto mayor cardinal. La definición del conjunto obtenido coincide con la de $S$ con lo que en particular

$$\forall\, t \in \mathbb{R}^+ \quad \text{prob}\,(|G_j| \geq t/2) < \text{prob}\,(|S| \geq t/2).$$

Si $d(k_j,T) \geq t$ tendremos que $|G_j| \geq t/2$ o bien $|L_j| \geq t/2$, con lo que

$$\text{prob}\,\{\, d(k_j,T) \geq t\,\} \leq \text{prob}\,(|G_j| \geq t/2) + \text{prob}\,(|L_j| \geq t/2) \leq 2\,\text{prob}\,(|S| \geq t/2)$$

Tomando $t = 2(\beta+1)H_n$, obtenemos

$$\text{prob}\,\{\, d(k_j,T) \geq 2(\beta+1)H_n\,\} \leq 2\,\text{prob}\,(|S| \geq (\beta+1)H_n) \leq 2/n^2.$$

Esta es la baja probabilidad que buscábamos de que cada elemento $k_j$ acabara ubicado a una alta profundidad. Si $\text{prof}(T) \geq 2(\beta+1)H_n$ al menos uno de los $k_j$ debería estar a dicha profundidad. O sea

$$\text{prob}\,(\,\text{prof}(T) \geq 2(\beta+1)H_n) \leq \sum_{j=1}^{n} \text{prob}\,\{\, d(k_j,T) \geq 2(\beta+1)H_n\,\} \leq \sum_{j=1}^{n} 2/n^2 = \frac{2}{n}$$

Podemos interpretar este resultado como que sólo $2/n$ partes del tiempo es mayor que $2(\beta+1)H_n$. Pero en tal caso siempre estará acotada por $n$. Entonces, $\text{media}\,(\text{prof}(T)) \leq (1-\frac{2}{n})\,2(\beta+1)H_n + \frac{2}{n}\,n \in O(\log n)$

2ª demostración

Ahora denotaremos por $X_n$ la variable aleatoria definida por la profundidad de un árbol de búsqueda aleatorio con $n$ nodos diferentes que, sin pérdida de generalidad podemos suponer que son los elementos del intervalo $1..n$. Tomamos entonces $Y_n = 2^{X_n}$ y observamos que si nos condicionamos al hecho de que la raíz de los árboles sea $R_n = i$, tenemos que $Y_n|_{R_n=i} = 1 + \max(X_{i-1}, X_{n-i})$, con lo que $Y_n|_{R_n=i} = 2 \cdot \max(Y_{i-1}, Y_{n-i})$.

Como quiera que $\text{prob}\,\{R_n = i\} = 1/n$ tenemos entonces que

$$\text{media}\,(Y_n) = \sum_{i=1}^{n} 1/n \cdot \text{media}\,(2 \cdot \max(Y_{i-1}, Y_{n-i}))$$

con lo que $\text{media}\,(Y_n) = \frac{2}{n} \sum_{i=1}^{n} \text{media}\,(\max(Y_{i-1}, Y_{n-i})) \leq \frac{2}{n} \sum_{i=1}^{n} \text{media}\,(Y_{i-1}) + \text{media}\,(Y_{n-i})$

De modo que $\text{media}\,(Y_n) \leq \frac{4}{n} \sum_{i=1}^{n-1} \text{media}\,(Y_i)$

Veamos entonces que podemos demostrar por inducción que

$$\text{media}\,(Y_n) \leq \frac{1}{4}\binom{n+3}{3}$$

Al efecto utilizaremos la igualdad $\sum_{i=0}^{n-1} \binom{i+3}{3} = \binom{n+3}{4}$, que de nuevo puede probarse por inducción sobre $n$.

Caso base: $\text{media}[Y_1] = Y_1 = 2^0 = 1 \leq \frac{1}{4}\binom{1+3}{3} = \frac{4}{4} = 1$.

Paso inductivo: $\text{media}[Y_n] \leq \frac{4}{n} \sum_{i=0}^{n-1} \text{media}[Y_i] \underset{\text{h.i.}}{\leq} \frac{4}{n} \sum_{i=0}^{n-1} \frac{1}{4}\binom{i+3}{3} =$

$$= \frac{1}{n}\binom{n+3}{4} = \frac{1}{n}\,\frac{(n+3)!}{4!\,(n-1)!} = \frac{1}{4}\,\frac{(n+3)!}{3!\,n!} = \frac{1}{4}\binom{n+3}{3}$$

En consecuencia, $\text{media}[Y_n] \in O(n^3)$, o sea $\text{media}(2^{X_n}) \in O(n^3)$.

Pero como quiera que $f(x) = 2^x$ es convexa, tenemos que $\text{media}(2^{X_n}) \geq 2^{\text{media}(X_n)}$, y tomando logaritmos concluimos $\text{media}(X_n) \in O(\log n)$.