Introducción a la Algoritmia (I): Eficiencia

aprende.olimpiada-informatica.org/algoritmia-introduccion-1-eficiencia

Un *algoritmo* es una serie de instrucciones y operaciones que permiten resolver un problema. Los algoritmos han sido parte de la condición humana desde la antigüedad: en el sentido más amplio de la palabra, cualquier actividad que suponga seguir unos planes o instrucciones conlleva un algoritmo, pero incluso en el sentido más usual, que se refiere a algoritmos para resolver problemas matemáticos, la invención de procedimientos y reglas para hacer cálculos se remonta a tiempos muy antiguos. Sin embargo, en la actualidad, con el surgimiento de máquinas capaces de seguir instrucciones precisas con gran eficiencia, el campo de la algoritmia ha sufrido una auténtica revolución y el estudio y desarrollo de algoritmos eficaces y eficientes es de vital importancia en muchos aspectos de nuestra vida. La *OIE* es una competición que busca introducir a los estudiantes de secundaria en este importante campo científico mediante el planteamiento de problemas para los que los concursantes tienen que diseñar un algoritmo que los resuelva.

Un aspecto muy importante de los algoritmos es la **eficiencia**. Para muchos problemas, es bastante fácil encontrar un procedimiento que acabe dando la respuesta correcta, pero la verdadera dificultad recae en encontrar un algoritmo que encuentre la respuesta eficientemente. Vamos a exponer un ejemplo muy sencillo:

Ejemplo 1: Búsqueda dicotómica (*Binary Search*)

Ana y Bob están jugando a un juego: Ana piensa en un número entre 1 y 100 y Bob intenta adivinarlo. Bob puede preguntar por un número en concreto, y Ana le responde si ese número es el número que está pensando o si el número es mayor o es menor que el que piensa.

Aquí, obviamente Bob siempre puede adivinar el número en el que piensa Ana: simplemente puede preguntar por el 1, después por el 2, después por el 3, y así sucesivamente hasta que Ana le responda que ha acertado. Este algoritmo es muy sencillo y siempre funciona. Sin embargo, comparémoslo con este otro algoritmo: Bob empieza preguntando por el número 50. Si acierta, ya ha acabado el juego; si Ana le responde que el número que piensa es mayor, el siguiente número que pregunta es el 75; si Ana le responde que el número es menor, el siguiente número que Bob pregunta es el 25. La gracia es que, aprovechando la información que brinda la respuesta de Ana, Bob puede descartar la mitad de los números preguntando de esta forma. Bob repite esta estrategia varias veces, cada vez reduciendo a la mitad el “intervalo de números posibles” hasta que inequívocamente llegará al número que piensa Ana.

Este segundo algoritmo también encuentra la respuesta siempre, pero es mucho más eficiente. Con el primer algoritmo, puede que Bob tenga que hacer hasta 100 preguntas (o hasta 99, si Bob deduce que el número que queda después de haber preguntado por todos los demás es la respuesta) si Ana escoge como número secreto el 100, mientras que con el segundo algoritmo, independientemente de el número que escoja Ana, Bob hace como mucho 6 preguntas. Esta idea de "ir dividiendo por la mitad" es muy frecuente en el área de la algoritmia, y tiene nombre propio: búsqueda dicotómica o búsqueda binaria (en inglés *binary search*).

En la OIE se valora que se diseñen programas eficientes, aunque algoritmos que encuentran la solución correcta sin ser completamente eficientes también reciben una puntuación parcial. Por ejemplo, si el problema anterior fuera un problema de un concurso de la OIE, un algoritmo como el primero, que realizara demasiadas preguntas, podría recibir 10 puntos sobre 100. La mayoría de problemas de la OIE, sin embargo, no miden la eficiencia en términos abstractos de "preguntas realizadas" (aunque algunos sí; en Introducción a la Algoritmia (II) se explican los diferentes tipos de problemas) sino en el uso de recursos de ordenador que se hacen para calcular una solución. Sin embargo, esta distinción técnica no afecta al tipo de ideas que se usan para resolver un problema: la búsqueda dicotómica se puede aplicar, por ejemplo, para minimizar los accesos a memoria de un ordenador que está buscando un elemento en una lista ordenada. En el siguiente manual hablaremos más sobre este tema.

Para acabar, presentamos dos ejemplos de problemas un poco más complicados que el ejemplo inicial que hemos dado. Os animamos a pensarlos un poco antes de leer la solución.

Ejemplo 2: *Egg Drop*

Hay un edificio de 100 plantas. Tienes dos huevos tales que cada uno de ellos se rompe cuando se deja caer desde el piso T o un piso superior pero no se rompe si se deja caer desde un piso inferior a T, donde T es un número entre 1 y 101 (si $T = 101$, los huevos no se rompen) que es desconocido. Una vez un huevo está roto, no se puede volver a lanzar. Diseña un algoritmo que permita determinar T haciendo menos de 20 lanzamientos.

Ejemplo 3: La moneda falsa

Tienes 12 monedas, de las cuales una es falsa y tiene un peso diferente a las demás. Tienes una balanza de dos platos. El objetivo es encontrar la moneda falsa y determinar si es más o menos pesada que las demás en 3 pesadas de la balanza. ¿Puedes hacerlo en 3 pesadas si tienes 13 monedas? En general, dadas n monedas de las cuales una es falsa, ¿cuál es el número mínimo de pesadas para encontrar la falsa y determinar si es más o menos pesada que las otras?

Solución Egg Drop:

Si n es el número de plantas del edificio, presentamos una solución que hace como máximo $2\sqrt{n}-1$ lanzamientos (por tanto, para nuestro caso con $n = 100$, hará menos de 20 lanzamientos).

La forma en la que se puede pensar esta solución es la siguiente: si sólo tuviéramos un huevo, tendríamos que ir lanzándolo desde cada piso yendo de uno en uno, ya que si nos saltamos algún piso puede que se nos rompa el huevo y ya no podamos saber cuál era el valor de T.

Por tanto, una vez se rompa el primer huevo, tendremos que ir subiendo piso por piso por los valores posibles de T que no hemos descartado con los lanzamientos del primer huevo. Queremos que el número de pisos que tengamos que subir en ese caso no sea demasiado grande. Una forma de hacerlo es la siguiente: dividimos los pisos en intervalos y vamos lanzando el huevo desde la parte superior de cada intervalo, empezando por los intervalos de abajo y subiendo hacia arriba. Cada vez que lanzamos el huevo y no se rompe descartamos como posibles valores de T los del intervalo actual (los de intervalos anteriores ya se habían descartado). Cuando se rompe el primer huevo, solo tenemos que comprobar los pisos del intervalo actual (uno por uno). Queremos que los intervalos no sean demasiado largos, pero tampoco queremos que haya demasiados intervalos, por lo que $\sqrt{n}$ intervalos de longitud $\sqrt{n}$ (con los redondeamientos apropiados si n no fuera cuadrado perfecto) suponen un buen balance.

Veamos, para aclarar la solución, cómo se desarrollaría para el caso $n = 9$ ($n = 100$ es muy grande). Empezamos lanzando desde el piso 3: si se rompe el huevo, solo tenemos que lanzar el otro desde los pisos 1 y 2 para determinar el valor de T; si no se rompe, descartamos 1, 2 y 3 como posibles valores de T. A continuación lanzamos desde el piso 6 y si se rompe comprobamos los pisos 4 y 5; si no, lanzamos desde el piso 9 y si se rompe comprobamos el 7 y el 8. El peor caso es si $T = 8$ o $T = 9$, donde hacemos 5 lanzamientos para determinar el valor. En general, si $T = n-1$ o $T = n$, hacemos $\sqrt{n}$ lanzamientos en la fase de los “intervalos” y $\sqrt{n}-1$ lanzamientos dentro del “último intervalo”, por lo que en total el peor caso es $2\sqrt{n}-1$.

Esta solución no es la mejor posible, sin embargo. Consideremos el caso $n = 9$ otra vez, pero esta vez distribuyendo los intervalos tal que empezamos a lanzar desde el piso 4, después desde el 7, y después desde el 9. Se puede comprobar que aquí hacemos como mucho 4 lanzamientos. Igualmente, para $n = 100$ se puede hacer mejor que con 19 lanzamientos en el peor caso. Dejamos como ejercicio al lector adaptar esta solución para obtener un número máximo de lanzamientos más bajo.

Solución Moneda Falsa:

Una posible estrategia para 12 monedas en 3 pesadas es la siguiente: divides las 12 monedas en 3 grupos de 4 monedas y pones dos en la balanza. Hay dos casos según el resultado:

- Que un grupo pese más que otro: Entonces la moneda falsa está en uno de los dos grupos. Haces la siguiente pesada: en cada uno de los platos pones dos monedas del grupo “ligero” y una del “pesado”. Te quedarán dos monedas del grupo “pesado” sin pesar.
    - Si la balanza se desequilibra, entonces sabes que la moneda falsa es o bien una de las “posibles ligeras” del lado que menos pesa o bien la “posible pesada” del grupo que más pesa. Pesas las dos ligeras: Si una es más ligera que la otra, esa es la falsa y es más ligera. Si no, la falsa es la pesada.
    - Si la balanza está equilibrada, simplemente pesas las dos “posibles pesadas” que te quedan y la falsa es la más pesada.
- Que los dos grupos pesen igual: la moneda falsa está entre las 4 que no has pesado. Pones en la balanza dos de ellas en un plato y una de ellas y **una de las que has pesado** (que sabes seguro que no es falsa) en el otro.
    - Si se desequilibra, sabes que la moneda falsa es de las tres que no habías pesado. Pesas las que estaban en el mismo plato: sabes si, en el caso de alguna ser falsa, es más ligera o pesada según el resultado de la pesada anterior. Si pesan igual, la falsa es la tercera, que también sabes si es pesada o ligera.
    - Si está en equilibrio, la falsa es la moneda que te queda. Haces una pesada (contra una moneda que sabes que no es falsa) para determinar si es pesada o ligera.

Con 13 monedas no es posible: en general, para n pesadas como máximo puedes usar $\frac{3^n-3}{2}$ monedas (o, para m monedas, necesitas $\log_3(2m+3)$ pesadas). Una forma de demostrar esto, con las mismas ideas que la estrategia anterior, es primero demostrar las siguientes dos proposiciones (más sencillas, se demuestran con argumentos de inducción):

- Si las monedas que tienes son de dos colores distintos, tales que si la moneda falsa es de un color entonces es más pesada y si es del otro es más ligera, entonces con n pesadas como máximo puedes pesar 3^n monedas.
- Si las monedas son como en el problema original pero además tienes a tu disposición monedas extra que sabes que no son falsas, entonces con n pesadas como máximo puedes pesar $\frac{3^n-1}{2}$ monedas.

La motivación para pensar esto es la siguiente: cuando haces la primera pesada, o bien la balanza se desequilibra y tienes candidatos a monedas pesadas y monedas ligeras (como si las pintaras de un color) o bien está equilibrada y vuelves a hacer el problema con la pila de monedas que no has pesado, pero ahora puedes utilizar monedas que sabes que no son falsas. Utilizando los dos resultados anteriores y este razonamiento, obtenemos que si tienes n pesadas, en la primera pesada como máximo puedes poner 3^{n-1} monedas en la balanza (pero como tiene que ser un número par es $3^{n-1}-1$) y como máximo puedes dejar $\frac{3^{n-1}-1}{2}$ fuera, así que en total puedes tener como máximo $3^{n-1}-1 + \frac{3^{n-1}-1}{2} = \frac{3^n-3}{2}$ monedas.