Introducción

Antes de pensar en las estrategias que nos llevan a resolver un problema algorítmico, es bastante útil clasificarlos en distintos tipos de acuerdo a lo que se está pidiendo. En esta entrada platicaremos de tres tipos de problemas algorítmicos: los de optimización, los de decisión y los de enumeración. En los primeros queremos minimizar o maximizar alguna cantidad. En los segundos se nos pide responder una pregunta de sí o no. Y en los terceros se nos pide encontrar todos los objetos que satisfagan alguna propiedad.

Tipos de problemas algorítmicos

Con el fin de saber con qué estrategias resolver un problema algorítmico, resulta útil identificar qué es lo que está pidiendo. Hay varios tipos de problemas algorítmicos. En esta unidad nos enfocaremos en los siguientes tres.

- **Optimización.** Se tratan de encontrar un mínimo o un máximo (en general, un **valor óptimo**) para una cierta función que depende de un objeto.
- **Decisión.** Se tratan de responder preguntas de sí o no.
- **Enumeración.** Se tratan de encontrar o contar todos los objetos que cumplan cierta propiedad.

Para los tres tipos de problemas usualmente limitamos la búsqueda a un **espacio de estados**. Este es un conjunto que limita el universo del cuál pueden venir los objetos que se usan como argumento en las funciones a optimizar, los objetos de los que queremos responder si cumplen una propiedad, o bien los objetos que queremos encontrar que cumplan las propiedades. Discutiremos el concepto de espacio de estados más detalle en la siguiente entrada.

Hay algunos problemas algorítmicos que no necesariamente entran en algunos de estos rubros. Por ejemplo, en el problema de agregar un elemento a una `Lista` no estamos buscando un óptimo, determinando si algo pasa, o contando cosas. Hay algunos problemas algorítmicos que pueden estar en más de uno de estos rubros, por ejemplo, quizás queremos encontrar todos los objetos que optimicen una función. No pensaremos en esta clasificación como algo muy formal, sino más bien como algo que oriente nuestras estrategias de diseño de algoritmos.

Problemas de optimización

En un problema de optimización queremos minimizar o maximizar una función. Esto puede ser algo numérico, como maximizar la expresión $a+b$ con la condición de que a y b son números reales positivos de suma 100. También puede ser algo geométrico, como encontrar de entre todos los polígonos de 6 lados y área 1 aquel con menor perímetro. Sin embargo, estamos pensando en muchos más tipos de problemas, en donde los objetos que estudiamos y la función que depende de ellos pueden ser más complicados.

Ejemplo. Algunos problemas de optimización son los siguientes:

- **Brazo robótico que une circuitos.** Dados puntos el plano, queremos un camino hamiltoniano (es decir que pase una y sóla una vez por cada uno de ellos) de longitud mínima que los recorra.
- **Problema de la 2-SUMA máxima.** Tenemos una lista de números y queremos encontrar cuáles son los dos ellos cuya suma es mayor.
- **Problema del número cromático.** Dada una gráfica, encontrar su número cromático.
- **Problemas de optimización lineales.** Dada una función f que depende linealmente de variables $x_1,\ldots,x_n$, encontrar el máximo de la función si se satisface una o más restricciones lineales del estilo $g(x_1,\ldots,x_n)\leq M$.

En algunos problemas de optimización podemos utilizar técnicas provenientes de cálculo, como usar derivadas, convexidad, $\square$

En algunos problemas de optimización podemos utilizar técnicas provenientes de cálculo, como usar derivadas, convexidad $\square$ o desigualdades. Sin embargo, muchas de las funciones que queremos optimizar no son diferenciables, continuas o ni siquiera tienen como dominio a los números reales. Cuando estamos en esta situación, conviene ver si podemos restringir nuestra pregunta de optimización a una cantidad finita de valores. Si este es el caso, entonces el problema podría ser resuelto mediante una exploración computacional. De ahí surge la importancia de desarrollar técnicas algorítmicas que nos permitan resolver este tipo de problemas.

Problemas de decisión

En los problemas de decisión queremos responder preguntas cuya respuesta es sí o no. En algunos casos la pregunta es sencilla y simplemente se trata de determinar si un objeto tiene una propiedad o no. En otros casos los problemas de decisión son problemas de **existencia**: ¿habrá un objeto que cumpla ciertas propiedades dadas? Cuando el objeto sí existe, típicamente queremos exhibir un ejemplo o un **testigo** de que la respueta es sí. Cuando el objeto no existe, queremos que el algoritmo nos diga que no hay tal objeto y que, por supuesto, esto sea correcto.

Ejemplo. Algunos problemas de decisión son los siguientes:

- **Solución de Sudoku.** Dado un tablero de Sudoku en donde ya se han llenado algunas de las entradas, ¿será cierto que se puede completar para obtener una solución?
- **Problema de la 2-SUMA.** Tenemos una lista de números y un número dado M. Queremos ver si hay 2 números de la lista cuya suma sea M. Una variante tipo enumeración podría ser contar cuántas parejas cumplen que su suma es M.
- **Problema de gráfica bipartita.** Dada una gráfica, decir si es bipartita o no.
- **Problema del conjunto independiente grande.** Dada una gráfica y un entero k, decidir si una gráfica tiene un conjunto independiente de k vértices.

A veces un problema de optimización se puede convertir en varios problemas de decisión. Por ejemplo, para encontrar el $\square$ número cromático de una gráfica con n vértices (que es un problema de optimización) podemos hacer lo siguiente. Para cada k podemos hacernos la pregunta de si la gráfica tiene una k-coloración propia (que es un problema de decisión). Ya que tengamos todas aquellas k para las cuales la respuesta es «sí», el número cromático será simplemente la menor de ellas.

Problemas de enumeración

Finalmente, tenemos los problemas de enumeración. Se parecen un poco a los problemas de existencia en el sentido de que nos preguntamos por objetos que tienen cierta propiedad. Sin embargo, en este tipo de problemas lo que nos interesa no es encontrar únicamente un objeto que cumpla la propiedad, sino todos ellos. En unas ocasiones nos interesará únicamente **contar** cuántos de estos objetos existen, y de esta manera lo que nos interesa como respuesta es simplemente un número entero. En otras ocasiones nos interesará **enlistarlos**, ya sea mostrándolos en pantalla, o almacenándolos en memoria.

Ejemplo. Algunos problemas de enumeración son los siguientes:

- **Soluciones de Sudoku.** Dado un tablero de Sudoku en donde ya se han llenado algunas de las entradas, encontrar todas las posibles formas de llenarlo.
- **Enumeración de 2-SUMAS.** Tenemos una lista de números. Queremos ver cuántos posibles números se pueden obtener sumando dos elementos de la lista.
- **Enumeración de cliques.** Dada una gráfica, contar cuántos cliques con cierta cantidad de vértices tiene.
- **Ceros de un polinomio.** Dado un polinomio, contar cuántas veces su gráfica cruza el eje x.

Un problema de enumeración puede convertirse en muchos problemas de decisión. Por ejemplo, para contar todas las $\square$ coloraciones propias con tres colores de una gráfica (que problema de enumeración) podríamos hacer lo siguiente. Tomamos todas las posibles formas de colorear la gráfica con tres colores. Para cada una de ellas, nos preguntamos si es una coloración propia o no (que es un problema de decisión). Para responder el problema de enumeración, simplemente llevamos la cuenta de para cuántas coloraciones la respuesta fue «sí».

Tarea moral

Los siguientes problemas te ayudarán a practicar lo visto en esta entrada. Para resolverlos, necesitarás usar herramientas matemáticas, computacionales o ambas.

1. Determina si los siguientes problemas son de optimización, de decisión, de enumeración, de varios de estos tipos o de ninguno:
   - Una compañía de productos químicos quiere transportar k substancias distintas, pero hay algunas de ellas que no pueden transportarse en el mismo vehículo pues corren el riesgo de hacer una reacción peligrosa. Lo que le gustaría es usar pocos vehículos.
   - Manuel quiere hacer un algoritmo que le ayude a resolver el crucigrama del periódico.
   - Un gobierno local quiere ver si puede distribuir vacunas a los hospitales cercanos de manera que no se gasten más de dos millones de pesos en gastos de transporte y logística.
   - Una empresa quiere ofrecer una promoción a sus clientes. Para ello, quiere encontrar en su base de datos a todos los clientes que compren más de 5 veces al mes, por una cantidad total de al menos 10,000 pesos al mes, y que vivan en el norte de la ciudad.
   - Se quiere entrenar una red neuronal que distinga perros de gatos y cuyo error sea lo más pequeño posible.
   - El restaurante de pollos quiere encontrar todas las posibles formas de hacer paquetes de modo que no sean muy caros (para que los clientes los compren), pero que de cualquier forma les dejen ganancia. Quiere muchos tipos de paquetes para ofrecer variedad.
2. Investiga qué es la optimización entera, la optimización combinatoria y la optimización no lineal.
3. Describe un algoritmo que decida si entre los siguientes números hay tres de ellos cuyo producto sea 261374274
   $$[666,678,653,614,668,690,699,694,631,621,657,602,609,600,688,639.]$$
   ¿Cómo le harías para verificar todas las posibilidades? ¿Se te ocurre alguna forma de resolver el problema más rápido si tuvieras n números?
4. Describe un algoritmo que cuente de cuántas formas se pueden poner tres caballos de ajedrez en un tablero de 4×4 sin que estos se ataquen entre sí.
5. Describe un algoritmo que determine cuál es el subconjunto de los siguientes números cuya suma sea lo más chica posible:
   $$[45,311,53,-15,21,-42,-15,20,-21,-22.]$$
   Primero, describe un programa que pase por todos los subconjuntos para ir verificando las sumas. Luego, encuentra una forma de resolver el problema en tiempo $O(n)$ si tuvieras n núemeros.

Anterior
Heurísticas para la creación de algoritmos

Siguiente
Espacios de estados y heurísticas

Por Leonardo Ignacio Martínez Sandoval