Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2133 — Estructuras de Datos y Algoritmos 2'2022

Guía 3
Grafos, algoritmos codiciosos y programación dinámica

1. Grafos

1. (**I3-2022-1**) Las alternativas para desplazarse en la ciudad son múltiples: metro, buses, bicicletas, servicios de plataformas, caminando, etc. La elección de los medios de transporte más adecuados para ir de A a B depende del tiempo necesario para realizar el viaje en el medio elegido y el costo incurrido en el viaje. Un mismo viaje puede involucrar más de un medio de trasporte en diferentes tramos (A-metro-C-bus-D-caminar-B).

   El tiempo de desplazamiento y el costo involucrado para moverse dentro de la ciudad en cada alternativa de transporte está disponible en archivos como el siguiente para cada medio de transporte. Cada celda contiene el tiempo necesario y el costo para ir del origen al destino en el medio de trasporte correspondiente (las celdas vacías indican que no es posible ir en ese medio desde ese origen al destino).

   Metro

Origen/Destino	A	B	C	D	E
A		10|22			25|42
B	15|22		56|44	60|100	
C					70|150
D	34|55	60|100			
E		56|78			

   a) Proponga una estructura de datos para representar toda la información disponible para los múltiples medios de transporte y proponga una forma de implementar de esta estructura de datos en un lenguaje de programación.

   b) Escriba en pseudo código un algoritmo que permita determinar la mejor ruta para ir de un lugar a otro de la ciudad, registrando los múltiples medios de transporte utilizados, el tiempo y costo total del viaje. *Hint:* considere que el tiempo es oro.

2. (**I3-2022-1**) La universidad ha publicado un catálogo con todas sus carreras. La información de cada carrera incluye un párrafo que dice algo así como, "Si un estudiante de esta carrera cumple las siguientes condiciones ¡...aquí vienen las condiciones...¿, entonces puede traspasarse a alguna de estas otras carreras: ¡...aquí vienen los nombres de algunas otras carreras de la universidad¿".

   ¿Será posible que un estudiante pueda traspasarse desde su carrera a cualquiera otra carrera? Es decir, si desde la carrera A es posible traspasarse a las carreras B y C, y desde B es posible traspasarse a las carreras D y E, y desde C es posible traspasarse a la carrera F; entonces, desde A se podría llegar a D, pasando entremedio por B, o a F pasando entremedio por C. Y así sucesivamente a otras carreras, de modo que podría ocurrir que fuera posible traspasarse de A a cualquier otra carrera de la universidad (pasando entremedio por otras carreras). Entonces la pregunta es si la especificación del catálogo es tal

que finalmente permite el traspaso de cualquier carrera a cualquiera otra carrera. Plantea este problema como un problema en grafos:

a) ¿Qué representan los nodos y las aristas? ¿Son las aristas direccionales o no direccionales?

b) Dado tu grafo de (a), ¿qué significa que desde una carrera sea posible traspasarse a otra carrera, posiblemente pasando entre medio por otras carreras? ¿Cómo se puede saber si es así?

c) Dado tu grafo de (a), ¿qué significa que desde una carrera sea posible pasarse a cualquiera otra carrera? Y en realidad, ¿qué significa que desde cualquier carrera sea posible traspasarse a cualquiera otra? ¿Cómo se puede saber si es así?

3. (**I3-2018-1**) El algoritmo de Dijkstra puede resumirse así: Primero, colocamos el vértice fuente s en la solución — un árbol de rutas mínimas, T, originalmente vacío. Luego, construimos T de a una arista a la vez, agregando siempre a continuación la arista que da una ruta más corta desde s a un vértice que no está en T; es decir, agregamos vértices a T en el orden de sus distancias (a través de T) a s. Esta es la ver- sión abstracta del algoritmo.

   a) Una versión concreta particular del algoritmo es la que estudiamos en clase, y que toma tiempo $\mathcal{O}(E \log(V))$. Este tiempo es conveniente cuando el número de aristas de G es $\mathcal{O}(V)$, o en general, significativamente menor que V^2. Pero si G es muy denso, es decir, si el número de aristas es más bien $\mathcal{O}(V^2)$, entonces sería preferible una versión del algoritmo que tome tiempo $\mathcal{O}(V^2)$, y que por lo tanto es lineal en el número de aristas de G. Describe una versión del algoritmo con esta propiedad y justifica que es así.

   b) Muestra que en general el algoritmo de Dijkstra efectivamente no encuentra (todas) las rutas más cortas a partir de s si G tiene algunas aristas con costos o pesos negativos.

4. (**I3-2018-2**) La gran ventaja del algoritmo de Floyd-Warshall frente a Dijkstra es que permite tener las rutas precalculadas y almacenadas. El tiempo que toma en construir la matriz de costos mínimos entre todos los pares de nodos es $\mathcal{O}(|V|^3)$. Digamos que el algoritmo de Floyd-Warshall es muy complicado de entender por lo que queremos utilizar el algoritmo de Dijkstra para calcular esta matriz de costos mínimos.

   Considerando que el algoritmo de Dijkstra toma tiempo $\mathcal{O}(E \log(V))$ en su implementación con un heap como cola de prioridad

   a) Proponga el pseudocódigo del algoritmo para resolver el problema.

   b) Determine su complejidad.

2. Heaps

1. (**6.5-8**[CLRS09]) Entregue el pseudocódigo de la operación `HeapDelete`(A, i) que recibe un heap representado como arreglo A y una posición i del arreglo, y elimina el elemento $A[i]$ reestableciendo la propiedad de heap balanceado.

2. (**6.-2**[CLRS09]) Un heap d-ario es como un heap binario, pero donde los nodos que no son hojas pueden tener d hijos en lugar de 2.

   a) ¿Cómo representaría un heap d-ario en un arreglo?

   b) Entregue una implementación eficiente de `Extract` para un heap d-ario. Analice su complejidad en términos de d y n.

   c) Entregue una implementación eficiente de `Insert` para un heap d-ario. Analice su complejidad en términos de d y n.

3. (**I3-2022-1**) El viaje en el tiempo es posible gracias al Kettle Asimov, el cual se alimenta de cristales de tiempo. La fabricación de cristales de tiempo es un proceso secuencial, rápido y barato, pero de alta variabilidad en la calidad C del cristal, la cual se mide y registra en cada cristal durante su fabricación. La calidad del cristal es de gran importancia para el funcionamiento estable del Kettle Asimov, el cual debe ser alimentado de forma continua con los cristales de mejor calidad disponibles.

   En la estación Villarica existen n líneas de producción de cristales (L_i, $i = 1 \dots n$) y existe una línea de alimentación del Kettle con n posiciones (K_i, $i = 1 \dots n$) en que cada posición está asociada únicamente a la línea de producción respectiva ($K_i \leftrightarrow L_i$).

   En cada ciclo de operación, el Kettle toma el mejor cristal K_i desde la línea de alimentación y lo consume. La posición libre K_i es inmediatamente cubierta por un proceso P_i que toma el siguiente cristal producido por la línea L_i hasta el momento y lo ubica en la posición K_i de la línea de alimentación del Kettle, de modo que ésta esté completa para el siguiente ciclo de operación.

   a) Proponga una estructura de datos apropiada para modelar el proceso de alimentación del Kettle y escriba el pseudo código necesario para controlar eficientemente su operación.

   b) Diseñe una forma de mejorar la calidad global de los cristales que llegan a la línea de alimentación del Kettle, modificando lo menos posible su actual configuración; y explique por qué se obtiene una mejor calidad global de los cristales con su solución.

3. Estrategias algorítmicas

1. (**15.1-5**[CLRS09]) Los números de Fibonacci se definen según la recurrencia

$$f_i = \begin{cases} 0 & \text{si } i = 0 \\ 1 & \text{si } i = 1 \\ f_{i-1} + f_{i-2} & \text{si } i \geq 2 \end{cases}$$

   Proponga un algoritmo de programación dinámica en tiempo $\mathcal{O}(n)$ que calcule el n-ésimo número de Fibonacci. Dibuje el grafo de subproblemas. ¿Cuántos nodos y aristas tiene el grafo?

2. (**16.1-4**[CLRS09]) Suponga que tenemos un conjunto de actividades para programar en un gran número de salas, de manera que cualquier actividad puede tomar lugar en cualquier sala, pero solo una actividad puede ocurrir en una sala a la vez. Nos interesa maximizar el número de actividades programadas. Considere que cada actividad i tiene hora de inicio s_i y de término t_i. Entregue un algoritmo codicioso eficiente para determinar qué actividad usa qué sala.

3. (**16.2-5**[CLRS09]) Describa un algoritmo eficiente que, dado un conjunto $\{x_1, x_2, \dots, x_n\}$ de puntos en la recta real, determine el conjunto más pequeño de intervalos cerrados de largo 1 que contiene a todos los puntos dados. Argumente la correctitud de su algoritmo.

4. (**I3-2022-1**) Una empresa quiere realizar una fiesta para sus empleados. La empresa tiene una estructura jerárquica; es decir, la relación supervisor forma un árbol en cuya raíz está la presidenta de la empresa. La oficina de personal de la empresa ha asignado a cada empleado un índice de simpatía, el cual es un número real. Finalmente, con el propósito de que la fiesta sea divertida para todos los asistentes, se he decidido que si se invita a una persona, entonces no se invita al supervisor inmediato de esa persona.

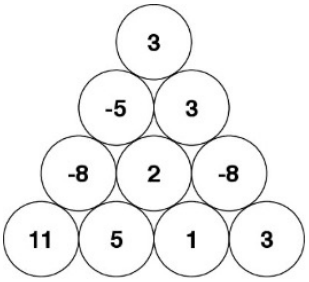
   Tienes ante ti el árbol que describe la estructura jerárquica de la empresa: cada nodo tiene el nom-bre de un empleado y su índice de simpatía (además de los punteros a sus supervisados y a su supervi-sor). ¿Cuál debería ser la lista de invitados de manera de maximizar la suma de los índices de simpatía de los invitados? Plantea una solución de programación dinámica para este problema; en particular:

a) Caracteriza la estructura de la solución óptima ("La presidenta de la empresa puede estar en la lista de invitados o no. Si está, entonces, ...; si no está, entonces ..."), de manera que quede claro que encontrar la solución óptima involucra encontrar soluciones óptimas a problemas del mismo tipo pero más pequeños.

b) Generalizando a partir de (a), define recursivamente el valor de una solución óptima.

c) Muestra con un ejemplo que no es buena idea calcular el valor de la solución óptima simple-mente aplicando (b), ya que esto significa resolver muchas veces un mismo subproblema.

5. (**I3-2018-2**) El canal DCC TV tiene un nuevo programa llamado EDD donde los participantes tienen la chance de concursar y ganar dinero en premios. El juego consiste en un triángulo de pelotas, donde cada pelota tiene impreso un número entero, tal como se muestra en la figura.



El jugador puede elegir una pelota de la pirámide y sacarla o no sacar ninguna. En caso de que saque una pelota, se quedará con los puntos de esa pelota y todas las que estén arriba de ella. En caso de que saque ninguna, se quedará con 0 puntos. En la figura 1, para sacar la pelota 1, se necesitan sacar también las pelotas 2, -8, -5, 3 y 3 por lo que el valor de sacar la pelota 1 sería -4.

Esta pirámide se representa como arreglo de arreglos M donde $M[0]$ es el nivel superior y $M[n]$ es el nivel de más abajo. Cada nivel $M[i]$ tiene $i+1$ elementos. La celda $M[i][j]$ contiene el valor de la pelota j-ésima de la fila i-ésima.

a) Se define $P(i, j, M)$ como el premio obtenido al sacar la pelota $M[i][j]$. Escriba una ecuación de recurrencia para calcular $P(i, j, M)$. *Hint:* note que hay pelotas que se repiten cuando se ve el premio de aquellas que están sobre $M[i][j]$.

b) Proponga un algoritmo iterativo (no recursivo) para resolver este problema y determina el premio al sacar cualquier pelota de la pirámide.

4. Referencias

Referencias

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.