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

IIC2133 — Estructuras de Datos y Algoritmos 2'2024

Interrogación 2

18 de noviembre de 2024

Condiciones de entrega: Debe entregar solo 3 de las siguientes 4 preguntas.

Tiempo: 2 horas

Entrega: Al final de la prueba tienen 10 minutos para subir la imagen de la prueba a Canvas,**cada pregunta y el torpedo** por separado en vertical

Evaluación: Cada pregunta tiene 6 puntos (+1 punto base). La nota es el promedio de las 3 preguntas entregadas.

1. Hash

Ud. es el dueño de una recién inaugurada veterinaria con bajo presupuesto y recibe mascotas, todas debidamente chipeadas; el chip es un dispositivo electrónico legible por NFC (como las tarjetas de crédito) con un número serial solamente, Existe un Registro Nacional de Mascotas que mantiene el par (número de chip, RUN del responsable)
Como medida de precaución, antes de cualquier atención, Ud. verifica si el chip instalado en la mascota corresponde efectivamente al Responsable. Suponga que, del universo de 100.000 mascotas chipeadas Ud. recibe un conjunto acotado de ellas y que los chips son solo números únicos desde 0 hasta 100.000.
Como su primo estudia computación, Ud. le encargó un sistema de registro de mascotas quien, muy astuto, solucionó el problema usando una **tabla de hash de 11 buckets** donde cada uno solo guarda un par (llave, RUN del Responsable) donde la llave es el número del chip.
Dibuje la tabla de Hash y especifique las funciones para los siguientes esquemas de resolución de colisiones y valores de los chips. 0,22,31,4,15,28,17,88,59. (no es relevante mostrar los valores del RUN)

1. (1 pt.) Especifique las funciones de hash1 y hash2 rellenando la tabla de hash.

   Para las siguientes preguntas, haga uso de su función de hash1

2. (0,5 pts.) Dibuje la tabla usando colisión por encadenamiento.
3. (0,5 pts.) Dibuje la tabla usando colisión por sondeo lineal.
4. (0,5 pts.) Dibuje la tabla usando colisión por sondeo cuadrático.

   Para las siguientes preguntas, haga uso de su función de hash1 y hash2

5. (0,5 pts.) Dibuje la tabla usando colisión por rehashing.
6. (2 pts.) Compare las diferentes técnicas de resolución de colisiones en términos del número de cálculos (no complejidad algorítmica) y la concentración de las llaves para las técnicas elegidas.
7. (1 pt.) Cómo cambia su respuesta anterior si en el bucket caben 2 pares (clave, RUN).

2. Backtracking

Considere el siguiente acertijo lógico: En cinco casas alineadas de izquierda a derecha, cada una de un color diferente, viven 5 personas de diferentes nacionalidades, cada una de las cuales prefiere un deporte diferente, una bebida diferente y una mascota diferente. Además, están dados los siguientes hechos:

- La persona inglesa vive en la casa roja.
- La persona española tiene el perro.
- La persona argentina vive en la primera casa de la izquierda.
- Se prefiere el fútbol en la casa amarilla.
- La persona que prefiere el basketball vive en la casa junto a la persona que tiene el gato.
- La persona argentina vive junto a la casa azul.
- La persona que prefiere el voleyball tiene el conejo.
- La persona que prefiere el tenis bebe jugo de naranja.
- La persona chilena bebe té.
- La persona japonesa prefiere el judo.
- Se prefiere el fútbol en la casa junto a la casa donde se tiene el caballo.
- Se bebe café en la casa verde.
- La casa verde está inmediatamente a la derecha (a su derecha) de la casa blanca.
- Se bebe leche en la casa del medio.

La pregunta del acertijo es: **¿Dónde vive la cebra y en qué casa se bebe agua?**. En este contexto se le pide:

a) (1 pt.) Modele este problema como un CSP (Problema de Satisfacción de Restricciones), para esto defina cuáles son las variables, dominios y restricciones del problema.

b) (2 pts.) Proponga el pseudo código de la función `solveCSP()` para resolver el acertijo, indicando los parámetros adecuados para realizar la llamada inicial y obtener la solución al acertijo. Asuma que dispone de una función `verificaCSP()` adecuada para verificar las restricciones del problema.

c) (2 pts.) Explique cómo aplicaría en su solución podas y propagación, de modo de beneficiar el desempeño de su solución al utilizar los hechos conocidos del acertijo.

d) (1 pt.) Adicionalmente a los hechos indicados en el acertijo, se sabe que habitualmente (la mayor parte de las veces): Los argentinos prefieren el fútbol y el café, los japoneses el judo y los gatos, los chilenos el fútbol y el té, los españoles los perros y el café, los ingleses el té y el tenis. ¿De qué forma utilizaría esta información para mejorar el desempeño de su solución?

3. Programación dinámica

Considera la ruta Santiago – Puerto Montt. A lo largo de esta ruta, la autoridad vial ha dispuesto n lugares en los cuales está permitido poner letreros con propaganda. Estos lugares —$x_1, x_2, \ldots, x_n$— están especificados por sus distancias desde Santiago en kilómetros, siendo x_1 el más cercano. Por otra parte, tu negocio contrató a una empresa de marketing que calculó que si pones un letrero en el lugar x_i, entonces vas a recibir una ganancia de r_i (que representa las ventas que va a hacer tu negocio gracias a esa propaganda). Hay, sin embargo, una única restricción legal que especifica que un mismo negocio no puede poner letreros a menos de 5 kms de separación entre ellos. Por lo tanto, la pregunta es, ¿En cuáles lugares — un subconjunto de $x_1, x_2, \ldots, x_n$— te conviene poner los letreros con propaganda de tu negocio, cumpliendo con la restricción anterior, de manera de maximizar el total de las ganancias que recibirías? Por ejemplo, si $n = 4$, $(x_1, x_2, x_3, x_4) = (6, 7, 12, 14)$ y $(r_1, r_2, r_3, r_4) = (5, 6, 5, 1)$, entonces la solución óptima sería poner los letreros en x_1 y x_3 para una ganancia total de 10. Plantea un algoritmo de programación dinámica para resolver este problema.

1. (1 pt.)Considera las dos alternativas de que el lugar x_n esté o no en la solución óptima; ¿Cuál es el problema que queda por resolver en cada caso?

2. (2 pts.)¿Cómo se generaliza este razonamiento si consideramos el problema definido sólo por los primeros k lugares: $x_1, x_2, \ldots, x_k$?

3. (2 pts.) Si **opt(k)** representa la ganancia de un subconjunto óptimo de lugares entre $x_1, x_2, \ldots, x_k$, ¿cuál es la recurrencia correspondiente? Es decir, si **opt(k)** = max ... , ... , ¿qué va en los ...?

4. (1 pt.) Así, finalmente, a partir de su respuesta anterior, plantea el algoritmo pedido.

4. Grafos

1. Una forma de recorrer (y descubrir) un grafo no direccional $G = (V, E)$ es la siguiente: Primero, elegimos un vértice cualquiera de V y lo ponemos en una cola Q inicialmente vacía; Q es una cola común, del tipo "el primero que entra es el primero que sale". Luego, hacemos lo siguiente: Repetidamente, sacamos el primer vértice de Q, llamémoslo u; recorremos la lista de adyacencias de u y cada vértice que encontramos en la lista lo ponemos en Q.

   a) (0.5 pts.) Explica qué se puede hacer para que este algoritmo no "repita" vértices, es decir, para que los vértices que ya han sido descubiertos no vuelvan a ser descubiertos.

   b) (0.5 pts.) ¿Cuál es la condición de término del algoritmo?

   c) (1 pt.) ¿Qué información adicional puede extraerse de este algoritmo en términos de la distancia, medida en número de aristas, que hay entre el primer vértice que pusimos en Q y cada uno de los otros vértices que van siendo descubiertos?

   d) (1 pt.) Describe una versión completa del algoritmo —es decir, incluyendo los tres puntos anteriores— en pseudocódigo similar al usado en clases para describir el algoritmo DFS.

2. Queremos saber si estando en una estación del metro, puedo ir a cualquier otra estación usando únicamente el metro. Para esto, representamos la red del metro como un grafo direccional, de la siguiente manera: Cada estación es un vértice, y, si en una misma línea del metro la estación v es la próxima estación a la estación u, entonces (y sólo entonces) incluimos una arista direccional (u, v).

   a) (2 pts.) Describe un algoritmo eficiente en pseudocódigo que permita responder la pregunta original; tu algoritmo puede hacer uso sólo del algoritmo DFS estudiado en clases.

   b) (1 pt.) Justifica que tu algoritmo es eficiente.

Respuesta Hash

Pregunta_____ No de alumno: _____
Nombre: _____

1. (1pt.) Especifique las funciones de hash1 y hash2 rellenando la tabla de hash.

   Para las siguientes preguntas, haga uso de su función de hash1

2. (0,5pt.) Dibuje la tabla usando colisión por encadenamiento.

3. (0,5pt.) Dibuje la tabla usando colisión por sondeo lineal.

4. (0,5pt.) Dibuje la tabla usando colisión por sondeo cuadrático.

   Para la siguiente pregunta, haga uso de su función de hash1 y hash2

5. (0,5pt.) Dibuje la tabla usando colisión por rehashing.

6. (2pt.) Compare las diferentes técnicas de resolución de colisiones anteriores en términos del **número de cálculos** (operaciones, no complejidad algorítmica) y la **concentración de las llaves** para las técnicas elegidas.

técnica	cálculos	concentración
encadenamiento		
sondeo lineal		
sondeo cuadrático		
rehashing		

7. (1pt.) Cómo cambia su respuesta anterior si en el bucket caben 2 pares (clave, RUN)

técnica	cálculos	concentración
encadenamiento		
sondeo lineal		
sondeo cuadrático		
rehashing		

Respuesta Backtracking

Pregunta_____ No de alumno: _____
Nombre: _____

Pregunta_____ No de alumno: _____
Nombre: _____ _____

Respuesta Programación dinámica

Pregunta_____ No de alumno: _____
Nombre: _____

Pregunta_____ No de alumno: _____
Nombre: _____ _____

Respuesta Grafos

Pregunta_____ No de alumno: _____
Nombre: _____

Pregunta_____ No de alumno: _____
Nombre: _____ _____