Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC2133 - Estructura de datos y algoritmos

Pauta Tarea 2

6 de diciembre de 2021

2º semestre 2021 - Profesores Yadran Eterovic, Martín Muñoz

Parte 1: Hash (max. 3 pts.)

¿Por qué utilizar *Tabla de Hash* es buena opción? (max. 0,5 pts.)

Problema con gran cantidad de input que hace poco factible resolverlo en el tiempo requerido si no se utiliza tablas de hash. Las tablas de hash ayudan a que el tiempo de búsqueda de las consultas sea mucho menor a que si se resuelve el problema de forma iterativa.

- Si menciona que el tiempo de búsqueda es menor a otra forma de resolución. (0,5 pts.)
- Si menciona algún beneficio de usar tablas de hash pero no menciona que el tiempo de búsqueda es menor a otra forma de resolución. (0,3 pts.)
- Si no menciona beneficios de usar tablas de hash. (0 pts.)

Presentar una *función de hash* que solucione el problema (max. 0,75 pts.)

Estudiante debe presentar una función de hash que sirva para resolver el problema, puede no demostrarlo directamente pero fundamentar bien.

- Si la función de hash resuelve el problema. (0,75 pts.)
- Si la función resuelve el problema, pero no está bien argumentado. (0,4 pts.)
- Si no se presenta función de hash. (0 pts.)

Demostrar que la solución es incremental (max. 0,75 pts.)

Para incrementalidad existen diferentes modelaciones, aunque basta con que el calcular el hash de un string `ABCD` sea marginalmente mas caro que calcular `ABC`. Esto es Hash(ABC) + Hash(D) u otras soluciones similares de tal manera que no vuelvan a calcular el hash *completo*, sino que utilicen el hash calculado anterior.

- Por comentar y justificar correctamente que la función es incremental (0,75 pts.)
- Por comentar y justificar parcialmente que la función es incremental (0,4 pts.)
- Cualquier otro caso. (0 pts.)

Explicar qué beneficios traería que la *función de hash* fuera uniforme (max. 0,5 pts.)

Mirar en la tabla en $\mathcal{O}(h)$ promedio con h el factor de carga. Si la función de hash fuera uniforme la cantidad de colisiones son menores a que si no lo fuera, esto afecta el tiempo de búsqueda de las consultas.

- Si se habla correctamente sobre la cantidad de colisiones y el tiempo de búsqueda. (0,5 pts.)
- Si no comenta las colisiones y tiempo de búsqueda. (0 pts.)

Comentar condiciones de y/o elementos de *resizing, probing*, u otros según corresponda (max. 0,5 pts.)

- Resizing: En caso de usar resizing, mencionar cuál es el factor a realizar el resize y por qué es una buena solución
- Factor de carga: justificar el factor de carga elegido y por qué se utilizó
- Mencionar y justificar si se usó direccionamiento abierto o cerrado (encadenamiento)

Puntajes.

- Justifica correctamente direccionamiento abierto o cerrado junto con parámetros de factor de carga y resizing (En caso de utilizarlo este último, si no solo respecto a factor de carga). (0,5 pts.)
- Justifica correctamente direccionamiento abierto o cerrado. (0,3 pts.)
- Por sólo justificar correctamente parámetros de factor de carga y resizing (En caso de utilizarlo este último, si no solo respecto a factor de carga). (0,2 pts.)

Puntaje máximo Parte 1 Hash: 3 pts.

Parte 2: Backtracking (max. 3 pts.)

¿Por qué backtracking es una buena opción para resolver el problema (max. 1 pts.)

Backtracking es una buena modelación al problema al ser un problema de clase CSP (Problema de satisfacción de restricciones). Esto indica que el numero de soluciones posibles es gigante

(problemas NP). Por lo que es conveniente implementar métodos de backtracking que nos permita aplicar estrategias que permitan reducir el árbol de ejecuciones.

- Por justificar la utilización de backtracking en este tipo de problemas. (1 pts.)
- Puntaje parcial asignado a criterio del ayudante.

Comparación complejidades con y sin backtracking (max. 1 pts.)

La complejidad como tal requiere herramientas que van mas allá del alcance del curso, por lo que basta con comentar a grandes rasgos como backtracking *reduce* la complejidad práctica del problema (Siendo así la teórica en notación $\mathcal{O}$ igual).

- Por comparar las complejidades (puede ser de forma abstracta) y concluir que backtracking disminuye el runtime pero la complejidad teórica se mantiene (en el peor caso). (1 pts.)
- Por sólo comparar las complejidades o sólo concluir que backtracking disminuye el runtime pero la complejidad teórica se mantiene. (0,5 pts.)

Presentar y justificar las podas utilizadas (max. 1 pts.)

Hay 3 podas claras y principales.

- Todas las filas y columnas tienen que sumar 260.
- El siguiente movimiento tiene que estar válidamente a un movimiento de caballo.
- En caso de estar asignado, el movimiento siguiente tiene que estar a un movimiento de caballo del anterior.

En lo particular la forma óptima de implementar estas podas era en el inicio de cada asignación, donde se revisa inmediatamente si sumar 260 es imposible en alguna fila o columna, si el movimiento es caballo válido, etc.

- Por mencionar las podas correctamente y justificar. (1 pts.)
- Por mencionar alguna poda y justificarla correctamente a criterio del ayudante. (0,3 pts. por poda)

En caso de mencionar alguna otra poda asignar puntaje correspondiente si se justifica correctamente.

Puntaje máximo Parte 2 Backtracking: 3 pts.