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

IIC2133 — Estructuras de Datos y Algoritmos 2'2024

Examen

11 de diciembre de 2024

Condiciones de entrega: Para los que rindieron la I1 y la I2, deben entregar solo 3 de las siguientes 4 preguntas. Para los que tienen Interrogación recuperativa, deben entregar entre las 3 preguntas la correspondiente a la materia de la Interrogación faltante.

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. La nota de la I recuperativa es el promedio de las preguntas recuperativas y la pregunta correspondiente del examen.

1. Orden

Dada la siguiente versión del algoritmo **Selection Sort**:

```
input: Secuencia de datos en Arreglo A[]
output: Nueva secuencia B[] ordenada
selectionSort(A[]):
    B ← vacia
    for i = 0 to n - 2:
        min ← 0
        for j = 1 to n - 1:
            if A[j] < A[min]:
                min ← j
        B[i] ← A[min]
        A[min] ← +infinito
    return B
```

a) (2 pts) Escriba el pseudocódigo para **Tournament Sort**, el cual es una mejora de **Selection Sort** al utilizar una cola priorizada para seleccionar el siguiente elemento en orden. Puede asumir que dispone de las funciones de una cola priorizada vistas en clases.

b) (2 pts) Determine la complejidad de tiempo de **Tournament Sort**: ¿hay un mejor o peor caso? ¿Cuál es su complejidad de memoria?

c) (1 pt) Argumente (no demuestre formalmente) que **Tournament Sort** es correcto.

d) (1 pt) ¿Es **Tournament Sort** estable? Argumente su respuesta.

2. B+ Hash

En el contexto de Bases de Datos existe el concepto de índice que es una estructura de datos B+ o Tabla de Hash donde se guarda la llave del registro. Si se tiene un indice A del tipo B+ y uno B en tabla de hash sobre la llave responda:

a) (3 pts) Escriba los algoritmos usando la mejor estructura para hacerlo.

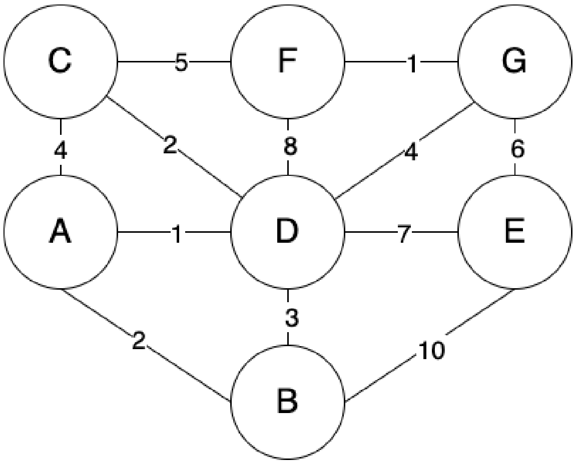
- Rango(i, f), con i,f llaves devolviendo un arreglo (o lista) con todos los registros tal que se encuentra en el rango $[i, f]$
- Record(i) que busca y entrega el registro con el valor de llave(i)

b) (3 pts) Escriba los algoritmos

- Insert(k, r), k llave r registro
- Delete(k), k llave que borra el registro de llave k.

3. MST Kruskal y Prim

Considera el siguiente grafo no direccional con costos.



Vértice	**T**	**Distancia**	**Padre**
A	-	∞	–
B	-	∞	–
C	-	∞	–
D	-	∞	–
E	-	∞	–
F	-	∞	–
G	-	∞	–

a) (2 pts) Ejecuta paso a paso el algoritmo de **Prim** para determinar un árbol de cobertura de costo mínimo (**MST**), tomando A como vértice de partida. En cada paso muestra la versión actualizada de la tabla a la derecha, en que $\boldsymbol{T}$ indica si el vértice ya está en la solución.

b) (2 pts) Demuestra con un contraejemplo que el árbol producido por **Prim** no necesariamente tiene las rutas más cortas desde el nodo inicial al resto de los nodos.

c) (2 pts) Una alternativa para determinar un **MST** es el algoritmo de **Kruskal**. Si todos los costos de las aristas son números enteros en el rango 1 a $|V|$, ¿qué tan rápido, en notación $O(\cdot)$, se puede hacer que ejecute el algoritmo de Kruskal? **Indicación:** Considera que el algoritmo incluye una inicialización, una ordenación, y finalmente la ejecución del algoritmo propiamente tal.

4. Bellman Ford

En los grafos direccionales con ciclos es imposible ordenar los vértices, tal que el vértice u esté antes que el vértice v indiscutiblemente. Por lo tanto, el algoritmo de Bellman-Ford para determinar rutas más cortas desde un vértice de partida s, debe actualizar (update) cada arista del grafo varias veces. En los grafos acíclicos, en cambio, sí es posible ordenar los vértices topológicamente, como vimos en clase.

a) Siguiendo este orden, podemos asegurar que una vez que la arista (u, v) ha sido actualizada, (updated) la distancia u.dist no cambia más. Demuestra esta afirmación.

b) Aprovechando a), escribe un algoritmo de tiempo O(V+E) para encontrar las rutas más cortas en un grafo direccional acíclico desde un vértice de partida s.

c) Justifica la corrección y tiempo de ejecución de tu algoritmo.