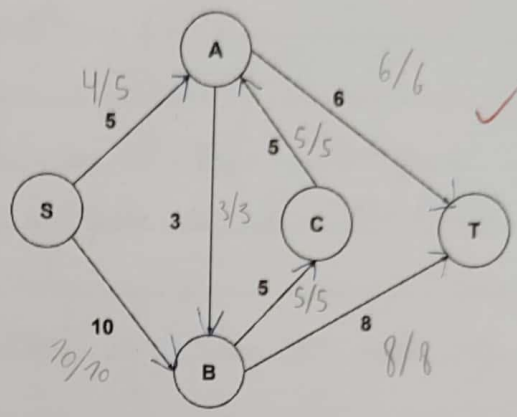
AyC 2023 - Actividad 12- Jueves 15/6/2023 - (Indicar Apellidos/Nombres): [redacted]

(1,00) 1) (2 ptos Apl/uso) Flujo máximo: Considere la siguiente red de flujo modelada a partir del siguiente grafo compuesto por un nodo inicial S, un nodo final T, otros tres nodos A, B y C; y los arcos que se muestran en la figura. Los valores indicados en los arcos corresponden a la capacidad de cada arco.



Se pide:

a. Indicar cuál es el flujo máximo de su red.
Indicando claramente el flujo sobre cada arco que le permite obtener el flujo máximo indicado.
b. Mostrar la red residual final
c. Mostrar claramente todos los posibles cortes con el valor de capacidad de cada uno de ellos, indicando cuál coincide con el flujo máximo encontrado.

(1,00) 2) (1 pto diseño) Considerando un recorrido BFS genérico, indique cuáles de las siguientes afirmaciones son correctas y cuáles no.

a. si todos los arcos pesan 1, se pueden hallar distancias mínimas desde un origen con un recorrido BFS V
b. el recorrido BFS es determinista F
c. el recorrido BFS almacena los nodos grises en una estructura de datos pila F
d. un recorrido BFS de un grafo dirigido puede generar arcos hacia atrás F

(1,00) 3) (1 pto diseño) Explique cómo utilizaría recorridos DFS para diseñar un algoritmo que obtenga el orden topológico de un grafo dirigido acíclico

---

4) El Algoritmo de Kruskal es un algoritmo greedy que resuelve el problema de encontrar un árbol de cubrimiento mínimo. Se caracteriza por seleccionar en cada iteración el menor de los arcos todavía no considerados. Si el arco seleccionado junto con la solución parcial es viable, entonces se incluye en la solución parcial. En caso contrario, es descartado.

(2,00) a. (4 ptos ED) Considere dos opciones distintas de ED's para almacenar los arcos, explique cómo las usaría para implementar Kruskal, y en cada caso muestre cómo quedaría el algoritmo.
(0,75) b. (3ptos A/C) Analice y discuta sobre el tiempo de ejecución en cada caso.

3) Se podría realizar lo siguiente:

- Realizamos un recorrido DFS en el grafo, empezando en cualquier nodo no visitado y aplicando DFS a partir de él. Mantenemos un conjunto de nodos visitados para evitar ciclos.
- Durante el recorrido DFS, después de visitar un nodo y antes de retroceder, agregamos ese nodo a una pila.
- Una vez que se haya completado el recorrido DFS en todos los nodos alcanzables, el orden topológico se puede obtener vaciando la pila en el orden inverso.

1) a) El máximo flujo es 14. ✓

✗ b) La capacidad residual del arco $S \to A$ es 1.
En el resto de los arcos es 0

nodos = V arcos = E

4) a) **Opción 1:** utilizar conjuntos disjuntos y ordenar los arcos al principio → lista para los arcos

1. Ordenar los arcos en orden creciente de peso $O(E \log E)$
2. Inicializar un conjunto disjunto para cada nodo del grafo, donde cada nodo es un conjunto separado
3. Iterar sobre los arcos ordenados, y para cada arco (u, v) con peso w: $O(E)$
   a) Verificar si los nodos u y v están en conjuntos disjuntos diferentes. Si es así, significa que añadir este arco no crearía un ciclo en el árbol parcial $O(a(V))$ → Función inversa de Ackermann
   b) Si los nodos u y v están en conjuntos disjuntos diferentes, unir los conjuntos disjuntos correspondientes a u y v.
   c) Añadir el arco (u, v) a la solución parcial del árbol de cubrimiento mínimo.
4. La solución parcial obtenida después de iterar sobre todos los arcos será el árbol de cubrimiento mínimo. → no necesariamente todos

Opción 2: Utilizar un Heap invertido → para los arcos

1. Inicializar un Heap invertido vacío para almacenar arcos
2. Para cada arco del grafo, insertarlo en el Heap invertido con su peso como prioridad. $O(E \log E)$
3. Inicializar un conjunto disjunto para cada nodo del grafo, donde cada nodo es un conjunto separado
4. Mientras el Heap invertido no esté vacío: $O(E)$ → mientras no se haya formado un árbol de cubrimiento
   a) Extraer el arco de menor peso del Heap invertido
   b) Verificar si los nodos del arco están en conjuntos disjuntos diferentes $O(a(V))$
   c) Si los nodos están en conjuntos disjuntos diferentes, unir los conjuntos disjuntos correspondientes y añadir el arco a la solución parcial del árbol de cubrimiento
5. La solución parcial obtenida luego de extraer todos los arcos del Heap invertido será el árbol de cubrimiento mínimo → no necesariamente todos

b) La Opción 1 tiene un tiempo de ejecución de $O(E \log E + E\, a(V))$ ¿Por qué?
La opción 2 tiene el mismo tiempo de ejecución. Sin embargo, debido al uso del Heap invertido, las constantes ocultas en el término $O(E \log E)$ pueden ser menores que las de la opción 1. ¿por qué? ¿a qué corresponde?

NOTA