UNIVERSIDAD TORCUATO DI TELLA

Tecnología Digital V: Diseño de algoritmos
Licenciatura en Tecnología Digital
Primer Semestre, 2024

Modelo del primer parcial

1. (30 puntos) Dado un multiconjunto (es decir, un conjunto con repeticiones) de números enteros $C = \{a_1, a_2, \ldots, a_n\}$ y un número k, el SUBSET-SUM PROBLEM consiste en determinar si existe un subconjunto $S \subseteq C$ tal que la suma de los elementos en S sea k. Por ejemplo, si tenemos como entrada el conjunto $C = \{2, 2, 5, 10\}$ y $k = 9$, entonces la respuesta es el multiconjunto $S = \{2, 2, 5\}$, cuya suma es 9. En cambio, para $k = 11$ no hay solución al problema.

   Consideremos el siguiente algoritmo de programación dinámica, que contiene dos bugs.

```
bool subsetsum_pd(int* C, int n, int k)
{
    // Retorna una matriz de bool de nxk
    // Los valores por default es False
    m** = crear_matriz<bool>(n+1,k+1);

    for (int i = 0; i <= n; i++)
        m[i][0] = False;
                  True!

    for (int l = 0; l <= k; l++)
        m[0][l] = False;

    for (int i = 1; i <= n; i++) {
        for (int l = 1; l <= k; l++) {
            if (l < C[i])
                m[i][l] = m[i-1][l]    # caso donde no lo incluyo
            else
                m[i][l] = m[i-1][l]  ||  m[i-1][l - C[i]];
        }
    }

    return m[n][k];
}
```

$C = \{2, 2, 3\}$ $k = 4$

$n = 3$ $3 \times 5 = 15$

n \ k	0	1	2	3	4
0	T	F	F	F	F
1	T				
2	T				
3	T				

   a) Dar dos casos de test con que consideren $|C| \geq 2$ para evidenciar cada uno de los bugs.[1]
   b) Proponer una corrección para este algoritmo, y argumentar por qué con esta corrección los casos de test del punto anterior dejan de fallar.

2. (30 puntos) Un grafo es un árbol si no tiene ciclos y es conexo.

   a) Existe algún árbol T con 4 vértices, tres de grado 1 y uno de grado 2?
   b) Demostrar que si se agrega una arista nueva a un árbol se crea un ciclo.

3. El algoritmo de Dijkstra resuelve el problema de camino mínimo en grafos dirigidos.

   a) Dado un digrafo $D = (N, A)$ con una función de distancia $w : A \to \mathbb{R}_{\geq 0}$, y sea $s \in N$. Proponer un algoritmo que retorne un subonconjunto $S \subseteq N$ con todos los nodos tal que el costo de llegar desde s es lo sumo C. Justificar por qué el algoritmo es correcto.

[1] No forma parte del modelo de parcial: se puede dar un caso de test donde la función actual retorne el valor correcto?

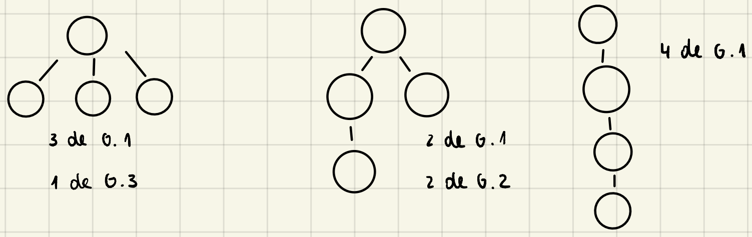
2.

2. (30 puntos) Un grafo es un árbol si no tiene ciclos y es conexo.

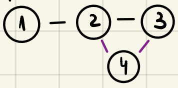
a) Existe algún árbol T con 4 vértices, tres de grado 1 y uno de grado 2?

b) Demostrar que si se agrega una arista nueva a un árbol se crea un ciclo.

a) 4 vértices, 3 grado 1, uno de grado 2.

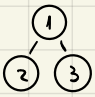


Parecería que no. Se puede empezar armando un árbol de 3 vert. de grado 1. Hasta ahí todo OK. Se ve algo así. 1 — 2 — 3. Ahora sucede lo siguiente. Fuerzo a que haya un 4to vértice que sea de grado 2. A ese 4TO lo tengo que conectar sí o sí con 2 vert. Se ve algo así

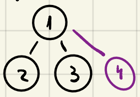


(uno de 3 posibilidades, es lo mismo). No cumple.

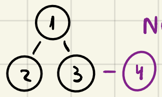
y si fuerzo que haya uno de grado 2?



1 tiene grado 2 y 2 y 3 grado 1. Por ahora todo OK. Agrego el 4to (grado 1 si o si).



NO cumple



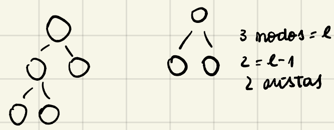
NO cumple

Lema handshacking

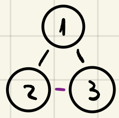
$\Sigma\, d(v) = 2 \times$ cant. nodos

$3 + 2 = 2 \times 4$

$5 \neq 8$ NO VALE!

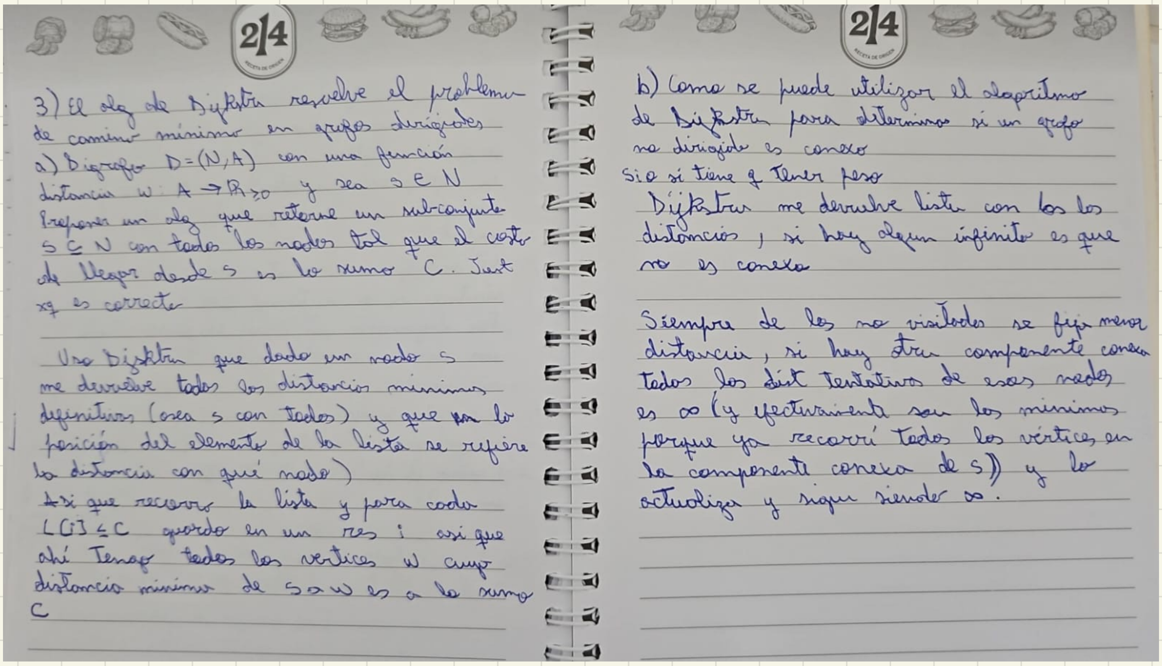


b)



ÁRBOL!!

Sin importar la cantidad de ramas, si yo tengo l cant. de nodos donde 1 es padre, l-1 son hijos y l-1 aristas, si agrego una arista (osea # aristas = l) yo AGREGO un camino que no existía y por lo tanto, como en los árboles yo tenía (x def) un camino UNICO para cada par de nodos, ahora en todo el árbol tengo 2 caminos para llegar a esos nodos que conecté, así se forma un ciclo.



3) El alg de Dijkstra resuelve el problema de camino mínimo en grafos dirigidos
a) Digrafo $D=(N,A)$ con una función distancia $w: A \to \mathbb{R}_{\geq 0}$ y sea $s \in N$
Proponer un alg que retorne un sub-conjunto $S \subseteq N$ con todos los nodos tal que el costo de llegar desde s es lo sumo C. Just xq es correcto

Uso Dijkstra que dado un nodo s me devuelve todas las distancias mínimas definitivas (osea s con todos) y que por la posición del elemento de la lista se refiere la distancia con qué nodo)
Así que recorro la lista y para cada $L[i] \leq C$ guardo en un res i así que ahí Tengo todos los vertices w cuya distancia mínima de s a w es a lo sumo C

b) Como se puede utilizar el algoritmo de Dijkstra para determinar si un grafo no dirigido es conexo
Si o si tiene q Tener peso
Dijkstra me devuelve lista con los los distancias, si hay algun infinito es que no es conexo

Siempre de los no visitados se fija menor distancia, si hay otra componente conexa todos las dist tentativas de esos nodos es ∞ (y efectivamente son los mínimos porque ya recorrí todos los vértices en la componente conexa de s)) y lo actualiza y sigue siendo ∞.

b) ¿Cómo se puede utilizar el algoritmo de Dijkstra para deteminar si un grafo **no dirigido** es conexo?. Justificar y escribir el pseudocódigo del algoritmo propuesto.