# Parcial de Programación 3

## 27 de noviembre de 2017

- Este parcial dura **3** horas y media y consta de 2 carillas. El total de puntos es **50**.
- **NO** se puede utilizar ningún tipo de material de consulta. Salvo que se indique lo contrario podrá usarse todo lo visto en el teórico, práctico y laboratorio sin demostrarlo, indicando claramente lo que se está usando.

**Se requiere:**

I) Numerar todas las hojas e incluir en cada una el **nombre, cédula de identidad, número de página y cantidad de hojas entregadas**.

II) Utilizar las hojas de **un solo lado** y escribir con lápiz.

III) Iniciar cada ejercicio en hoja nueva.

---

**Ejercicio 1 (20 puntos)**

Dada una secuencia $A = a_1, \ldots, a_n$ con $n$ caracteres.

Considere las siguientes definiciones:

Subsecuencia: $A'$ es una subsecuencia de $A$, si es una secuencia formada por caracteres pertenecientes a $A$, y además se cumple que el orden en que aparecen en $A$ se respeta en $A'$. Por ejemplo: Si $A$=f**ot**og**ra**fo, entonces **otra** es una subsecuencia de $A$, y **rato** no lo es.

Substring: $A'$ es un substring de $A$, si es una subsecuencia de $A$ que además cumple que los caracteres en $A'$ ocurren de forma consecutiva en $A$. Por ejemplo, si $A$=foto**grafo**, **grafo** es un substring de $A$, y otra no lo es.

Considere el siguiente pseudocódigo,

```
int f (char* A, char* B) {
    n = length(A)
    m = length(B)
    int Mt [n+1][m+1];
    int Mp [n+1][m+1];

    for(int i=0; i<=m; i++){
        Mt[0][i] = 0;
        Mp[0][i] = 0;
    }

    for(int i=0; i<=n; i++){
        Mt[i][0] = 0;
        Mp[i][0] = 0;
    }

    for(int i=1; i<=n; i++){
        for(int j=1; j<=m; j++){
            if (A[i] == B[j]){
                Mp[i][j] = 1+ Mp[i-1][j-1];
            }
            else{
                Mp[i][j] = 0;
            }
            Mt[i][j] = max(Mp[i][j], Mt[i-1][j],Mt[i][j-1]);
        }
    }

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

Listado 1: Algoritmo 1

(a) Realice el proceso de ejecución para la entrada $A = la$ $B = ala$, escriba en su respuesta solamente el estado final de las matrices $Mt$ y $Mp$.

(b) Indique qué se almacena en cada estructura auxiliar $Mt$ y $Mp$.

(c) ¿Qué problema resuelve?

(d) Plantee la recurrencia a partir de la cual se construyó el pseudocódigo.

(e) I. Escriba el pseudocódigo una solución recursiva del problema. Considere la siguiente firma para el pseudocódigo: `int f (int n, int m, int** Mp)`

II. Indique qué invocaciones serían necesarias para resolver el problema Mt(2,3). Sugerencia: utilice el árbol de recursión.

III. ¿Qué inconveniente le encuentra a esta solución?

(f) Determine la complejidad en espacio de memoria del Algoritmo 1. ¿Hay alguna forma de reducirla? Justfique.

**Ejercicio 2 (10 puntos)**

Para las siguientes afirmaciones sobre los Algoritmos Aleatorizados indicar si son verdaderas o falsas, justificando brevemente y dando un ejemplo:

(a) Solamente son útiles para diseñar algoritmos para problemas intratables.

(b) La aleatoriedad sirve, entre otras cosas, para diseñar algoritmos eficientes incluso ante el peor caso, y correctos con cierta probabilidad.

(c) Son deterministas en función de la entrada que reciben.

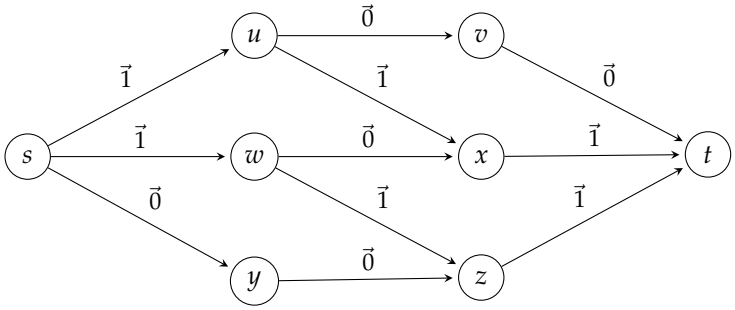
(d) La aleatoriedad sirve, entre otras cosas, para diseñar algoritmos correctos que en el caso promedio el orden de tiempo de ejecución es distinto al del peor caso.

(e) La idea de los Algoritmos Aleatorizados es estudiar cómo se comporta un algoritmo cualquiera en el caso promedio (no dar un ejemplo para esta parte).

**Ejercicio 3 (10 puntos)**

Sea $G(V, E)$ una red de flujo $s$-$t$ con capacidades enteras en sus aristas.

(a) Sea la instancia de la red con capacidad constante de valor uno y flujo según aristas




determinar su flujo máximo aplicando el algoritmo de Ford-Fulkerson a partir del flujo dado. [Mostrar los pasos del algoritmo mediante secuencias de grafos residuales y grafos con flujo.]

(b) Suponer que la red, con capacidad constante de valor uno, se construye a partir de un grafo bipartito al que se le agregan un nodo fuente, un nodo destino, aristas entre el nodo fuente y los nodos de una partición y aristas entre los nodos de la otra partición y el nodo destino. Formalmente, $G(V, E)$ se construye a partir de un grafo bipartito $G'(U \cup W, E' \subseteq U \times W)$ tal que $m = |U|$, $n = |W|$, y donde $V = U \cup W \cup \{s, t\}$ y $E = \{(u, w) | \{u, w\} \in E'\} \cup \{(s, u) | u \in U\} \cup \{(w, t) | w \in W\}$.

Determinar el largo máximo (cota superior) de los caminos de incremento (augmenting path) obtenidos al aplicar el algoritmo de Ford-Fulkerson en todo $E' \subseteq U \times W$ posible. [Establecer la cota en función de $m$ y $n$.]

**Ejercicio 4 (10 puntos)**

Para cada una de las siguientes preguntas, indique si la respuesta es "Sí". "No", o "No se sabe. Si se supiera resolvería el problema de decidir si $\mathcal{P} = \mathcal{NP}$". Dé una explicación breve (una o dos oraciones) de su respuesta. Sugerencia: utilice su conocimiento de a que clase pertenece cada uno de los problemas mencionados en las preguntas.

Definamos la siguiente versión del problema de la *Programación de Intervalos* (Interval Scheduling) como en formato de problema de decisión: dada una colección de intervalos en una linea de tiempo, y una cota $k$, ¿la colección contiene un subconjunto de intervalos no superpuestos de tamaño al menos $k$?

(a) ¿Es verdad que *Programación de Intervalos* $\leq_P$ *Cubrimiento de Vértices* (Vertex Cover)?

(b) ¿Es verdad que *Conjunto Independiente* (Independent Set) $\leq_P$ *Programación de Intervalos*?