Auxiliar 2: Algoritmos y Estructuras de Datos

Profesor: Patricio Poblete
Auxiliares: Iván Sipiran - Mauro Escobar
15 de abril de 2010

P1. Mediante un método recursivo, diseñe un algoritmo que permita invertir una lista. Suponga que una lista se representa como: `L==[H|T]`, donde `H` es la cabeza de la lista (primer elemento), y `T`, la cola de la lista (resto de los elementos). Suponga que ya está implementada la siguiente interfaz de la clase Lista:

```
public Lista(Object o); //retorna una lista con un solo elemento (o)
static public Object head(Lista l); //retorna el primer elemento de la lista
static public Lista tail(Lista l); //retorna la cola de la lista
static public boolean isEmpty(Lista l); //retorna true si la lista está vacía,
                //o false en caso contrario
static public Lista append(Lista l1, Lista l2); //retorna una nueva lista en
                //que primero vienen todos los elementos de l1, y luego,
                //todos los de l2.
```

P2. Se le llama *Distancia de Levenshtein* o *Distancia de Edición* al número mínimo de operaciones requeridas para transformar una cadena de caracteres en otra, realizando sólo las operaciones elementales: insertar un carácter, borrar un carácter o reemplazar un carácter por otro. Considere dos arreglos: $x_1x_2\ldots x_n$ e $y_1y_2\ldots y_m$, no necesariamente del mismo largo. Implemente un algoritmo que determine la distancia de edición entre ambos. Se recomienda usar programación dinámica.

P3. Considere una expresión del tipo

$$(x \vee \bar{z} \vee w) \wedge (\bar{x} \vee \bar{y}) \wedge (w \vee \bar{z}) \wedge (v \vee \bar{w} \vee \bar{x} \vee z).$$

Queremos estudiar la satisfacibilidad de este tipo de expresiones, es decir, saber si existe una asignación de valores de verdad a las variables involucradas, tal que la expresión sea verdadera. Implemente la función `boolean sat(int[][] exp)`, que verifica si la expresión `exp` es satisfacible o no. En el ejemplo, `exp` tiene la forma:

$$\begin{matrix} & v & w & x & y & z \end{matrix}$$
$$\texttt{exp} = \begin{bmatrix} 0 & 1 & 1 & 0 & -1 \\ 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 1 & -1 & -1 & 0 & 1 \end{bmatrix}$$

donde cada fila corresponde a una cláusula en la expresión (en el ejemplo, hay 4 clausulas), y cada columna corresponde a cada variable involucrada. Luego, `exp[i][j]` será 1 si en la i-ésima clausula aparece la variable j, y -1, si aparece la variable negada.
Utilice un método de backtracking.

P4. Resuelva la ecuación de recurrencia:

$$a_n = 5a_{n-1} - 6a_{n-2}, \qquad \text{con } a_0 = 0,\ a_1 = 1.$$