Proyecto 5: Árboles Binarios de Búsqueda Óptimos

I. Descripción General

Se debe programar el algoritmo visto en clases para encontrar el árbol binario de búsqueda óptimo que corresponda a una serie de datos ponderados con una cantidad que podria ser una probabilidad o un número arbitrario. La interfaz debe ser gráfica. Toda la programación debe realizarse en C sobre Linux. No se pueden cambiar las especificaciones de este documento.

II. Entrada y Salida

La interacción con el usuario se hará por medio de interfaces gráficas que deben ser desarrolladas con `GTK` y `Glade`. Debe haber un estilo uniforme en el *look & feel* de todas las interfaces. El programa tendrá la posibilidad de grabar archivos con los datos particulares de cada problema ingresado por el usuario, para que puedan ser cargados de nuevo y editados si el usuario así lo desea. El formato de este archivo queda a discreción de cada grupo de trabajo. Se espera que, el día de la revisión, cada grupo cuente con bastantes archivos de pruebas para mostrar las capacidades de su proyecto.

III. Menú Principal

En el Proyecto 0 de este curso se desarrolló un menú principal que podía "lanzar" diversos algoritmos. El algoritmo solicitado en este proyecto será implementado como un programa independiente que será ejecutado desde el menú mencionado. Se debe activar una "burbujita" con una descripción general de este algoritmo cuando el cursor se pose sobre la opción respectiva en el menú principal (*tooltip*).

IV. Árboles Binarios de Búsqueda Óptimos

Usando los algoritmos vistos en clase, este programa indicará como hacer el árbol binario de búsqueda de costo promedio mínimo. Se espera que la interfaz gráfica sea lo más flexible posible.

El usuario debe proporcionar la siguiente información:

- Número de llaves (n), se puede suponer que $n \leq 10$
- Para cada llave hay un texto o código por el cuál se harán las búsquedas. El usuario no necesariamente introducirá estas llaves en orden.
- Para cada llave hay un peso asociado expresado como un número real (con o sin decimales).

Trabajo extra opcional 1: Manejar $n > 10$

El programa debe ordenar los textos o códigos de las n llaves en orden lexicográfico. Los pesos asociados a cada llave se deben convertir en probabilidades (peso particular dividido entre la suma de todos los pesos) para ser usados en el resto del algoritmo.

La salida debe ser gráfica también, mostrando las tablas A y R similar a las hechas en clases para este tipo de problema.

Trabajo extra opcional 2: Desplegar de manera gráfica el árbol

V. Requisitos Indispensables

La ausencia de uno solo de los siguientes requisitos vuelve al proyecto "no revisable" y recibe un 0 de calificación inmediata:

- Todo el código debe estar escrito en C
- El proyecto debe compilar y ejecutar en Linux
- Todas las interfaces deben ser gráficas
- Se debe usar `GTK` y `Glade`
- El programa se debe invocar desde el menú desarrollado en el Proyecto 0.
- No debe dar "Segmentation Fault" bajo ninguna circunstancia.

Trabajo extra opcional 3: hacer la demostración en una máquina que levante Linux de manera real (puede ser dual), es decir no usar máquinas virtuales.

VI. Fecha de Entrega

Revisiones a las 11:30am el **Viernes 6 de Octubre**. Mande además un `.tgz` con todo lo necesario (fuentes, makefile, readme, etc.) a `torresrojas.cursos@gmail.com`. Ponga como subject: `I.O. - Proyecto 5 - Fulano - Mengano`, donde Fulano y Mengano son los 2 miembros del grupo.