Proyecto 1: Rutas Óptimas (Algoritmo de Floyd)

Viernes 12 de Septiembre

I. Descripción General

Se debe programar el Algoritmo de Floyd para obtener las rutas óptimas entre cualquier par de nodos en un grafo ponderado con distancias. La interfaz debe ser gráfica. La salida será un documento LaTeX que incluya el problema original, todas las tablas intermedias y su solución final detallada. Toda la programación debe realizarse en C sobre Linux. No se pueden cambiar las especificaciones de este documento.

II. 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**[1] que será ejecutado desde el menú mencionado. Se debe activar una "burbujita" con una descripción muy informativa de este algoritmo cuando el cursor se pose sobre la opción respectiva en el menú principal (*tooltip*). 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.

III. Manejo de Archivos

Habrá un par de botones que permitan grabar archivos con los datos particulares de cada problema ingresado por el usuario (el usuario selecciona el nombre que desea para el archivo), y para que puedan ser cargados de nuevo y editados si así se 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.

IV. Problema de las rutas más cortas

Usando el algoritmo de Floyd estudiado en clases, este programa resolverá el problema de encontrar las rutas más cortas entre cualquier par de nodos de un grafo con ponderaciones en los arcos.

Se espera que la interfaz gráfica sea lo más flexible posible. Como dato inicial, el usuario indica la cantidad de nodos del grafo. Con esto se activa una interfaz donde el usuario da las distancias directas entre ellos. Debe haber un mecanismo para indicar el caso en que las distancias son infinitas (*i.e.*, no ruta directa) entre 2 nodos. Este es el valor por defecto para todas las entradas de la tabla al inicio (excepto la diagonal de la tabla que tendrá siempre ceros)[2]. Para este proyecto, el usuario podrá indicar entre 1 y 8 nodos. Nótese que esta tabla ingresada por el usuario corresponde a la Tabla $D(0)$ de acuerdo a la notación usada en clases.

Por la naturaleza del problema, la tabla será cuadrada y cada nodo aparecerá en una fila y en una columna, y estarán identificados con un nombre (que por defecto serán A, B, C, etc.). Dichos nombres son editables, tanto en la fila como en la columna. Al cambiar un nombre en una fila, su contraparte en la columna se actualizará automáticamente, y viceversa.

Trabajo extra opcional 1: hacer que se manejen apropiadamente más de 8 nodos.

V. Reporte

Cuando el usuario considere que su problema ya está listo, presionará algún botón para arrancar la ejecución del algoritmo. Esto generará un documento con todo el proceso de solución.

Se debe generar un archivo "`.tex`" que será compilado con `pdflatex` o algún equivalente para que se produzca un archivo "`pdf`" que debe ser desplegado usando `evince` (o algún equivalente) en modo presentación.

La invocación de todos estos comandos debe realizarse internamente desde su programa, de manera transparente para el usuario. Deben quedar disponibles tanto el `pdf` como el fuente de LaTeX. Recuerde que la calidad final del documento es muy importante para la calificación de este proyecto.

El documento debe incluir por lo menos:

1. **Portada**: se identifica claramente a los miembros del grupo, el curso, el semestre y el nombre del proyecto.
2. **Algoritmo de Floyd**: explicar el algoritmo y hacer una semblanza del autor original.
3. **Problema**: presentar el problema en la forma de un grafo (investigue y seleccione algún `package` de LaTeX para dibujar grafos). Se deben usar los nombres de nodos ingresados por el usuario.
4. **Tablas Iniciales**: se muestra la $D(0)$ y la tabla P inicial. Los infinitos deben ser manejados apropiadamente.
5. **Tablas Intermedias**: se deben presentar las tablas D y P intermedias de la ejecución del algoritmo. Cada paso y tabla deben estar rotulados apropiadamente. Cualquier entrada de las tablas D o P que hayan cambiado con respecto a la inmediata anterior debe estar marcada en algún color llamativo.
6. **Distancias y Rutas Óptimas**: después de las últimas tablas se debe mostrar la ruta óptima (con todas las

[1] No puede ser una función del programa de menú. La interfaz de este programa se desarrolla con Glade en un archivo independiente.

[2] No usen -1 como infinito, sean creativos

paradas intermedias) y la distancia mínima entre todo par de nodos del grafo original.

VI. Requisitos Indispensables

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

- El programa se debe invocar desde el menú desarrollado en el Proyecto 0, el cual debe estar correcto (considerar cualquier observación que se les haya hecho).
- La colaboración entre grupos se considera fraude académico.
- Todo el código debe estar escrito en C
- Recuerde que usar IA para completar cualquier parte de su trabajo es fraude académico y tendrá muy serias consecuencias.
- El proyecto debe compilar y ejecutar en Linux. Todo debe estar **integrado**, explicaciones del tipo "*todo está bien pero no pudimos pegarlo*"[3] provocan la cancelación automática de la revisión.
- Para el momento de la revisión, su programa no debe tener "prints" de depuración. Luce muy poco profesional.
- Todas las interfaces deben ser gráficas, robustas y de mucha calidad.
- Se debe usar `GTK` y `Glade`. Es inaceptable hacer una interfaz de texto desplegada en un *widget*.
- No debe dar "Segmentation Fault" bajo ninguna circunstancia.
- La demostración debe hacerse en una máquina que levante Linux de manera real (puede ser dual), es decir no usar máquinas virtuales. No se puede levantar de un disco o memoria externa. No se pueden prestar máquinas entre grupos.

VII. Fecha de Entrega

Revisiones a las 11:30am del **Viernes 12 de Septiembre** en la oficina del profesor. Mande además un `.tgz` con todo lo necesario (fuentes, makefile, readme, etc.) a `torresrojas.cursos.04@gmail.com`. Ponga como subject: `[IO] Proyecto 1 - Fulano - Mengano`, donde Fulano y Mengano son los miembros del grupo.

Mucha suerte...

[3] esto incluye los supuestos casos cuando alguien del grupo de trabajo no hizo su parte – el profesor no está interesado en sus problemas de organización.