Proyecto 4: Índice de Poder de Banzhaf

Martes 3 de Junio

I. Descripción General

En un sistema de votos asimétricos, cada votante tiene una cantidad diferente de votos. El *Índice de Poder de Banzhaf* (IPB) es una métrica del poder real de cada votante tomando en cuenta la proporción de votos críticos (*swing votes*) que cada uno tenga. En este proyecto, calcularemos estos índices para el modelo de votación ingresado. Toda la programación debe hacerse en Lenguaje C sobre Linux, usando GTK y Glade.

II. Entrada

Con una interfaz gráfica de calidad, el programa solicita la cantidad n ($3 \leq n \leq 12$) de votantes. Esto despliega n campos enteros donde el usuario introduce la cantidad de votos v_i de cada participante. Por defecto todos tienen un voto. Se debe validar que estos valores sean enteros positivos. De manera similar, se solicita la cantidad K de votos necesarios para ganar una votación (validar únicamente que sea un entero mayor o igual a 0).

Al inicio, cada votante tendrá un color asignado, el cual será mostrado como parte de la interfaz. Debajo del campo de los votos de cada votante se desplegará eventualmente el IPB de cada uno.

Trabajo extra opcional 1: Permitir que el usuario edite en una subventana el color asignado a cada votante.

Trabajo extra opcional 2: Solicitar un nombre o etiqueta para cada uno de los n votantes que debe ser usado en todos los despliegues posteriores.

El programa desplegará el modelo de la votación en el estilo $(K; v_1, v_2, ..., v_n)$ y un rectángulo del ancho máximo de la pantalla donde los votos de cada votante serán representados en un área de su color proporcional a su cantidad de votos, de manera similar a la **Figura 1**.

Figura 1

Trabajo extra opcional 3: Aparte del rectángulo obligatorio, despliegue la situación de una manera creativa al estilo de un "parlamento" (no se aceptan gráficos de pastel). Idealmente, se muestra cada voto de manera independiente. Por ejemplo, considere la **Figura 2**.

Habrá un botón de "ejecución" que empieza el proceso de solución.

III. Proceso

Se deben reutilizar funciones del proyecto previo que resolvía el problema de la Suma de Subconjuntos usando *backtracking*[1] para encontrar todas las coaliciones ganadoras. A su vez, dentro de ellas se deben encontrar los votos críticos necesarios para calcular el IPB de cada votante.

IV. Salida

En una zona de la interfaz con *scroll* vertical se muestran **todas** las coaliciones ganadoras (*i.e.*, un subconjunto de los votantes que reúnan una cantidad de

[1] el profesor revisará esto

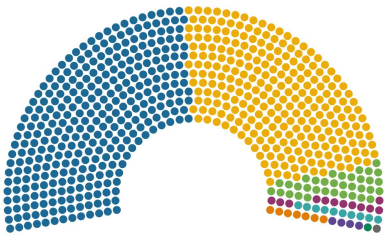

Figura 2

votos mayor o igual a K). Recuerden que la cantidad de coaliciones ganadoras podría ser alta. En esta zona, cada coalición ganadora será mostrada como un rectángulo de las mismas proporciones y tamaño que el rectángulo que muestra todos los votos (ver **Figura 1**). Dentro de este rectángulo los votos críticos deben estar marcados de alguna forma[2].

Trabajo extra opcional 4: En vez del rectángulo, despliegue cada coalición ganadora, y sus votos críticos sobre el gráfico del parlamento (ver **Figura 2**). Por ejemplo, mostrar todo el parlamento pero con los asientos de los que no son miembros de la coalición en gris, los miembros de la coalición en sus colores y los votos críticos más brillantes (o con estrellas en vez de círculos).

El programa indicará la cantidad total de votos críticos en el modelo. En los campos establecidos debajo de los votos de cada votante se mostrará el IPB de cada uno, como número real (*e.g.*, 0.2353) y como la fracción de votos críticos que poseen (*e.g.*, $\frac{4}{17}$).

V. Requisitos Indispensables

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

- La colaboración entre grupos se considera fraude académico.
- Todo el código debe estar escrito en C (no C++).
- 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.
- Todas las interfaces deben ser gráficas.
- Se debe usar `GTK` y `Glade`.
- La presentación debe ser de mucha calidad.
- No debe dar "Segmentation Fault" bajo ninguna circunstancia.
- Hacer la demostración en una máquina que levante Linux de manera real (puede ser dual), es decir no usar máquinas virtuales.

[2] ¿colores más brillantes?

[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.

VI. Fecha de Entrega

Revisiones a las 11:30 am el **Martes 3 de Junio** en la oficina del profesor. Mande además un `.tgz` con todo lo necesario (fuentes, makefile, readme, etc.) a `torresrojas.cursos.05@gmail.com`. Ponga como subject: `[AA] - Proyecto 4 - Fulano - Mengano`, donde Fulano y Mengano son los miembros del grupo.