Reporte sobre el proyecto de Modelos de Optimización 2

Laura Tamayo LAURA.TAMAYO@ESTUDIANTES.MATCOM.UH.CU
Grupo C411
Yasmin Cisneros Y.CISNEROS@ESTUDIANTES.MATCOM.UH.CU
Grupo C411
Jessy Gigato J.GIGATO@ESTUDIANTES.MATCOM.UH
Grupo C411
Nelson Mendoza Alvarez NELSON.MENDOZA@ESTUDIANTES.MATCOM.UH
Grupo C411

Problema

Conformación automática de grupos en la facultad:

Una de las primeras tareas de Secretaría Docente de la facultad cuando matriculan los estudiantes de primer año es formar los grupos. Los grupos deben ser los más homogéneos posibles. Al formar los grupos hay varios criterios que se deben tener en cuenta. Esos criterios pueden variar de un año a otro, pero usualmente es conveniente que sean lo más parecidos posibles con respecto a varios criterios, como la relación entre varones y hembras, becados y no becados, las procedencia de los alumnos, etc.
El objetivo de este proyecto es diseñar e implementar una aplicación que le permita a las personas encargadas de confeccionar los grupos hacerlo de la mejor manera posible. "De la mejor manera posible" depende de los intereses del año en cuestión, así que la propuesta debe ser lo suficientemente flexible como para incorporar diversos criterios.

Un primer enfoque

Al leer el objetivo del problema se aprecia que puede resultar conveniente modelar un problema de asignación para dar solución al mismo y obtener así los grupos deseados. Un problema de asignación consiste en encontrar un emparejamiento de costo óptimo en un grafo bipartito ponderado. Generalmente para resolver este tipo de problemas se utilizan mecanismos como el algoritmo húngaro o el método simplex y se obtienen buenos resultados.

Entonces solo quedaría modelar el problema para aplicar uno de los algoritmos mencionados. Lo más intuitivo sería poner de un lado de la partición los nodos que representan los grupos y a los estudiantes del otro, desafortunadamente al existir más de una categoría con respecto a cómo clasificar a los estudiantes se deben añadir aristas que hacen que el grafo deje de ser bipartito volviendo el problema un emparejamiento en un grafo cualquiera por lo que los algoritmos mencionados previamente dejan de ser aplicables.

Refinando el enfoque anterior

La idea de ver el problema como un problema de asignación puede seguir siendo aplicada si se le da una prioridad a cada criterio por el que se quiere clasificar a los estudiantes y se resuelve

el problema para cada categoría teniendo en cuenta la asignación que se tiene hasta el momento.

Para resolver el problema con este enfoque no es necesario utilizar algoritmos de gran complejidad temporal como el húngaro, se puede resolver el problema con trabajar con listas de la siguiente forma:

Supongamos que se tiene una lista con todos los estudiantes agrupados con respecto a los distintos valores de una categoría, entonces para generar la respuesta deseadas bastaría con tomar estudiantes de cada uno de estos grupos y distribuirlos de manera pareja, si hubiera más de una categoría basta con ordenar cada grupo por los valores de la siguiente categoría y así sucesivamente hasta obtener una lista donde se tienen todos los estudiantes distribuidos de una forma pareja. En otras palabras, se particionaría el conjunto con respecto a la categoría actual (la cual tendrá mayor prioridad que la siguiente) y se ordenara el grupo de estudiantes con respecto a la misma y luego por cada uno de los subconjuntos anteriores se realiza el mismo proceso con la categoría siguiente, efectuandose esto de forma recursiva.

El siguiente pseudocódigo muestra el algoritmo descrito:

```
ordenar por categoría (estudiantes, categorías)
        if categorías.length == 0
                return estudiantes
        lista = ordenar y dividir(estudiantes, categorías[0]) #agrupa con respecto a una categoría
        resultado = []
        for item in lista:
                resultado ++ ordenar por categoría(item, categorías[1:])
        return resultado
```

Figura 1: Pseudocodigo del algoritmo

El metodo "ordenar_y_dividir" devuelve una lista de listas de estudiantes, esta se particiona por categorías iguales. Este método presenta complejidad temporal $O(m * logm)$ ya que:
m: representa la cantidad de estudiantes de la lista
La ordenación de los elementos se realiza en: $O(m * logm)$ y su división en $O(m)$
Luego por ley de la suma: $O(m * logm) + O(m) = O(m * logm)$

La complejidad temporal de este algoritmo es $O(n * m * \log m)$ donde n es la cantidad de categorías y m la cantidad de estudiantes, ya que en el peor de los casos si cada uno de los elementos de las categorías seleccionadas son iguales este entrara las n veces de las categorías y realizara la ordenación con el conjunto de los m estudiantes.

Luego de Generar esta lista que ya estará ordenada por el conjunto de categorías se reparte entre los N grupos, asignando cada uno de los valores de uno en uno por cada grupo.

Tutorial de uso de la aplicación

A continuación se presentan los pasos a seguir para utilizar la aplicación:

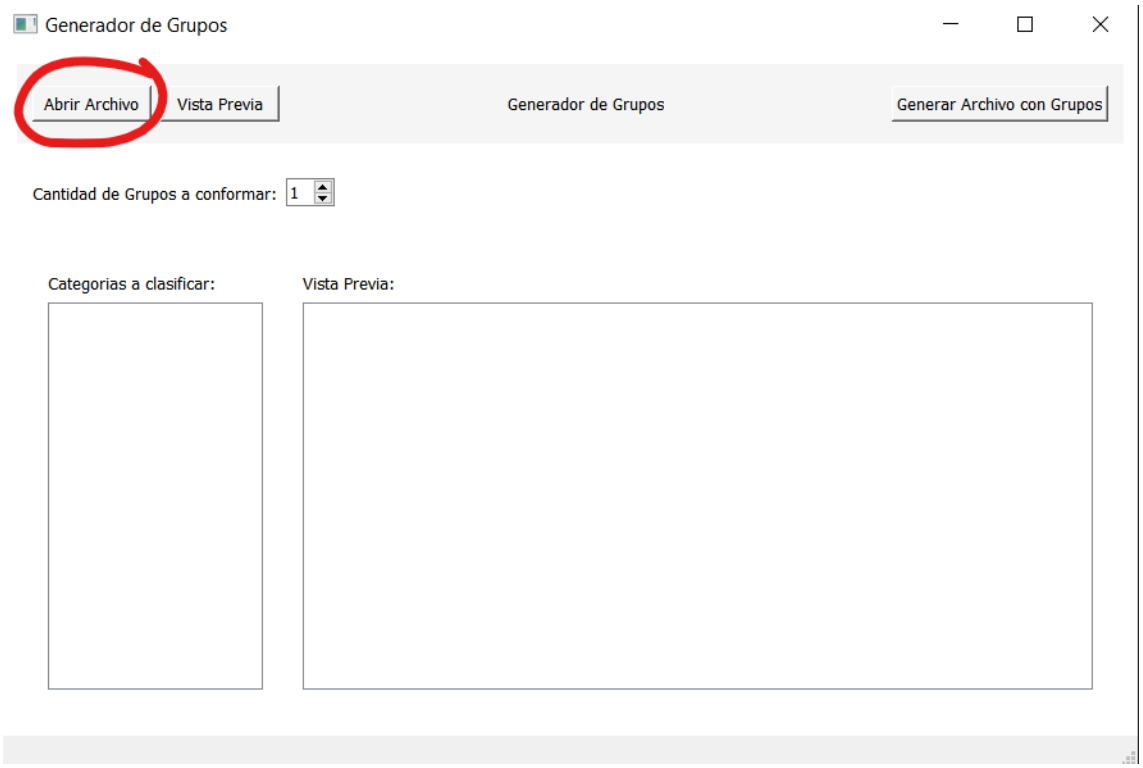

Figura 2: Para su uso primeramente debe de agregar el archivo excel (.xlsx)

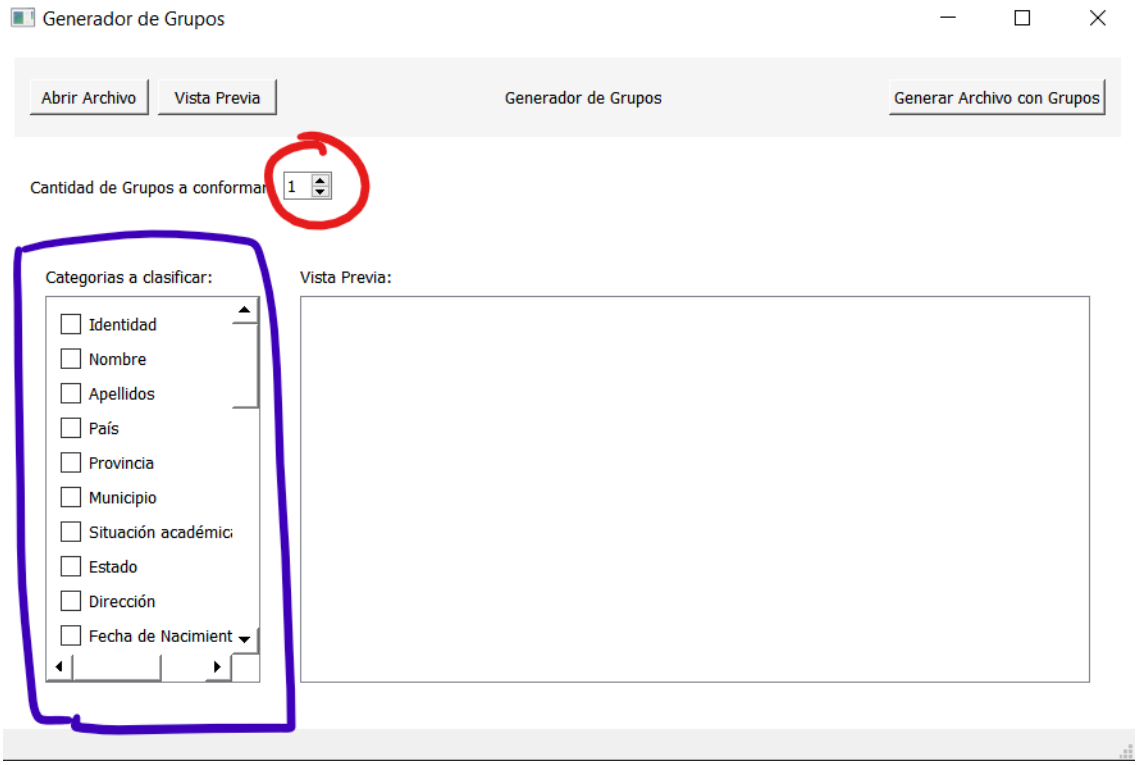

Figura 3: Se desplegara el menú de categorías por las cuales se puede seleccionar para realizar una repartición de los grupos de forma homogénea tomando como referencia las mismas. Ademas de la selección de la cantidad de grupos deseados

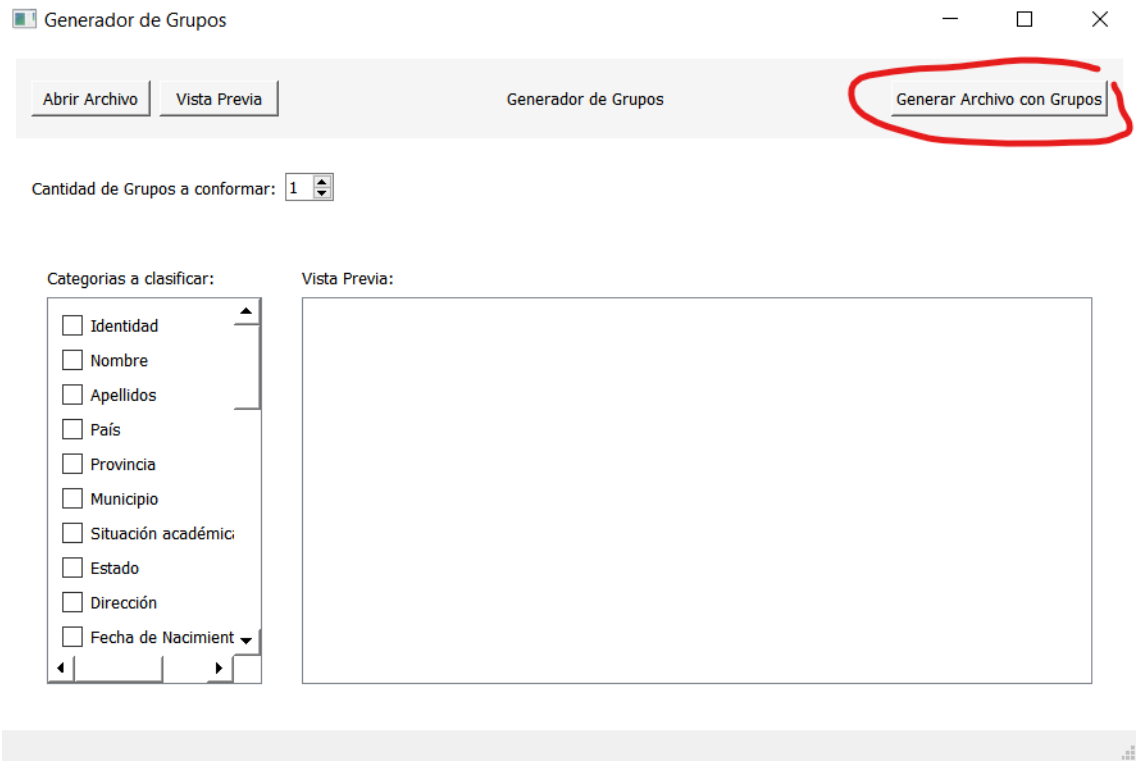

Figura 4: Una vez se guarden los archivos, estos serán guardados en la carpeta "Grupos" que se generará automáticamente

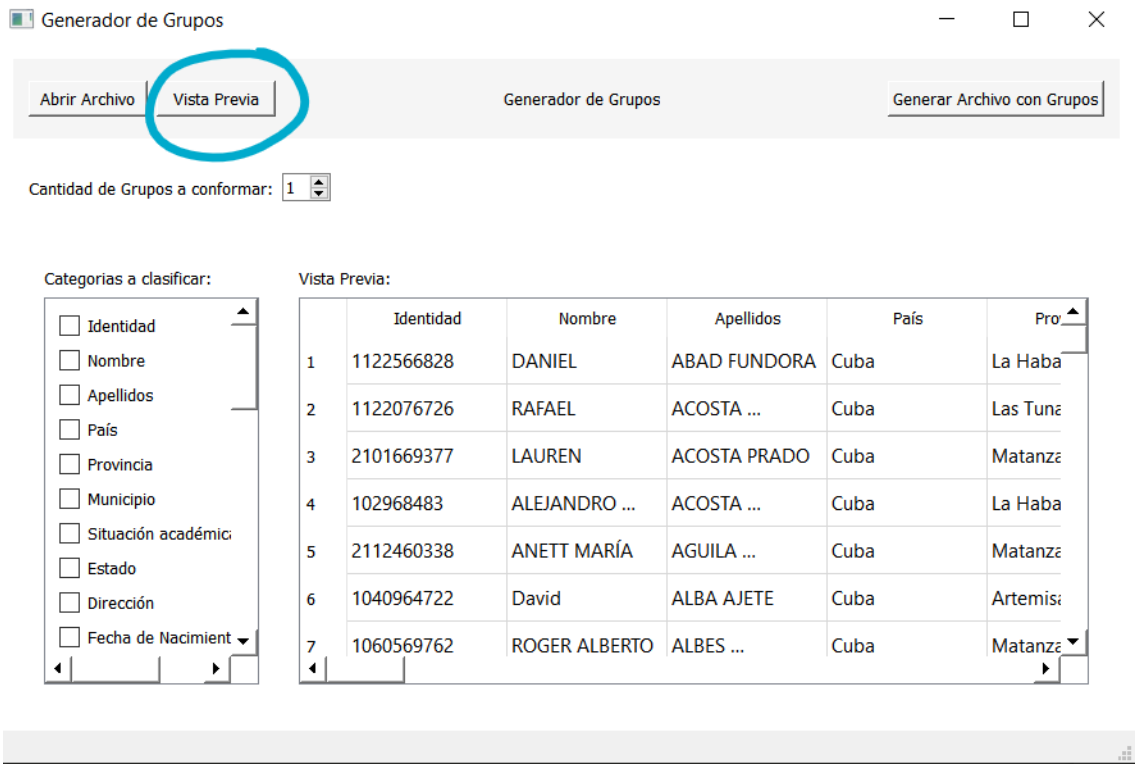

Figura 5: Para chequear si el archivo que abrió es el correcto siempre puede previsualizarlo