Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2023.

Analizador de frecuencias de sonido

202201117 – Miguel Ricardo Galicia Urrutia

Resumen

El proyecto consta en la lectura de un archivo XML el cual contendrá información acerca de señales de audio, las cuales tendrán los atributos de tiempo y amplitud. El cual al analizarse se almacenará en una lista simple enlazada y esta se ordenará de manera ascendente según el tiempo y la amplitud. Luego el programa realizara la creación de una matriz binaria la cual convertirá los valores a 1 y 0, se creará con el valor 1 siempre y cuando la señal sea diferente a 0. Luego de esto creara la matriz reducida y sumara los datos de la lista original. Luego redactara el archivo de salida XML y creara imágenes acerca de las listas y las ordenara como matriz. En una de sus opciones esta mostrara los datos del estudiante que realizo el código del programa. Y finalmente tendrá una opción de inicializar sistema, cuya función específica es limpiar la memoria de donde se encuentran las señales, es decir eliminara los datos de las listas.

Palabras clave

- Listas
- Patrones
- Señales
- Amplitud
- Matriz reducida

Abstract

The project consists of reading an XML file which will contain information about audio signals, which will have the attributes of time and amplitude. Which when analyzed will be stored in a simple linked list and this will be ordered in ascending order according to time and amplitude. Then the program will create a binary matrix which will convert the values to 1 and 0, it will be created with the value 1 as long as the signal is different from 0. After this it will create the reduced matrix and add the data from the original list. Then you will write the XML output file and create images about the lists and sort them as an array. In one of its options this will show the data of the student who made the code of the program. And finally, you will have an option to initialize system, whose specific function is to clean the memory of where the signals are located, that is, it will eliminate the data from the lists.

Keywords

1. Lists
2. Patterns
3. Signals
4. Amplitude
5. Reduced matrix

Introducción

El programa de Python que presentamos en este documento lee datos de ondas de sonido desde un archivo XML y genera matrices binarias con los datos de las señales. Estas matrices se agrupan en patrones, que luego se guardan en un archivo XML. El programa también implementa Graphiz para la implementación de la matriz.

El programa es útil para el análisis de datos de audio. Las matrices binarias proporcionan una representación compacta de los datos de las señales, que luego se pueden agrupar en patrones. Estos patrones pueden usarse para identificar diferentes tipos de sonidos, como voces, instrumentos musicales o ruido.

El programa es fácil de usar. El usuario simplemente debe proporcionar un archivo XML con los datos de las señales. El programa leerá los datos del archivo XML y generará las matrices binarias, los patrones y el archivo XML de los grupos.

Desarrollo del tema

El Centro de Investigaciones de la Facultad de Ingeniería está experimentando un método para lograr comprimir señales de audio, por lo que se han enfocado en dos parámetros propios de las ondas de sonido: Frecuencia y Amplitud, estos parámetros describen la señal de audio en función del tiempo. La frecuencia es la cantidad de ciclos que se realizan en un segundo y se mide en Hertz (Hz), mientras que la amplitud representa la altura de la onda y hace referencia a la intensidad del sonido, la amplitud se mide en decibelios (Db). Para implementar esta metodología de agrupamiento, el Centro de Investigación ha definido una matriz de tiempo (t), amplitud (A) y frecuencias (f) para distintas señales de audio S [t][A], transformarla en una matriz de patrones de frecuencia y agrupar las tuplas con el mismo patrón. La matriz de patrones de frecuencias para una matriz de frecuencias dada es la matriz binaria que indica en qué tiempos y amplitudes hay frecuencias en la señal de audio en estudio

Para su implementación se crearon las listas:

a. Lisata Senales
b. Listas Datos
c. Listas Grupos
d. Listas Patrones

El código corresponde a un proyecto de análisis de señales implementado en Python utilizando la biblioteca Tkinter para la interfaz gráfica y otras clases definidas en el proyecto para gestionar y procesar los datos de las señales. El objetivo principal del proyecto es cargar archivos XML que contienen datos de señales, procesar estos datos para generar análisis y gráficas, y realizar operaciones específicas en los datos.

A continuación, se explica el funcionamiento del proyecto y su estructura:

Importación de Librerías y Clases: El código comienza importando las librerías y clases necesarias para su funcionamiento, incluyendo Tkinter para la interfaz gráfica, clases relacionadas con listas de señales, datos y patrones, y clases de modelos como senal y dato.

Variables Globales y Definición de Funciones: Se definen variables globales que se utilizarán a lo largo del programa, como archivo, contador, route y root. Luego, se define la función menu(), que será el punto de entrada para interactuar con el programa. El menú ofrece varias opciones, como cargar un archivo, procesar datos, generar gráficas, entre otras.

Cargar Archivo: La opción "Cargar Archivo" permite al usuario seleccionar un archivo XML. La función askopenfilename de la librería Tkinter se utiliza para abrir una ventana de diálogo de selección de archivos. Una vez seleccionado el archivo, se parsea utilizando ET.parse() y se almacena la raíz del XML en la variable global root.

Procesar Archivo: La opción "Procesar Archivo" procesa los datos del archivo XML previamente cargado. Se itera sobre las etiquetas "senal" y se extraen atributos como el nombre, tiempo y amplitud de la señal. Los datos de cada señal se procesan y se crean instancias de la clase dato, que se almacenan en listas temporales (lista_datos_temporal y lista_binaria_temporal). Estas listas almacenan tanto los datos originales como los binarizados (si el valor es distinto de 0).

Generar Gráfica Original: La opción "Generar Gráfica" utiliza la función generar_grafica_original para generar una gráfica de la señal original. Esta función recibe el nombre, tiempo y amplitud de la señal y utiliza librerías de visualización como Matplotlib para crear la gráfica.

Inicializar Sistema: La opción "Inicializar Sistema" reinicia las listas temporales y otras variables globales para preparar el sistema para un nuevo análisis.

Datos del Estudiante: La opción "Datos del Estudiante" muestra información sobre el autor del proyecto, incluyendo su nombre, número de estudiante, curso y semestre.

Finalización del Programa: Una vez que el usuario selecciona la opción "Salir", el programa muestra un mensaje de despedida y finaliza su ejecución.matriz.

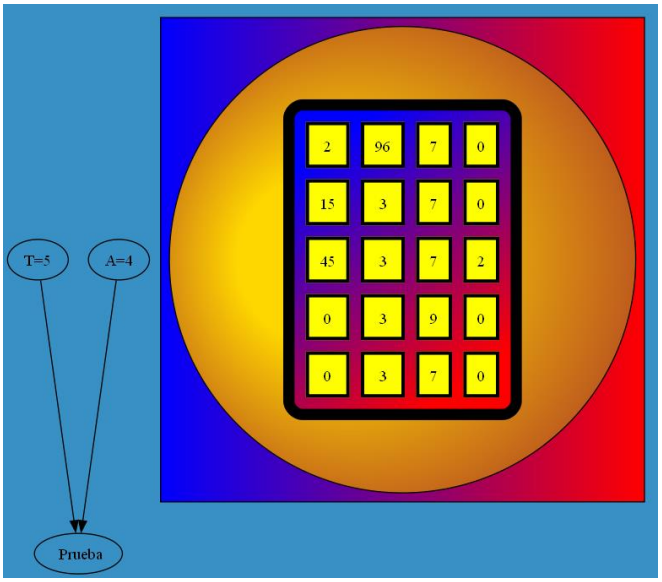

Figura 1. Matriz de pruebas generada por graphiz.

Tabla I.

Módulos del Programa.

Modulos	Ejemplo
Nodos	Nodo_senal
Listas	Lista_datos
Modelos	grupos
Archivos XML	Test1
App.py	Archivo principal

Conclusiones

Este proyecto demuestra el uso de la biblioteca Tkinter para crear una interfaz gráfica en Python y cómo puede ser utilizada para cargar archivos XML, procesar datos de señales y realizar análisis. Además, muestra la importancia de organizar el código en funciones y clases para una mejor modularidad y reusabilidad del código. El enfoque modular facilita la adición de nuevas características y la gestión de datos complejos en proyectos de análisis de señales y aplicaciones similares.

Referencias bibliográficas

- GitHub
    - Ejemplos realizados en Clase
    - https://github.com/Erwin14k/IPC2_2S23_CLASSES
- Graphiz
    - .Documentation
    - https://graphviz.org/documentation/
- Manejo de realises
    - Github
    - https://docs.github.com/es/repositories/releasing-projects-on-github/managing-releases-in-a-repository

C. J. Date, (1991). *An introduction to Database Systems.* Addison-Wesley Publishing Company, Inc.