Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2024

SISTEMA DE ANÁLISIS Y REDUCCIÓN DE MATRICES UTILIZANDO ARCHIVOS XML EN PYTHON

202300848 – Brayan Emanuel Garcia

Resumen

Este ensayo expone el desarrollo de un sistema informático diseñado para leer, procesar y reducir matrices almacenadas en archivos XML, utilizando el lenguaje de programación Python. La novedad de este proyecto radica en su capacidad para optimizar el procesamiento de datos estructurados, lo que tiene aplicaciones tanto en el contexto académico como en la industria. A nivel técnico, el sistema utiliza estructuras de datos como listas enlazadas y clases específicas para manejar la manipulación de matrices y la creación de grafos. La importancia de este tipo de sistemas radica en su versatilidad y capacidad de adaptación a diferentes contextos, como el análisis de grandes volúmenes de datos. Las conclusiones destacan cómo el uso de lenguajes interpretados como Python puede facilitar la creación de soluciones eficientes y escalables para el manejo de información estructurada en formatos XML.

Palabras clave

Máximo cinco palabras que servirán para identificar el estudio realizado.

Abstract

Abstract

This essay discusses the development of a computer system designed to read, process, and reduce matrices stored in XML files using the Python programming language. The novelty of this project lies in its ability to optimize the processing of structured data, which has applications in both academic and industrial contexts. Technically, the system utilizes data structures such as linked lists and specific classes to handle matrix manipulation and graph creation. The importance of such systems lies in their versatility and adaptability to different contexts, such as analyzing large volumes of data. The conclusions highlight how the use of interpreted languages like Python can facilitate the creation of efficient and scalable solutions for handling structured information in XML formats.

Keywords

Matrices, XML, Python, data processing, graphs.

Introducción

El análisis y procesamiento de grandes cantidades de datos es un tema de creciente relevancia en el contexto tecnológico actual. Los datos estructurados, que se almacenan en formatos como XML (Extensible Markup Language), permiten una organización clara y jerárquica de la información, facilitando su manipulación y análisis. Este ensayo aborda el desarrollo de un sistema basado en Python que tiene como objetivo principal leer, procesar y reducir matrices provenientes de archivos XML. La implementación de este sistema no solo permite mejorar la eficiencia en la manipulación de datos, sino que también ofrece una solución flexible y adaptable a diferentes escenarios. La importancia de este proyecto radica en su aplicación práctica en áreas como el análisis de datos, la optimización de recursos y la visualización de información a través de grafos, proporcionando una herramienta robusta para la investigación académica y la industria.

Desarrollo del tema

Subtema 1: Lectura y Procesamiento de Archivos XML

El sistema inicia con la lectura de archivos XML que contienen matrices. Para este propósito, se emplea la clase LectorXML, que extrae los datos almacenados en los archivos y los convierte en estructuras manejables en Python, como listas enlazadas. La elección de listas enlazadas para el almacenamiento de matrices permite una gestión eficiente de grandes volúmenes de datos, ya que facilita la inserción y eliminación de elementos sin necesidad de reestructurar la memoria. Este enfoque es particularmente útil para la manipulación dinámica de matrices, optimizando el rendimiento general del sistema.

Subtema 2: Análisis y Reducción de Matrices

La clase AnalizadorMatrices se encarga del procesamiento de las matrices. Este módulo identifica patrones en las matrices y lleva a cabo la reducción de su tamaño, un proceso crucial para optimizar el rendimiento del sistema. La reducción se realiza mediante la eliminación de redundancias y la agrupación de datos similares, lo que no solo disminuye el espacio de almacenamiento requerido, sino que también mejora la velocidad de acceso a la información. La capacidad de reducir matrices sin perder datos críticos permite un análisis más eficiente y preciso de grandes conjuntos de datos.

Subtema 3: Generación de Grafos

Uno de los aspectos más interesantes del proyecto es la generación de grafos, que representan visualmente las matrices procesadas. Utilizando la clase Grafo, el sistema crea nodos y aristas que facilitan la visualización de la estructura interna de las matrices. Esta representación gráfica es particularmente útil para el análisis y la toma de decisiones en contextos donde la visualización de datos facilita la interpretación y comprensión de patrones complejos. La generación de grafos también permite identificar relaciones entre diferentes elementos de la matriz, lo que puede ser crucial para la optimización de procesos y la identificación de áreas de mejora.

Subtema 4: Escritura de Matrices Reducidas

El proceso final del sistema es la escritura de las matrices reducidas en un nuevo archivo XML, utilizando la clase EscritorXML. Este archivo de salida contiene las matrices procesadas y optimizadas, listas para ser utilizadas en aplicaciones posteriores o para análisis adicionales. La capacidad de generar archivos XML optimizados garantiza que los datos reducidos mantengan una estructura

organizada y accesible, lo que facilita su integración con otras herramientas y sistemas.

Conclusiones

Este proyecto demuestra cómo el uso de herramientas modernas de programación, como Python, facilita la creación de sistemas eficientes para el procesamiento de grandes volúmenes de datos. El enfoque utilizado para la lectura, análisis, reducción y visualización de matrices mediante archivos XML es una solución robusta que puede aplicarse en múltiples áreas, desde la ciencia de datos hasta la ingeniería de software. Además, el sistema es adaptable, lo que permite su escalabilidad y personalización en distintos contextos. Finalmente, el uso de lenguajes interpretados como Python permite reducir tiempos de desarrollo sin comprometer la eficiencia, lo que lo convierte en una opción viable para proyectos que requieren procesamiento intensivo de datos.

Apéndices

Apéndice A: Ejemplo de Archivo XML

```
<matrices>
  <matriz id="1">
    <fila>1 2 3</fila>
    <fila>4 5 6</fila>
    <fila>7 8 9</fila>
  </matriz>
  <matriz id="2">
    <fila>1 0 0</fila>
    <fila>0 1 0</fila>
    <fila>0 0 1</fila>
  </matriz>
</matrices>
```

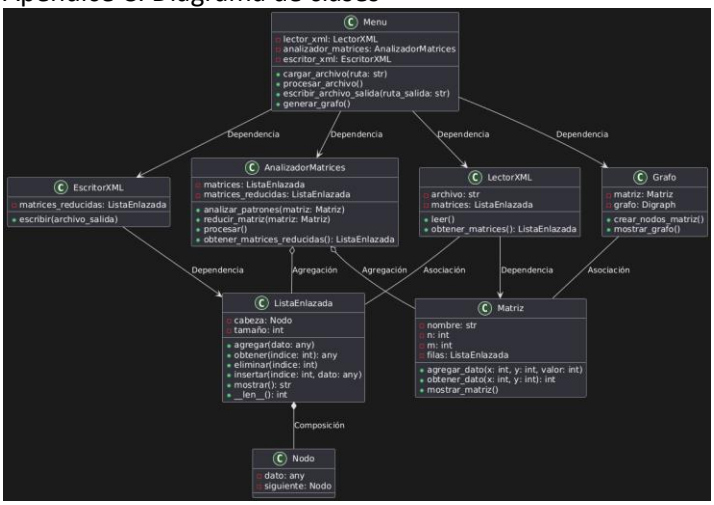
Apéndice B: Codigo de clase LectorXML

```
import xml.etree.ElementTree as ET

class LectorXML:
  def __init__(self, archivo):
    self.archivo = archivo

  def leer_matrices(self):
    arbol = ET.parse(self.archivo)
    raiz = arbol.getroot()
    matrices = []
    for matriz in raiz.findall('matriz'):
      datos = [list(map(int, fila.text.split())) for fila in matriz.findall('fila')]
      matrices.append(datos)
    return matrices
```

Apéndice C: Diagrama de clases



Apéndice D: Diagrama de actividades