Parcial 3: Datos y Algoritmos

Juan Andrés Young Hoyos, Joseph Saldarriaga, Sofia Flores Suárez

1 Importaciones

Las siguientes bibliotecas son necesarias para el procesamiento de datos, construcción de gráficos y visualización:

- `pandas`: Manipulación de archivos CSV.
- `networkx`: Manipulación de grafos.
- `matplotlib.pyplot`: Visualización de grafos.

2 Clase Nodo

La clase `Node` representa un nodo para una lista enlazada, que incluye un puntero al siguiente nodo.

3 LinkedList Class

La clase `LinkedList` implementa una lista enlazada. Incluye métodos para agregar (`append` y `prepend`) y para convertir una lista enlazada a una lista de Python.

4 Funciones

4.1 load_data

Esta función lee un archivo CSV con datos de películas utilizando la librería `pandas` y devuelve solo las columnas relevantes: `Series_title`, `Director`, `Star_1`, `Star_2`, `Star_3`, `Star_4`.

4.2 build_graph

Construye un grafo usando `networkx`, donde los nodos son directores y actores, y las aristas representan colaboraciones en películas. También utiliza un `defaultdict` para almacenar una lista enlazada de actores por película.

4.3 bfs_shortest_path

Implementa la búsqueda de la ruta más corta utilizando BFS (*Breadth-First Search*) en el grafo.

4.4 dfs_paths

Implementa la búsqueda de todas las rutas posibles utilizando DFS (*Depth-First Search*) en el grafo.

4.5 visualize_graph

Visualiza el grafo utilizando `matplotlib.pyplot`.

5 Descripción del Código

Este código realiza varias operaciones relacionadas con la construcción y análisis de un grafo basado en una base de datos de películas, y luego proporciona funciones para visualizar el grafo y encontrar caminos entre actores en el grafo. A continuación, se describe cada parte del código:

1. **Importación de bibliotecas**: Las primeras líneas importan las bibliotecas necesarias para el procesamiento de datos, construcción de gráficos y visualización.

2. **Definición de Clases**: Las clases `Node` y `LinkedList` definen nodos y una lista enlazada, respectivamente. Estas estructuras se utilizan para almacenar información sobre los actores y sus conexiones en las películas.

3. **Funciones para cargar y construir datos**:

   - `load_data(file_path)`: Esta función carga los datos desde un archivo CSV y selecciona las columnas relevantes para el análisis (título de la película, director y actores principales).
   - `build_graph(data)`: Construye un grafo donde los nodos representan directores y actores, y las aristas representan su participación en películas. También crea un diccionario que asigna cada película a una lista enlazada de actores y directores asociados.

4. **Funciones de búsqueda de caminos**:

   - `bfs_shortest_path(graph, start, goal)`: Implementa la búsqueda del camino más corto utilizando el algoritmo de búsqueda en amplitud (BFS) en el grafo dado.
   - `dfs_paths(graph, start, path=None, visited=None)`: Implementa la búsqueda de todos los caminos posibles utilizando el algoritmo de búsqueda en profundidad (DFS) en el grafo dado.

5. **Función para visualizar el grafo**:

    - `visualize_graph(graph)`: Utiliza `networkx` y `matplotlib.pyplot` para visualizar el grafo. Los nodos se representan como círculos azules, y las aristas se etiquetan con los títulos de las películas.

6. **Ejecución Principal**: En esta parte, el código carga los datos, construye el grafo, lo visualiza y luego muestra ejemplos de uso de las funciones de búsqueda de caminos.

En resumen, este código proporciona herramientas para cargar datos de películas, construir un grafo basado en las relaciones entre actores/directores y películas, visualizar el grafo resultante y realizar análisis de caminos entre actores en el grafo.