Proyecto 2: Ranking de Resultados en Buscador de Documentos

Objetivo

Extender el buscador construido previamente incorporando un sistema de reordenamiento de resultados basado en el algoritmo **PageRank** (o una variante simplificada que haga uso del grafo de co-ocurrencias). Para ello, los estudiantes deberán construir un **grafo de co-relevancia entre documentos** y aplicar PageRank sobre este grafo para obtener una medida de importancia estructural por documento.

Descripción del problema

El buscador actual utiliza un índice invertido para recuperar documentos relevantes a una consulta (ver Figura 1). Sin embargo, no distingue entre documentos que aparecen frecuentemente como parte de los resultados de muchas consultas y aquellos que son periféricos. En motores de búsqueda reales, este tipo de información estructural (como enlaces o referencias cruzadas) es clave para mejorar la calidad del ranking.Aunque la colección utilizada (GOV2) no contiene hipervínculos, podemos **simular una relación estructural** entre documentos creando un **grafo de co-relevancia**: si dos documentos aparecen juntos como respuesta a una misma consulta, se considera que están relacionados. Con ese grafo, se puede aplicar PageRank para obtener una puntuación global de importancia de cada documento.

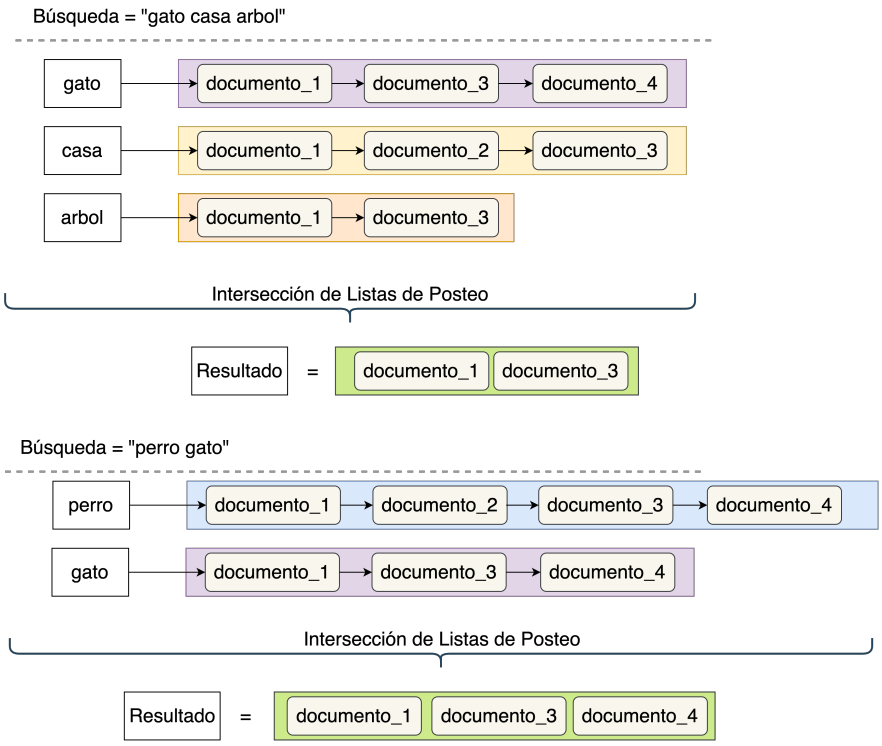
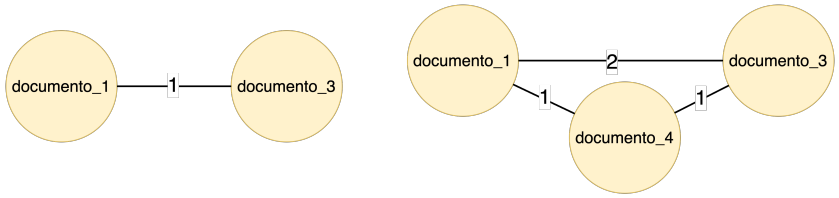


Figura 1: Ejemplo de dos búsquedas con índice invertido.

Esta puntuación se utilizará para **reordenar los resultados de nuevas consultas**, mejorando la calidad y consistencia del ranking.

Para generar un grafo a partir de los documentos se puede seguir los siguientes pasos:

1. **Preparación de los datos**
   a. Construcción del índice invertido a partir de (un subconjunto) de la colección GOV2. Esto ya fue realizado en la primera entrega del proyecto.
2. **Ejecución de consultas,** este procedimiento se realiza de manera "offline"
   a. Utilizar el "log" de consultas, y ejecutarlas "una a una".
   b. Para cada consulta q_i, se obtienen los top-K (con $K = 10$) documentos relevantes.
   c. Guardar el conjunto $R_i = \{d_{i1}, d_{i2}, \dots, d_{in}\}$ de documentos relevantes para cada consulta q_i
3. **Construcción del Grafo**
   a. Inicializar el grafo (dirigido o no) $G = (V, E)$, donde:
      i. Cada documento es un nodo en V
      ii. Las aristas corresponden a una co-relevancia.
   b. Para cada conjunto de documentos relevantes R_i:
      i. Por cada $(d_a, d_b) \in R_i$
         1. Si aún no existe una arista entre d_a y d_b, entonces créela.
         2. Si el grafo es dirigido, puede decidir arbitrariamente la dirección (o hacer dos aristas).
         3. Por cada vez que se repita el par (d_a, d_b) un conjunto de documentos relevantes puede incrementar el peso de dicha arista.
      ii. **Ejemplo (ver Figura 2)**:
         1. Consulta q_1 tiene resultados $\{d_1, d_2, d_3\}$, entonces se agregan las aristas: $E = \{(d_1, d_2);\ (d_1, d_3);\ (d_2, d_3)\}$
         2. Suponga que luego otra consulta, sea q_2 devuelve $\{d_2, d_3, d_4\}$, entonces: (d_2, d_3) incrementa su peso, se agregan (d_2, d_4) y (d_3, d_4) al conjunto E.
4. **Representación del Grafo**
   a. La que cada grupo decida (lista de adyacencia, matriz de adyacencia, etc) para un grafo con pesos.
5. **Aplicar PageRank**
   a. Investigue sobre el algoritmo PageRank.
   b. Aplicar el algoritmo PageRank (o una versión simplificada realizada por Ud) al grafo de co-relevancia.
   c. Los documentos con mayor centralidad (con más conexiones) obtendrán mayor ranking.



(a) Búsqueda "gato casa árbol" (b) Búsqueda "gato perro"

Figura 2: Ejemplo (incremental) de grafo resultante a partir de dos consultas.

Importante:

Queda estrictamente prohibido hacer uso de LLMs (como chatGPT, DeepSeek o cualquier otro) para la generación de código de su proyecto. Su uso será sancionado con nota mínima y acciones reglamentarias.

Requisitos funcionales mínimos

1. Construcción del grafo
2. Implementación de PageRank
3. Uso del PageRank en el buscador
4. Reporte de Métricas:
   a. Número total de consultas usadas para construir el grafo.
   b. Número de nodos y aristas en el grafo.
   c. Número de iteraciones requeridas para converger.
   d. Tiempos estimados de construcción del grafo y cálculo de PageRank.
   e. Comparación de ranking con y sin PageRank para al menos 5 consultas distintas.

Entrega y Presentación Final

- Martes 8 o Miércoles 9 de Julio, según corresponda a su sección.

Rúbrica de Evaluación de la Presentación

1. Comprensión del problema (10 pts)
   a. Contextualiza correctamente los proyectos anteriores.
   b. Justifica la necesidad del algoritmo de ranking, mencionar otros algoritmos similares de la literatura.
2. Solución implementada en código (50 pts)
   a. Descripción clara del código de la generación del grafo.
   b. Descripción clara del código con que implementó PageRank, o similar.
   c. Explicación y justificación de la estructura de datos seleccionada.
3. Resultados y demostración en vivo (30 pts)
   a. Demostración del funcionamiento del programa.
   b. Presentación clara de las métricas solicitadas.
   c. Reflexión sobre resultados y eficiencia.
4. Comunicación y organización (10 pts)

a. Claridad y estructura de la exposición.
b. Uso adecuado del tiempo.
c. Participación equilibrada entre integrantes (si corresponde).

Estructura para la Presentación

1. **Introducción (1 min)**
   a. Explicación del objetivo. Breve repaso del Proyecto 1.
2. **Diseño del grafo (3 min)**
   a. Cómo se generaron las consultas, cómo se conectan los documentos.
3. **PageRank (3 min)**
   a. Cómo se aplicó el algoritmo. Justificación si se usó una librería.
4. **Integración y resultados (2 min)**
   a. Comparación de resultados con y sin PageRank (ejemplifique con un par de consultas).
5. **Conclusiones (1 min)**
   a. Discusión.
   b. Lecciones aprendidas. Potenciales mejoras futuras.

Instrucciones de Entrega:

Deberá subir el código junto a todas las indicaciones que permitan su compilación y uso en Aula Virtual.

El profesor, discrecionalmente, podría citarles para realizar una revisión en persona del proyecto.