Práctica IBD

- Matías Mussini
- Antonio Reviejo
- Pablo Alonso
- Marcos Cedenilla

Arquitectura para el análisis y almacenado de artículos científicos a partir de DOIs.

Estos serán proporcionados en un archivo txt, el cual se recorrerá iterativamente, mediante la herramienta semantic scholar se extraerá un json con los datos de cada artículo.

Arquitectura

1. **Almacenamiento**: Debido a la flexibilidad que aporta y el tipo de datos que almacena, MongoDB es la base de datos idónea para el almacenado de los json, los cuales serán documentos guardados directamente (embedidos, debido a que se guardará 1 documento por artículo y dentro de un campo llamado authors irá una lista de documentos embedidos con la información de cada uno), Mongo nos permite a su vez satisfacer con su sistema de queries las consultas que necesitamos sin problema, además para queries más complejas y guardar la información relacional entre artículos y autores se opta por elegir Neo4j, en ella se guardarán también los datos, aunque no se harán campos como el abstract o características no importantes para medir relaciones.
2. **Procesamiento**: se opta por spark, debido a sus ventajas respecto a hadoop en las que destaca el uso de memoria RAM, lo cual nos permite ir cargando los datos de Mongo en memoria e ir trabajando con ellos sin la necesidad de crear archivos intermedios, pese a eso el clúster spark esta montado sobre una simulación de un HDFS, al cual también estará conectado el nodo con el que se accede programáticamente a la infraestructura.
3. **Acceso**: el acceso se hará mediante python, su driver para mongo, neo4j y pyspark, a través de un nodo con soporte para jupyterlab, todos los procesos de carga y descarga de datos y procesado de los DOIs se hará en este lenguaje y el desarrollo queda plasmado en el Jupyter Notebook adjuntado.

En todo momento intentamos simular con Docker lo que sería nuestra arquitectura a gran escala (distribuida y con gran cantidad de datos) y actuamos como tal.

De manera que desplegamos 2 nodos para Mongo, 1 para Neo4j y un clúster spark que esta explicado con más detalle en el archivo pdf cluster_spark, contamos además con un sistema de carga, descarga y procesamiento de datos en batch para no saturar la memoria principal.

Toda la arquitectura se podrá desplegar mediante el archivo docker-compose, las imágenes de nodos pertenecientes al clúster, aunque esten específicados los dockerfile de creación de sus imágenes en la carpeta Dockerfiles, están subidos a dockerhub por lo que no es necesario la construcción de ninguna imagen en local.

Funcionalidades

Creación de datos estáticos

Mediante queries mongo únicamente, nuestra aplicación es capaz de crear los siguientes archivos csv con datos relevantes sobre los artículos.

1. Información sobre **Documentos**: csv con información de id del documento, fecha de publicación y título.
2. Información sobre **Autores**: csv con información de nombre de cada autor y la cantidad de artículos publicados.

Creación de datos dinámicos

Mediante la potencia de computo distribuido y escalabilidad que nos brinda spark podemos crear una serie de datos dinámicos, estos datos se corresponden a la cantidad de veces que aparecen una serie de palabras dadas en el corpus, guardando cada palabra y su número de apariciones en un archivo csv, el texto se procesa para que no afecten caracteres especiales o signos de puntuación.

Soporte para consultas

Mediante nuestro sistema somos capaces de responder con una eficiencia suficiente a consultas simples tales como:

- Obras en las que un **Autor** dado ha participado, ordenado por su relevancia en estas.
- Obras en las que una **Palabra Clave** aparece, ordenadas por la frecuencia de esta, útil para buscadores, por ejemplo...

También somos capaces de responder a consultas más elaboradas tales como:

- Listado ordenado de Autores con los que un **Autor** dado ha colaborado.
- Cantidad de palabras de un **Tamaño** específico en todo el corpus.

Mejoras posibles

Las mejoras más relevantes que se podrían llevar a cabo para futuras versiones son dos:

1. Adjuntar un sistema de análisis de pdfs para no depender de una lista de DOIs dada.
2. Modificación del código para que tanto las descargas de datos de mongo como su posterior volcado o escritura en csv, se hagan de manera asíncrona reduciendo el impacto de trabajar en batches.