PROYECTO INTEGRADOR IA + SE

Universidad Politécnica Salesiana
Jorge Arévalo, Gustavo Guallpa
jarevalop1@est.ups.edu.ec
fguallpa@est.ups.edu.ec

RESUMEN:

Los métodos de búsqueda son muy importantes en nuestro mundo actúa ya que ellos nos permiten en base a ciertos criterios poder recomendar soluciones acordes a las necesidades del usuario mediante el desarrollo del presente documento presentaremos algunos problemas y soluciones que nos hemos planteado para dar soluciones optimas en situaciones de la vida real.

PALABRAS CLAVE: *Inteligencia Artificial, Neo4j, Java, Algoritmos, Grafos, Python, etc.*

ABSTRACT:

Search methods are very important in our world, since they allow us, based on certain criteria, to be able to recommend solutions according to the user's needs. Through the development of this document, we will present some problems and solutions that we have set ourselves to give optimal solutions in real life situations.

KEY WORDS: *Artificial Intelligence, Neo4j, Java, Algorithms, Graphs, Python, etc...*

1 INTRODUCCION

Para el desarrollo de nuestro sistema de recomendación partimos desde la base que es el conocimiento de cada uno de los algoritmos que previamente seleccionamos a continuación se detalla de manera general cada uno de ellos a fin de poder de una mejor manera el presente trabajo.

Al momento de nuestra elección sobre que algoritmos vamos a utilizar depende en gran medida del caso de uso que le vayamos a dar, para ello puede influir elementos como:

1. **SECCION 1 "ALGORTIMO DE CENTRALIDAD"**

CERCANÍA CENTRALIDAD

El algoritmo calcula la suma de sus distancias a todos los demás nodos, la suma resultante se invierte y determina la puntuación de proximidad para ese nodo. Es importante destacar que, a pesar de ser unos algoritmos altamente funcionales, estos funcionan de forma óptima en grafos conectados, es decir, que su aplicación en grafos no conectados puede arrojarnos algunos errores de análisis. Pudiésemos obtener una distancia infinita entre dos nodos en componentes conectados por separado. Al ocurrir esto se estaría otorgando una puntuación de centralidad de cercanía infinita cuando hagamos la sumatoria de las distancias de un vértice determinado**.**

2. **SECCION 2 "ALGORITMO DE DETENCION DE LA COMUNIDAD"**

LOUVAIN

Los algoritmos de detección de comunidades cumplen con una función de gran importancia para el análisis de datos en grafos que contienen información compleja, Estos tomando en cuenta sus características, formas de relación y diferentes tipos de conexiones nos ayudan a establecer o determinar la existencia de comunidades complejas de datos.

Dentro de este conjunto importante de algoritmos podemos contar el algoritmo de Louvain, el algoritmo de propagación de etiquetas y dos tipos interesantes de algoritmos orientados a estudiar diferentes tipos de grafos. Para entender el funcionamiento de grafos dirigidos contamos con el algoritmo de componentes fuertemente conectados y para grafos no dirigidos el algoritmo de componentes débilmente conectados.

3. **SECCION 3 "ALGORITMOS DE SIMILITUD"**

VECINOS APROXIMADOS MAS CERCANOS

Los algoritmos de similitud permiten comparar un conjunto de nodos en función de los nodos que estos están conectados, sabemos que dos nodos se pueden considerar similares siempre y cuando estos compartan muchos de sus vecinos más cercanos. Dentro de su amplia gama cada uno de los algoritmos tiene un enfoque distinto en los que se toma en cuenta diferentes características.

Alguna vez se ha preguntado, como es que empresas como Amazon, Best Buy o alguna tienda de compras en línea, cuando una persona compra algo la página empieza a recomendar productos muy similares o productos que se estaba necesitando, uno se pone a pensar que me estarán espiando, pero la realidad es que la empresas utilizan algoritmos, que permiten realizar estas búsquedas y en base a las compras previamente realizadas o compras realizas por

usuarios muy similares, permite brindar una recomendación, que en muchas de la ocasiones suelen ser muy acertadas. En la amplia variedad de algoritmos que se emplea dependiendo de la necesidad de la empresa, en este punto les vamos a hablar de el algoritmo de vecinos aproximados más cercanos que permite que usuarios pueden encontrar lo que necesitan en un tiempo más razonable, se basa en encontrar el vecino más cercano en la mayoría de los casos, todo ello depende de la cantidad de datos que se consulta, este tipo de algoritmo es muy utilizado en el reconocimiento de imágenes, el aprendizaje automático, lingüística computacional, etc.

4. **SECCION 4 "ALGORITMOS DE BÚSQUEDA DE RUTAS"**

BÚSQUEDA POR AMPLITUD

En nuestra actualidad donde el tiempo de respuesta es algo muy importante al momento de tomar decisiones, este tipo de algoritmos resultan ser muy interesantes porque para llegar desde un nodo a otro permiten detectar una ruta eficiente.

5. **SECCION 5 "ALGORITMOS DE PREDICCIÓN DE ENLACES"**

VECINOS COMUNES

En el apartado de Algoritmos de Predicción de Enlaces, se nos presenta algo interesante con este tipo de algoritmo, tiene que ver con la capacidad de poder predecir nuevas relaciones entre los nodos, se puede interpretar como la capacidad que tiene para poder analizar e interpretar la estructura de este y la cercanía con un par de nodos y en base a ello dar como resultado un valor de la probabilidad de que estos nodos se conecten.

2 DESARROLLO

2.1 DESCRIPCION DETALLADA DEL PROBLEMA Y SU PROPUESTA DE SOLUCION.

Cuando una persona llega a la ciudad de Cuenca ¿, es difícil poder obtener información de primera mano sobre cada uno de los atractivos que hacen de Cuenca, Cuenca, es por ello que mediante este trabajo integrador se implementa un sistema que haga como un guía del nuestro turista, para ello se ha definido varios problemas y sus soluciones que se detallaran a continuación, para ello basándonos en información de:

- Parques
- Iglesias
- Hospitales
- Bomberos
- Policías
- Escuelas
- Centros Médicos
- Lugares o atractivos turísticos
- Centros de Estimulación temprana
- Centros Educativos para el desarrollo de niños.

Toda la información de ha obtenido de la página de Google Maps.

Problema 1

Descripción del problema

Un profesor de un país Extranjero llega al Ecuador y visita Cuenca, donde necesita que le recomienden el Centro de Estimulación Temprana (Técnicas educativas especiales). El profesor desea conocer el nombre del Centro de Estimulación.

Problema 2

Descripción del problema

Un padre de familia desea conocer un centro educativo para el desarrollo de niños, la cual necesita una recomendación de un centro educativo para inscribir a su hijo en el centro educativo.

Problema 3

Descripción del problema

Un turista desea hacer sus exámenes de sangre la cual necesita que le recomiende un hospital o Centro médico donde estos lugares tengan más hospitales o centros médicos vecinos.

Problema 4

Descripción del problema

Un turista cristiano católico se encuentra de visita en la ciudad de Cuenca, resulta que para su punto de partida siempre va a tomar de referencia la Catedral de la Inmaculada Concepción en lo referente a iglesias, en lo que corresponde a parques su punto de referencia va a ser el parque Calderón, el turista desea conocer que camino debe seguir para llegar a cada uno de sus destinos.

Problema 5

Descripción del problema

En este apartado tenemos a un familiar determinado que se encuentra de visita en la ciudad de Cuenca el quiere en base a lugares turísticos o escuelas de su preferencia ver qué tanto de similitud tiene con sus amigos y en base a ello poder tomar decisiones para poder salir de viaje o ver a que escuela se puede matricular.

Problema 6

Descripción del problema

Una persona en caso de emergencia necesita conocer un centro de policía y una estación de bomberos a la cual necesita llamar

Propuesta de solución al problema 1

Para dar solución al problema se tiene un sistema que tiene todos los centros de estimulación temprana, este sistema usa el algoritmo de centralidad por cercanía la cual devuelve el nombre del Centro de Estimulación con mayor peso de centralidad de todos los otros centros.

Propuesta de solución al problema 2

Para esta solución se utiliza el mismo método de las cuales se carga una base de datos de centros educativos para el desarrollo de niños.

Propuesta de solución al problema 3

Para esta solución el sistema utiliza el algoritmo de Louvain donde tiene la base de datos de los hospitales y centros médicos, la cual el usuario debe ingresar el lugar al que el sistema le recomienda. Este algoritmo muestra los hospitales o centros médicos que tengan mayor conexión con hospitales cercanos.

Propuesta de solución al problema 4

Para dar solución a este problema se tendrá un sistema en que nuestro turista pueda seleccionar que tipo de acción desea hacer ya sea en parques o en iglesias, el programa le mostrará todas las posibles rutas, para al final recomendarle una ruta para llegar a su destino.

Propuesta de solución al problema 5

Para dar solución a este problema se tendrá un sistema en que el familiar podrá escoger entre las opciones de parque o escuelas, debe seleccionar los lugares que sean de su preferencia

Propuesta de solución al problema 6

Para este problema se utiliza el algoritmo de vecinos cercanos donde este algoritmo muestra la similitud entre nodos, se tiene cargado la base de datos de la estación de bomberos y de policías.

2.2 DIAGRAMA ESQUEMATICO QUE EXPLIQUE COMO SE DESARROLLO Y MODELO DE LA PROPUESTA.

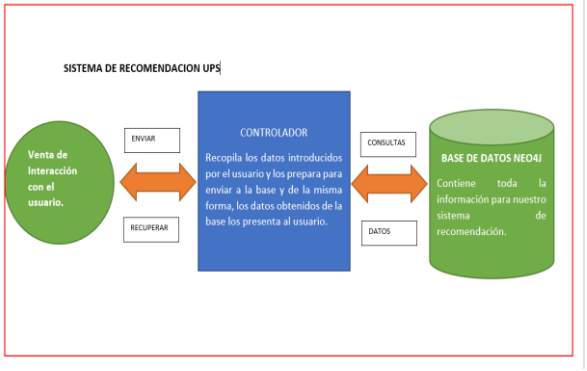

Fig. 1. Diagrama Esquelético del Sistema de Recomendación.

Nuestro programa va a ser relativamente sencillo se contará con una ventana principal que mostrará cada uno de los algoritmos con cada una de sus opciones, un controlador que gestiona la comunicación entre la interfaz gráfica y la base de datos.

2.3 ARQUITECTURA DEL SISTEMA DE SIMULACION

Como se mencionó en el problema para el procesamiento de todos nuestros de datos se utilizará la base orientada a grafos de Neo4j.

Que una herramienta muy potente para este tipo de implementaciones.

2.4 DESCRIPCION DE LA SOLUCION Y PASOS SEGUIDOS (NIVEL INVESTIGATIVO).

Para la implementación de solución para todos los problemas que se mencionó anteriormente todo esto se realizó en el lenguaje de Java, para ello se tuve que investigar la forma en la que podía conectar el entorno de Java con NEO4J.

```
public void Conectar() {
    try {
        String uri = "bolt://localhost:7687";
        String user = "neo4j";
        String password = "admin";
        driver = GraphDatabase.driver(uri, AuthTokens.basic(user, password));
    } catch (Exception e) {

    }
}
```

Fig. 2. Conexión desde Java a Neo4.

Un punto importante recalcar, a mi compañero de trabajo le sucedió que no se podía conectar al Neo4j, por estas dos razones:

-Al momento que uno va a la página oficial de Neo4j para la descarga de su software, la empresa nos proporciona una contraseña que dice que es neo4j pero

resulta que al querer hacer la conexión como localhost desde el navegador, con los credenciales proporcionados por la empresa, la conexión no se puede realizar, resulta que investigando algunos foros menciona que la contraseña es **admin**, en efecto esta contraseña nos funcionó,

-Es muy importante a la hora de crear el proyecto dentro de nuestro entorno de programación favorito, hacerlo de tipo **MAVEN,** para que de esa forma se pueda agregar las dependencias necesarias para la conexión.

```
    <dependency>
        <groupId>org.neo4j.driver</groupId>
        <artifactId>neo4j-java-driver</artifactId>
        <version>4.0.1</version>
    </dependency>
</dependencies>
```

Fig. 3 Dependencia necesaria para poder realizar la conexión con Neo4j.

Una vez que se tupo la conexión exitosa con la base de datos de grafos, reto que se tuvo fue sabe cómo construir las estructuras para poder recuperar los valores de cajas de texto, combo box y otros elementos de interfaz gráfica y una vez recuperados poderlos enviar a la base de grafos y de la misma forma poder tratar los datos que nos devuelve.

```
try (Session session = driver.session()) {
    List<Nodo> vec = new ArrayList<>();
    String greeting = session.writeTransaction((Transaction tx) -> {
        Result result = tx.run(
                "MATCH (start:Iglesia {name: '" + inicio + "'}), (end:Iglesia{name: '" + fin + "'}) "
                + "CALL gds.alpha.shortestPath.stream({ "
                + "nodeProjection: 'Iglesia', "
                + "relationshipProjection: { "
                + "LINK: {type: 'LINK',properties: 'cost'}}, "
                + "startNode: start, "
                + "endNode: end, "
                + "relationshipWeightProperty: 'cost' "
                + "}) "
                + "YIELD nodeId, cost "
                + "RETURN gds.util.asNode(nodeId).name AS name, cost ");
        for (Record record : result.list()) {
            String nomi = record.get("name").asString();
            Double costo = 0.00;
            Nodo ns = new Nodo(nomi, costo);
            vec.add(ns);
        }
        return null;
    });
    return vec;
```

Fig. 4 Recuperar datos que nos brinda la base de datos.

Como se puede evidenciar es necesario crear una sesión que nos permita realizar una transacción para que nos permita movernos y poder realizar la acciones, es importante destacar que para recupera los datos es necesario tener la clase, para de esta forma crear los objetos con los datos que nos devuelve la base.

```
        + "RETURN gds.util.asNode(nodeId).name AS name, "
for (Record record : result.list()) {
    String nomi = record.get("name").asString();
    Double costo = 0.00;
    Nodo ns = new Nodo(nomi, costo);
    vec.add(ns);
}
return null;
```

Fig. 5 Creando objetos con los datos que nos devuelve la base.

Una vez explicado todo esto, estamos listo para poder realizar las acciones que se necesiten para cada algoritmo.

Veamos ahora como se realizó con cada uno de nuestros algoritmos de recomendación.

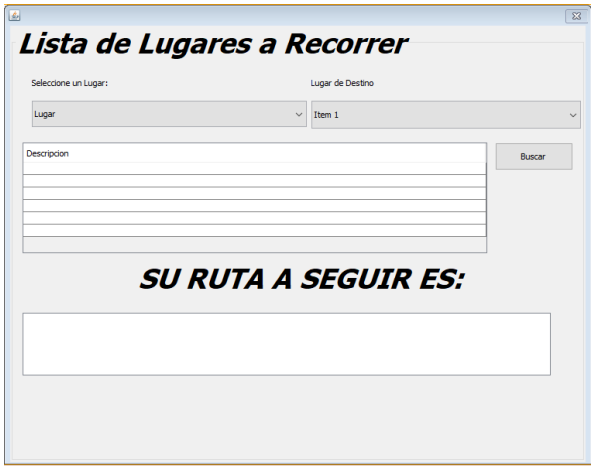

Fig. 6 Venta del Algoritmo de Amplitud.

En esta venta se le presentar al turista la iglesia a las que se puede llegar todo ellos partiendo desde Catedral de la Inmaculada Concepción, al finalizar se le mostrará la ruta a seguir para llegar al lugar indicado.

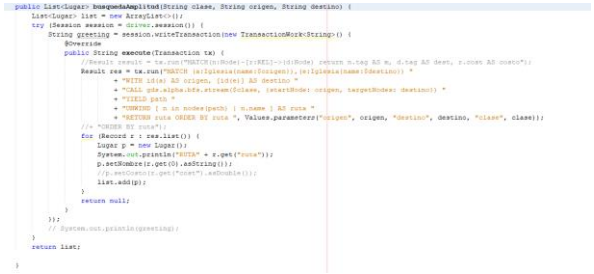

Fig. 7 Algoritmo Búsqueda por Amplitud.

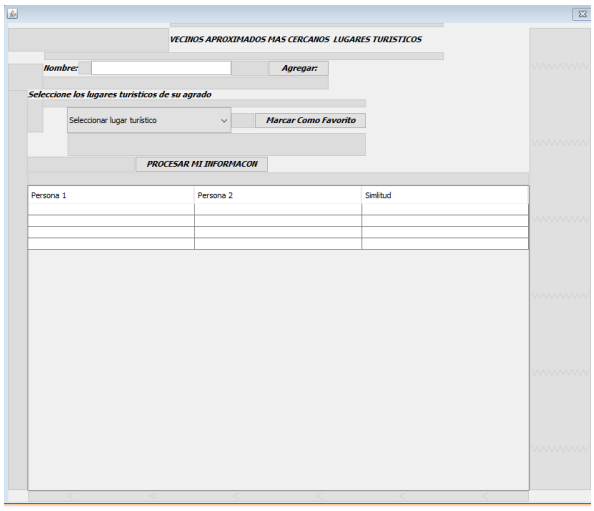

Fig. 8 Venta del Algoritmo de Vecinos Aproximados más cercanos.

Fig. 9 Algoritmo de Vecinos Aproximados más cercanos.

3. CONCLUSIONES:

El desarrollo del presente trabajo nos ha permitido poder profundizar nuestros conocimientos y ver que este tipo de algoritmos en los que quizá de alguna manera se intuye inteligencia nos ayudan a poder saber que decisiones poder con una certeza de que las cosas se hacen en base a datos que pueden apoyar nuestra decisión y no quizá por nuestra intuición , realmente al inicio de la realización del proyecto realmente esto fue un desafío , porque no se conocía realmente la forma de funcionamiento o las ventajas que nos ofrece Neo4j , y la aplicación de los algoritmos con datos reales.

4. RECOMENDACIONES:

-Como una recomendación general se pediría que los estudiantes recurran al profesor para obtener más información en el caso de que no se comprenda el tema a investiga, debido a que como el tema es nuevo para nosotros muchas veces no se entiende claramente que los que el docente nos pide y se corre el riego de realizar el proyecto de la forma que no se debe hacer.

La realización del proyecto nos ha permitido generar nuevas destrezas.

5. ANEXO

En este apartado adjuntamos el enlace de nuestra página en la que se encuentra de manera detallada los primeros pasos a seguir con NEO4J, un video corto explicativo que habla un poco de esta base orientada a grafos.

https://sites.google.com/view/proyecto-integrador-ia-se/inicio

Toda la información de los algoritmos y sus aplicaciones se encuentra en el cuaderno de Júpiter.

6. REFERENCIAS

[1] Neo4j Graph Platform – The Leader in Graph Databases. (2020). Retrieved 9 June 2020, from https://neo4j.com/

[2] Using Neo4j from Java - Neo4j Graph Database Platform. (2020). Retrieved 9 June 2020, from https://neo4j.com/developer/java/