NOMBRE:

ALEX BENAVIDEZ

CARRERA:

INGENIERIA EN SISTEMAS

MATERIA:

INTELIGENCIA ARTIFICIAL

PROFESOR:

DIEGO QUISI

FECHA:

20/05/20

1. INTRODUCCION

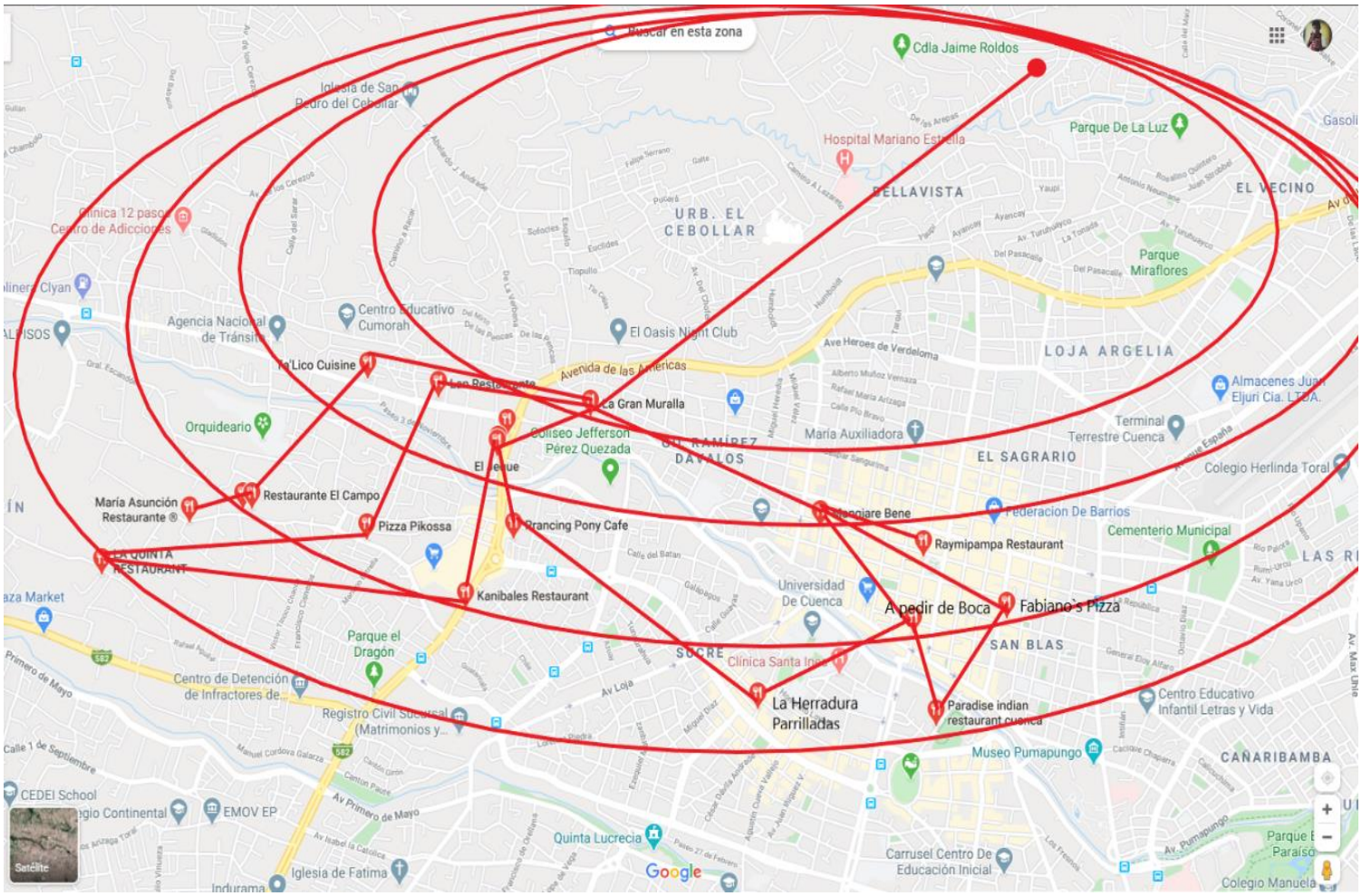
En la siguiente practica se empleará el método de búsqueda por ascenso de colinas, en donde utilizaremos los restaurantes ubicados en la ciudad de Cuenca y sus distancias hacia un restaurante meta el cual.

1. DESARROLLO DE CONTENIDOS

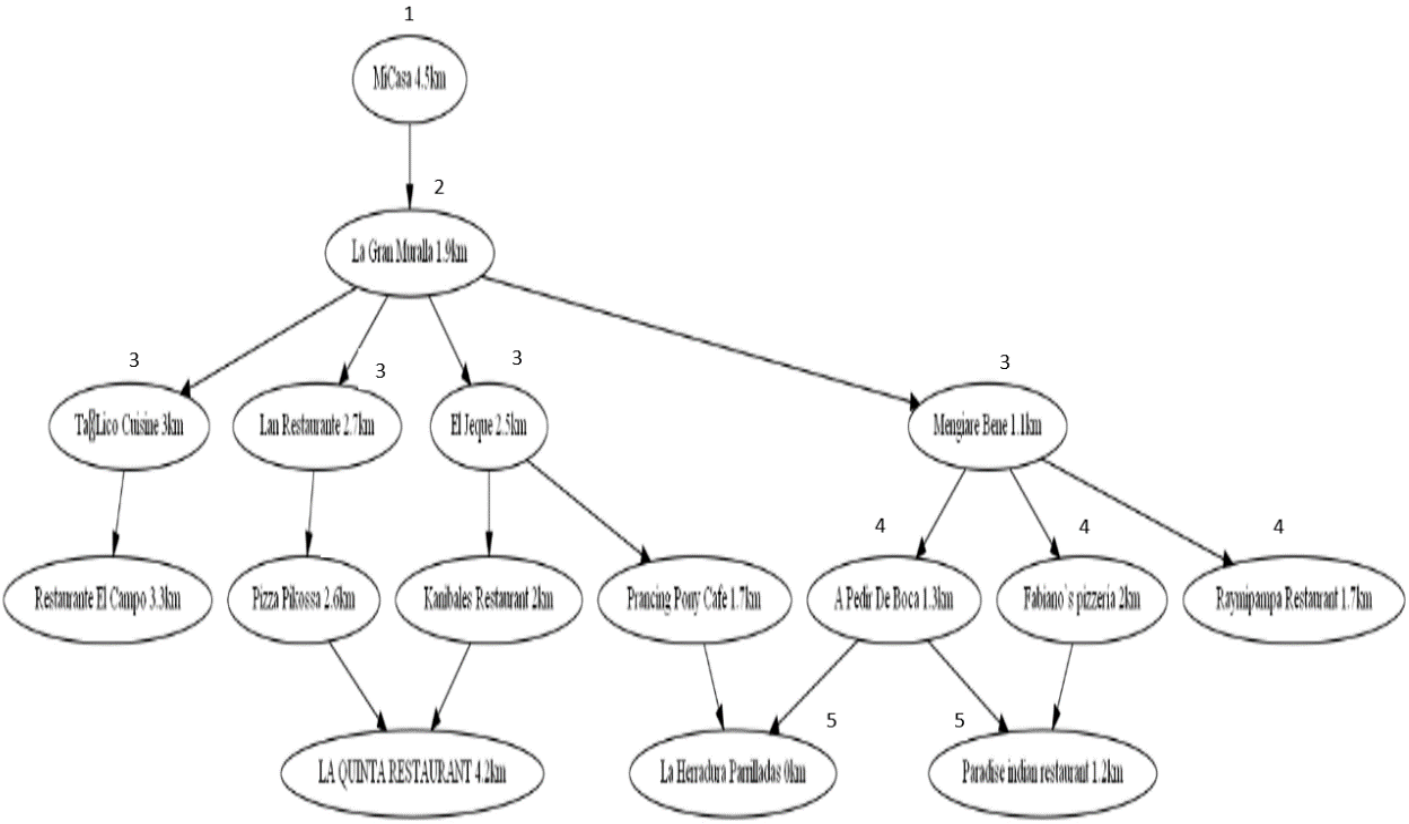
Es una de las técnicas que comúnmente se emplean cuando el algoritmo A* tiene inconvenientes o falla, este método se encarga de hacer realizar una búsqueda local y también se mueve por los estados vecinos. Su fundamento base es comenzar con una estimación inicial de la solución e ir mejorándola gradualmente hasta que sea una solución.

- **Ventajas:**
  - Usa poca memoria
  - Es capaz de encontrar con frecuencia soluciones razonables en espacios de estados muy grandes o infinitos.

1.1. Realización del grafo con la herramienta google maps.



1.2. Graficación del árbol mediante el lenguaje de programación Python y desarrollo del método de manera manual.



Visitados= {MiCasa, LaGranMuralla}

Visitados= {MiCasa, LaGranMuralla, MengiareBene}

Visitados= {MiCasa, LaGranMuralla, MengiareBene}

Visitados= {MiCasa, LaGranMuralla, MengiareBene, APedirDeBoca}

Visitados= {MiCasa, LaGranMuralla, MengiareBene, APedirDeBoca, La Herradura Parrilladas}

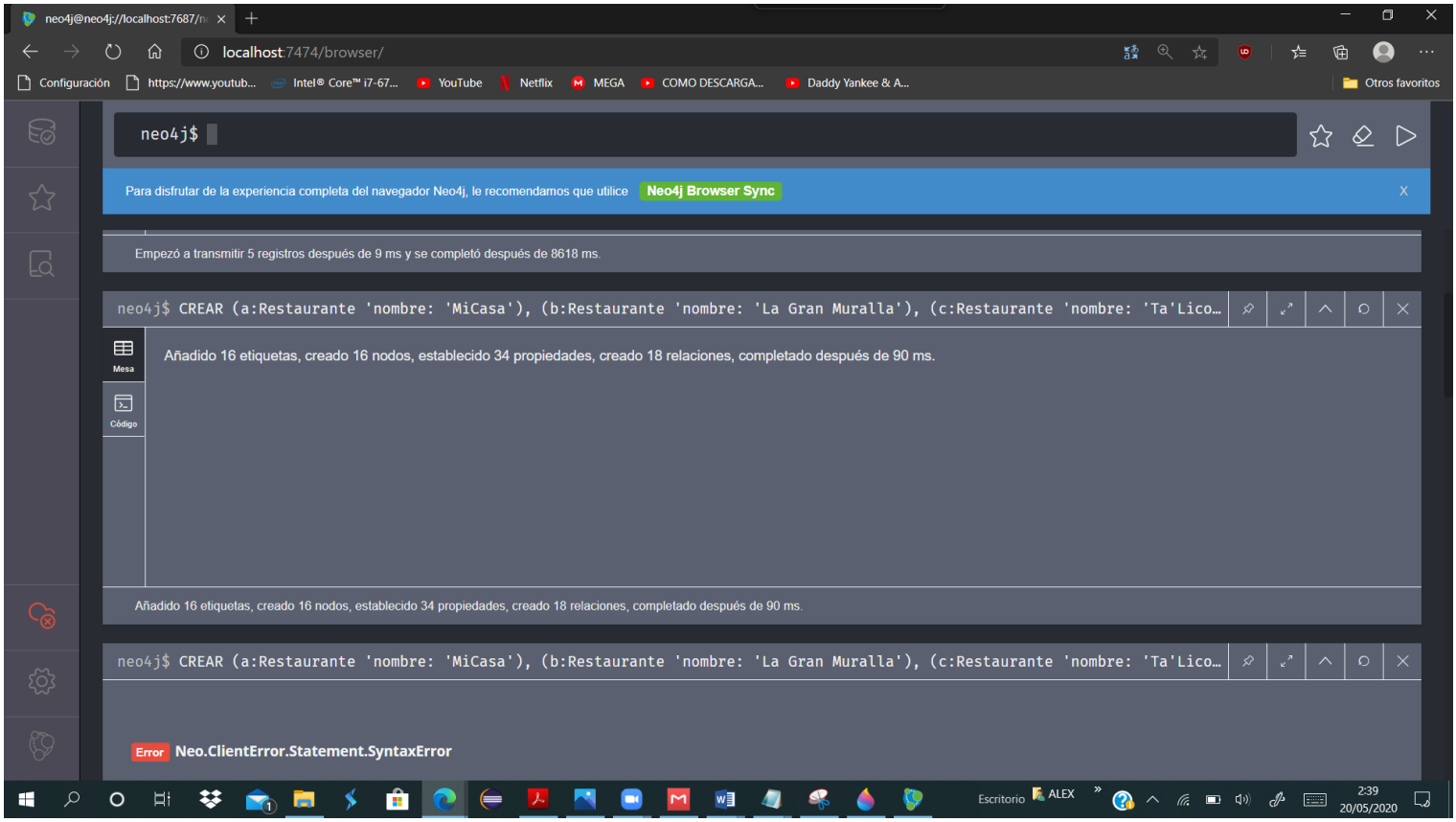
Visitados = {MiCasa, LaGranMuralla, MengiareBene, APedirDeBoca, La Herradura Parrilladas}

Ruta = {MiCasa, LaGranMuralla, MengiareBene, APedirDeBoca, La Herradura Parrilladas}

1.3. Verificación del método utilizando la base de datos orientada a grafos Neo4J.

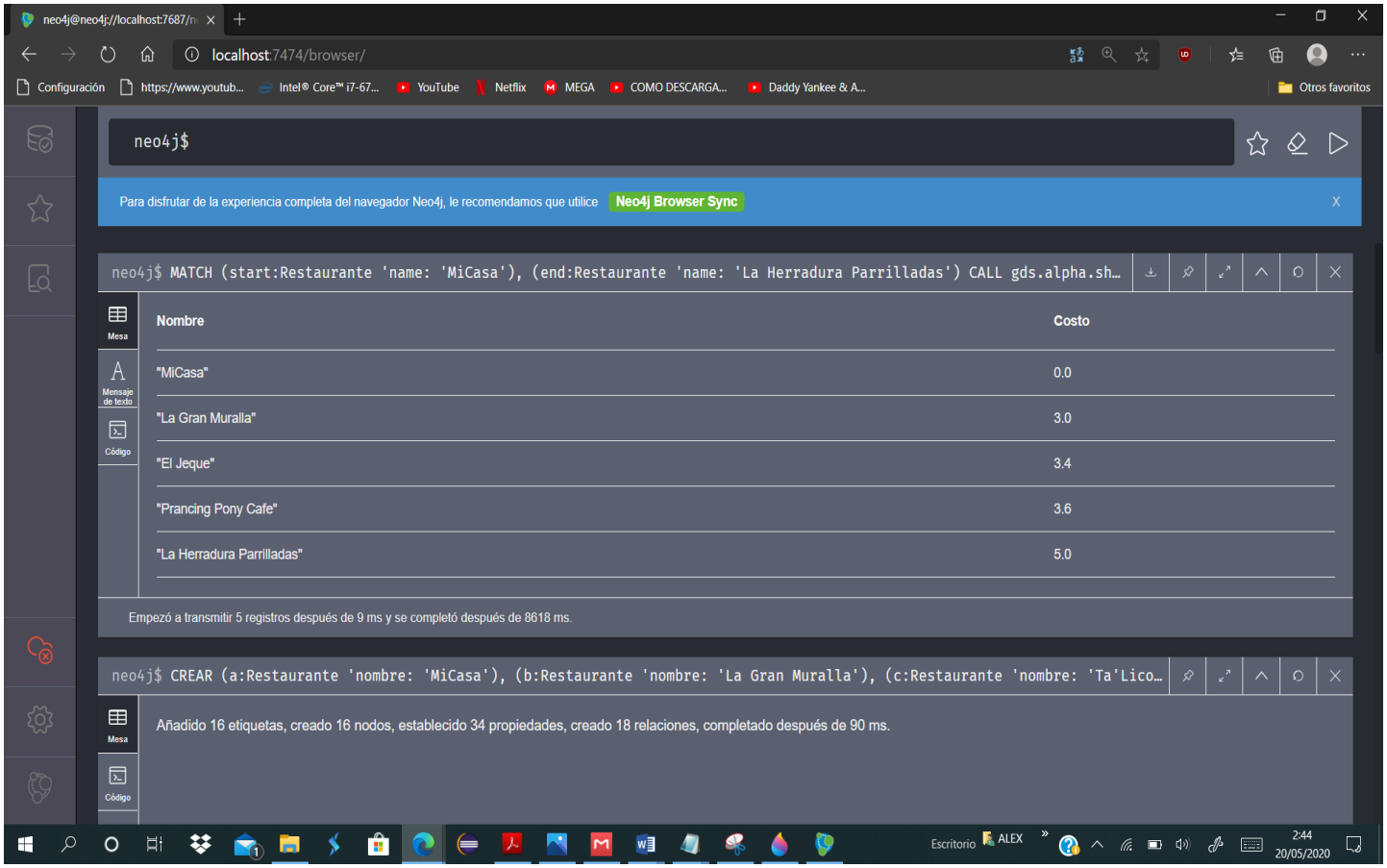
Debido a que el método por Ascenso de Colinas no se encuentra implementado en la base de datos Orientada grafos Neo4j, implementaremos el algoritmo shortest-path, el cual se asemeja bastante al por asenso a colinas. Para lo cual utilizaremos las siguientes sentencias para crear los nodos y sus relaciones.

```
CREATE (a:Restaurante {name: 'MiCasa'}),
       (b:Restaurante {name: 'La Gran Muralla'}),
       (c:Restaurante {name: 'Ta´Lico Cuisine'}),
       (d:Restaurante {name: 'Lan Restaurante'}),
       (e:Restaurante {name: 'El Jeque'}),
       (f:Restaurante {name: 'Mengiare Bene'}),
       (g:Restaurante {name: 'Restaurante El Campo'}),
       (h:Restaurante {name: 'Pizza Pikossa'}),
       (i:Restaurante {name: 'Kanibales Restaurant'}),
       (j:Restaurante {name: 'Prancing Pony Cafe'}),
       (k:Restaurante {name: 'A Pedir De Boca'}),
       (l:Restaurante {name: 'Fabiano`s pizzería'}),
       (m:Restaurante {name: 'Raymipampa Restaurant'}),
       (n:Restaurante {name: 'LA QUINTA RESTAURANT'}),
       (o:Restaurante {name: 'La Herradura Parrilladas'}),
       (p:Restaurante {name: 'Paradise indian restaurant'}),
       (a)-[:ROAD {cost: 3}]->(b),
       (b)-[:ROAD {cost: 0.8}]->(c),
       (b)-[:ROAD {cost: 0.5}]->(d),
       (b)-[:ROAD {cost: 0.4}]->(e),
       (b)-[:ROAD {cost: 1.1}]->(f),
       (c)-[:ROAD {cost: 0.2}]->(g),
       (d)-[:ROAD {cost: 0.6}]->(h),
       (e)-[:ROAD {cost: 0.6}]->(i),
       (e)-[:ROAD {cost: 0.2}]->(j),
       (f)-[:ROAD {cost: 0.6}]->(k),
       (f)-[:ROAD {cost: 0.8}]->(l),
       (f)-[:ROAD {cost: 0.3}]->(m),
       (h)-[:ROAD {cost: 1}]->(n),
       (i)-[:ROAD {cost: 1.7}]->(n),
       (j)-[:ROAD {cost: 1.4}]->(o),
       (k)-[:ROAD {cost: 0.6}]->(o),
       (k)-[:ROAD {cost: 0.3}]->(p),
       (l)-[:ROAD {cost: 0.3}]->(p)
```



1.4. Para finalizar utilizamos la siguiente sentencia para poder realizar el método shortest-path en los restaurantes, en donde el nodo inicio será MiCasa y el nodo destino será 'La Herradura Parrilladas'.

```
MATCH (start:Restaurante {name: 'MiCasa'}), (end:Restaurante {name: 'La Herradura Parrilladas'})
CALL gds.alpha.shortestPath.stream({
  nodeProjection: 'Restaurante',
  relationshipProjection: {
    ROAD: {
      type: 'ROAD',
      properties: 'cost',
      orientation: 'UNDIRECTED'
    }
  },
  startNode: start,
  endNode: end,
  relationshipWeightProperty: 'cost'
})
YIELD nodeId, cost
RETURN gds.util.asNode(nodeId).name AS name, cost
```



- **CONCLUSIONES**
  - Se ha realizado la práctica de manera satisfactoria de manera correcta utilizando el método de ascenso por colinas de manera manual y mediante la utilización de la base de datos Neo4j.
  - Este método es muy eficiente para lograr obtener una solución de manera rápida.

- **BIBLIOGRAFÍA**

(s.f.). Obtenido de https://htmlpreview.github.io/?https://raw.githubusercontent.com/vlarobbyk/busquedas-ia/master/bsqueda_por_ascenso_de_colinas.html

Navarro, R. B. (2006). *Meta-Heurísticas híbridas para optimización mono-objetivo y multi-objetivo* . Almeria.