

ANÁLISIS DE ALGORITMOS

INTRODUCCIÓN

En este análisis, abordamos la evaluación de la eficiencia de cinco métodos de búsqueda, a saber, BFS, DFS, UCS, A*, e IDA*, aplicados a dos nodos seleccionados al azar desde nuestra ubicación inicial. Este análisis exhaustivo busca determinar cuál de estos métodos se destaca como el más eficiente en el contexto específico de nuestra tarea. A través de una meticulosa comparación, exploraremos y evaluaremos la idoneidad de cada algoritmo, buscando revelar sus fortalezas y limitaciones en la exploración de nodos en nuestro espacio.

UBICACIÓN

TIMES SQUARE, MANHATTAN
COMMUNITY BOARD 5,
MANHATTAN, CONDADO DE
NUEVA YORK, NUEVA YORK,
10036, ESTADOS UNIDOS DE
AMÉRICA



Moisés Hiram Pineda Campos - A01625510
Emilio Berber Maldonado - A01640603
Samuel García Berenfeld - A01642317

METODOLOGÍA

BFS

Este algoritmo comienza explorando los nodos vecinos al nodo inicial antes de moverse a los nodos más lejanos. Utiliza una estrategia de expansión en anchura, asegurando que todos los nodos a una distancia dada se exploren antes de pasar a la siguiente capa de nodos. FIFO

DFS

A diferencia de BFS, DFS adopta una estrategia de expansión en profundidad. Explora tan lejos como sea posible a lo largo de cada rama antes de retroceder. Puede ser implementado de manera recursiva, utilizando una pila para rastrear los nodos.

LIFO

UCS

Este algoritmo considera el costo acumulado desde el nodo inicial hasta cada nodo alcanzado. Prioriza la expansión de nodos de menor costo, garantizando una búsqueda que optimiza el costo total.

A *

A* combina la eficiencia de UCS con la heurística. Evalúa los nodos basándose en el costo acumulado y una estimación del costo restante hasta el objetivo. Esto permite una búsqueda más informada y puede ser considerablemente más eficiente en términos de tiempo.

IDA *

IDA* es una variante de A* que realiza una búsqueda en profundidad iterativa con límites crecientes. Combina la capacidad de exploración en profundidad con la heurística de A*, ajustando los límites para lograr una búsqueda eficiente y completa.

BFS

Times Square, Manhattan, NYC

P 1

orig_node = 103098680

dest_node = 42498881

Distancia recorrida por BFS:

28144.965000000007 metros



P 2

orig_node = 103098680

dest_node = 2298803501

Distancia recorrida por BFS:

25908.645999999993 metros



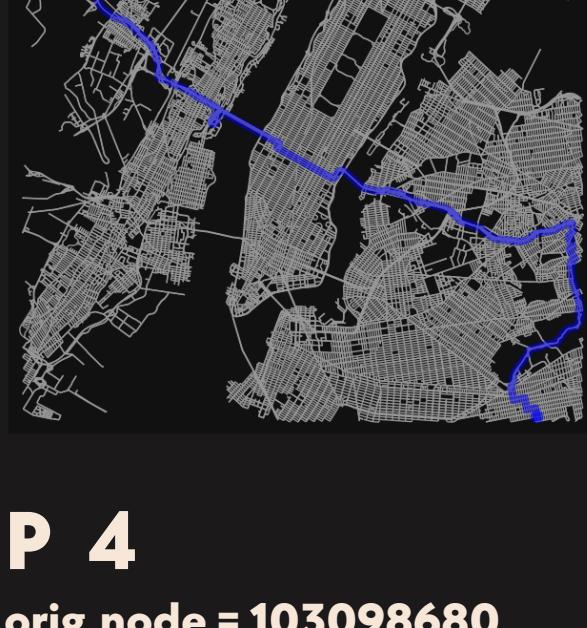
P 3

orig_node = 103098680

dest_node = 42499712

Distancia recorrida por BFS:

34454.83299999999 metros



P 4

orig_node = 103098680

dest_node = 42826843

Distancia recorrida por BFS:

19092.492000000006 metros



P 5

orig_node = 103098680

dest_node = 103238167

Distancia recorrida por BFS:

5208.226000000001 metros



DFS

Times Square, Manhattan, NYC

P 1

orig_node = 103098680
dest_node = 42498881

Distancia recorrida por DFS:
697967.4039999989 metros



P 2

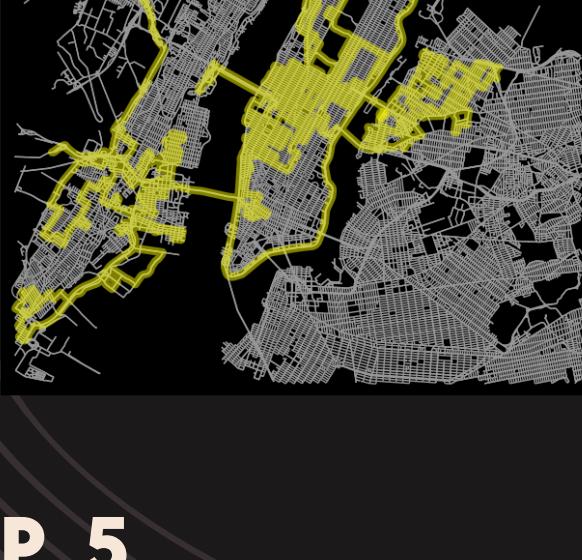
orig_node = 103098680
dest_node = 2298803501

Distancia recorrida por DFS:
427268.01799999975 metros

P 3

orig_node = 103098680
dest_node = 42499712

Distancia recorrida por DFS:
668493.7039999989 metros



P 4

orig_node = 103098680
dest_node = 42826843

Distancia recorrida por DFS:
287490.74399999983 metros

P 5

orig_node = 103098680
dest_node = 103238167

Distancia recorrida por DFS:
219515.79800000016 metros



UCS

Times Square, Manhattan, NYC

P 1

orig_node = 103098680

dest_node = 42498881

Distancia recorrida por UCS:

23979.372999999996 metros



P 2

orig_node = 103098680

dest_node = 2298803501

Distancia recorrida por UCS:

24762.78700000002 metros



P 3

orig_node = 103098680

dest_node = 42499712

Distancia recorrida por UCS:

28879.74999999999 metros



P 4

orig_node = 103098680

dest_node = 42826843

Distancia recorrida por UCS:

18772.149999999998 metros



P 5

orig_node = 103098680

dest_node = 103238167

A*

Times Square, Manhattan, NYC

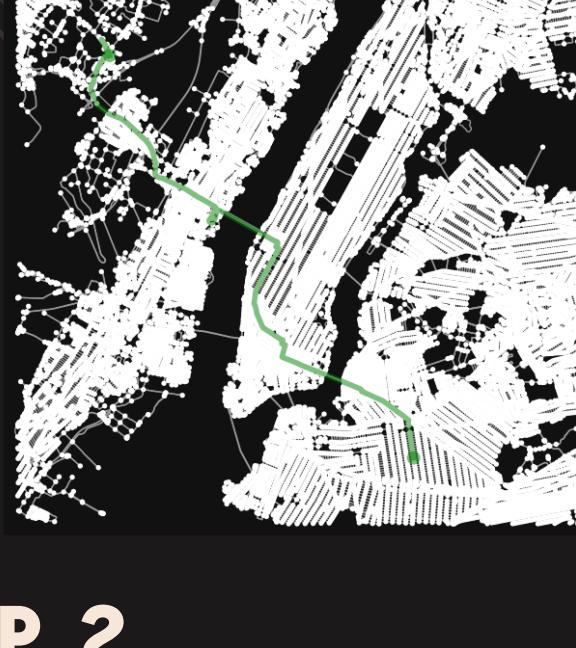
P 1

orig_node = 103098680

dest_node = 42498881

Distancia recorrida por A*:

23979.372999999996 metros



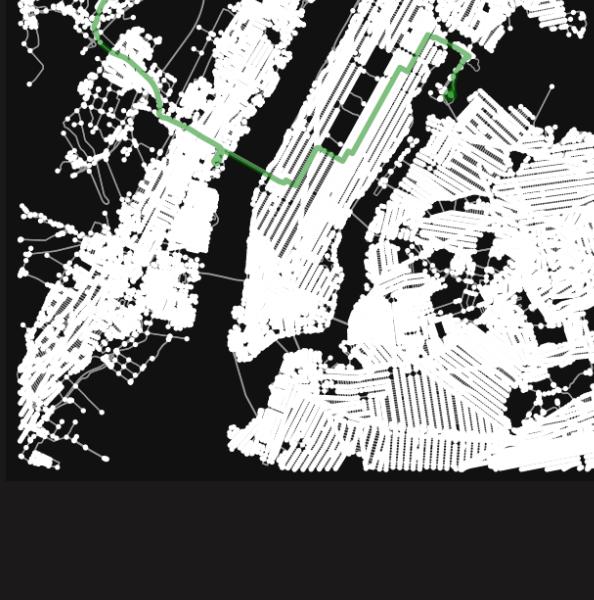
P 2

orig_node = 103098680

dest_node = 2298803501

Distancia recorrida por A*:

24762.787000000002 metros



P 3

orig_node = 103098680

dest_node = 42499712

Distancia recorrida por A*:

28879.74999999999 metros



P 4

orig_node = 103098680

dest_node = 42826843

Distancia recorrida por A*:

18772.149999999998 metros



P 5

orig_node = 103098680

dest_node = 103238167



IDA*

Times Square, Manhattan, NYC

P 1

orig_node = 103098680

dest_node = 42498881

Distancia recorrida por IDA*:

63117.78700000002 metros



P 2

orig_node = 103098680

dest_node = 7717125068

Distancia recorrida por IDA*:

60396.75700000002 metros



P 3

orig_node = 103098680

dest_node = 42499712

Distancia recorrida por IDA*:

73076.27799999995 metros



P 4

orig_node = 103098680

dest_node = 42826843

Distancia recorrida por IDA*:

78678.50300000003 metros



P 5

orig_node = 103098680

dest_node = 103238167

Distancia recorrida por IDA*:

4708.427000000001 metros

WALK VS DRIVE

Supongamos que un usuario está en un lugar y decide ir a otro lugar, pero quiere saber que le conviene más, si caminar o manejar para llegar a su destino más rápido considerando la distancia.

Hemos importado los mapas para "walk" (para obtener un grafo que tenga en cuenta rutas peatonales) y "drive" en donde para ambas se

considera lo siguiente:

- 'highway' (determina si es calle en donde un coche puede transitar),
- 'stop' (determina donde hacer stop para calles donde no puede pasar un coche),
- 'street_count' (conteo de camino de calles)

El programa toma una pareja aleatoria (nodo origen y nodo destino) para ambos mapas y determina las rutas usando el algoritmo BFS para determinar cual es más corta, considerando que el peaton puede cruzar por calles peatonales y el coche no.

CONCLUSIÓN

En este estudio, hemos realizado una exhaustiva evaluación de cinco métodos de búsqueda: Breadth-First Search (BFS), Uniform Cost Search (UCS), y A* han surgido como los enfoques más eficientes, demostrando consistentemente un rendimiento superior en la búsqueda de rutas. Estos algoritmos destacaron por su capacidad para generar rutas eficientes, equilibrando de manera efectiva la exploración exhaustiva y la heurística. Por otro lado, tanto Depth-First Search (DFS) como Iterative Deepening A* (IDA*) presentaron desafíos significativos, mostrando una tendencia a complejizar la tarea y producir rutas menos eficientes en comparación con sus contrapartes. Estos resultados subrayan la importancia de considerar la eficiencia y la complejidad al seleccionar métodos de búsqueda para aplicaciones específicas de rutas, ofreciendo valiosas perspectivas para la implementación práctica de algoritmos en contextos del mundo real.

REFERENCIAS

- GeeksforGeeks. (2023, June 9). Depth first search or DFS for a graph. <https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/>
- GeeksforGeeks. (2023, June 9). Breadth first search or BFS for a graph. <https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/>
- GeeksforGeeks. (2023, April 20). Uniform Cost Search Dijkstra for large Graphs. <https://www.geeksforgeeks.org/uniform-cost-search-dijkstra-for-large-graphs/>
- GeeksforGeeks. (2023, March 8). A search algorithm. <https://www.geeksforgeeks.org/a-search-algorithm/>
- GeeksforGeeks. (2023, March 28). Iterative Deepening A algorithm IDA Artificial Intelligence. <https://www.geeksforgeeks.org/iterative-deepening-a-algorithm-ida-artificial-intelligence/>
- AlgorithmSin. (s.f.). IDA-Star(IDA*) Algorithm in general. De: <https://algorithmsinsight.wordpress.com/graph-theory-2/ida-star-algorithm-in-general/>

Link al código:

<https://colab.research.google.com/drive/1WPFTMoP7ZaZzAMY2Z9VCMGJgO3DIERbd?usp=sharing>

CONCLUSIONES PERSONALES

MOÍSES PINEDA

Este análisis detallado de algoritmos de búsqueda y su aplicación en la búsqueda de rutas ha proporcionado una perspectiva fascinante sobre la complejidad y la eficiencia inherentes a este proceso. Haciéndome reflexionar sobre la conexión práctica de estos resultados con plataformas de navegación ampliamente utilizadas, como Google Maps. La comprensión de que la eficiencia en la búsqueda de rutas implica un equilibrio cuidadoso entre la exploración exhaustiva y el uso inteligente de heurísticas me lleva a apreciar aún más la complejidad detrás de los servicios de mapas en línea. La clara disparidad en la eficiencia, especialmente con DFS, resalta la importancia de algoritmos avanzados para garantizar rutas óptimas en entornos del mundo real.

EMILIO BERBER

En este proyecto de búsqueda de rutas entre nodos utilizando cinco algoritmos distintos (BFS, DFS, UCS, A* y IDA*), he llegado a la conclusión de que la elección del algoritmo depende en gran medida de las características específicas del problema y los requisitos del sistema. La eficiencia de cada algoritmo varía dependiendo de las características específicas del problema que intenta resolver, es por eso que es necesario conocer a fondo los 5 e implementarlos para encontrar sus diferencias. El algoritmo BFS y DFS son efectivos para explorar estructuras de grafos, pero suelen carecer de optimización en términos de tiempo y recursos. Por otro lado, UCS destaca al considerar el costo de las aristas (que en este caso son las distancias), mientras que A* y IDA* ofrecen una mayor eficiencia al incorporar heurísticas para estimar la mejor ruta.

La elección adecuada de cada algoritmo puede marcar la diferencia en la eficiencia del sistema, especialmente en entornos donde el tiempo y los recursos son críticos, como ejemplos: Waze, Google Maps, Maps, etc. Por ejemplo para este caso, en donde quisieramos aplicar una especie de Waze, podemos observar que es mejor el BFS que el DFS para encontrar la mejor ruta entre dos nodos.

SAMUEL GARCÍA

En general, ha sido muy interesante comenzar a meternos a aplicaciones reales de los algoritmos que vemos en clase, tanto durante nuestra exploración de redes neuronales humanas como al ejecutar los casos de estos algoritmos. Al comparar los resultados de estos algoritmos, podemos ver muy claramente cuáles manejan este tipo de aplicaciones con mayor eficacia (UCS, A*, e IDA*), y cuáles quizás podrían ser mejor utilizados para otros casos (BFS, DFS). Sin embargo, basta con los conocimientos vistos en clase para llegar a esta conclusión aun antes de correrlos, ya que sabemos que los primeros 3 están diseñados para encontrar eficientemente la ruta más corta, mientras que los últimos operan haciendo una búsqueda exhaustiva. Me llama mucho la atención pensar en los ingenieros que trabajaron en las aplicaciones de mapa que llegaron a tener un impacto en nuestra sociedad, y que en algún momento todo su trabajo se centraba al rededor de encontrar el mejor algoritmo y la mejor manera de alimentarlo con información. Los algoritmos son el centro de cualquier aplicación y gobiernan su eficaz funcionamiento, por lo que nunca hay que perderlos de vista como ingenieros en computación.