مجلة العلوم البحثة والتطبيقية
Journal of Pure & Applied Sciences
www.Suj.sebhau.edu.ly ISSN 2521-9200
Received 27/05/2019 Revised 25/08/2019 Published online 11/12/2019

A Web Application for Typical Shortest Path of Roads Networks Based on an Improved Dijkstra Algorithm

*Ibtusam A. Alashoury[a], Mabroukah A. M. Amarif[b]
[a]Department of Computer Sciences, Faculty of Sciences/Sebha University, Libya
[b]Department of Computer Sciences, Faculty of Information Technology/Sebha University, Libya
*Corresponding author: Ibt.alashouri@sebhau.edu.ly

Abstract Google Map is one of the most common applications which have been used for calculating distance and time between one point, as a source, and another, as a destination, within specific roads networks. This application often uses a heuristic function to find the shortest path between two points. This function is stopped if the shortest path has been found within a specific area of roads map. In fact, there is no guarantee that the result path is the shortest one ever. This paper presents a novel web application for calculating the typical shortest path between two points of roads networks within a specific area. The application reads an extracting map from Google application and calculates and reduces the searching time by displaying the overall an analytic information of the source and destination point including the explanation if that path is a part of the pervious solution path or not. The application adopts the improved of Dijkstra algorithm which has been developed by emerging a perfect data structure within it. The data structure, which is an array list of linked hash map, is used as storage for solution path. The result shows that the searching time for an ideal shortest path is reduced comparing with the original Dijkstra algorithm if the path is an implicit path. We believe that our application provides researchers with an accurate analysis of different time spent in the search and gives valuable results of the shortest route between any two points within roads networks.
Keywords: Dijkstra Algorithm, Google map, linked hash map, roads networks, Web Application.

تطبيق ويب لمسار أقصر نموذجي لشبكات طرق اعتماداً على خوارزمية ديكسترا المحسنة

*ابتسام عبد السلام العاشوري[1] و مبروكه علي امعرف[2]
[1] قسم علوم الحاسب–كلية العلوم–جامعة سبها، ليبيا
[2] قسم علوم الحاسب–كلية تقنية المعلومات–جامعة سبها، ليبيا
*للمراسلة: Ibt.alashouri@sebhau.edu.ly

الملخص يُعد Google Map واحداً من أكثر التطبيقات شيوعاً التي تم استخدامها لحساب المسافة والوقت بين نقطة واحدة، كمصدر ، وأخرى ، كهدف، داخل شبكات طرق محددة. غالباً ما يستخدم هذا التطبيق الدالة heuristic ذات الوظيفة الاستكشافية للعثور على أقصر مسار بين نقطتين. يتم إيقاف هذه الوظيفة إذا تم العثور على أقصر مسار في منطقة معينة من خريطة طرق. في الواقع، ليس هناك ما يضمن أن المسار النتيجة هو أقصر مسار على الإطلاق. تقدم هذه الورقة تطبيق ويب جديداً web application لحساب أقصر مسار نموذجي بين نقطتين لشبكات الطرق داخل منطقة معينة. يقرأ التطبيق خريطة مستخرجة من تطبيق google ويحسب ويقلل وقت البحث من خلال عرض معلومات تحليلية شاملة عن المصدر والوجهة المقصودة، بما في ذلك التوضيح ما إذا كان هذا المسار جزءاً من مسار الحل السابق أم لا. يعتمد التطبيق على تحسين خوارزمية Dijkstra التي تم تطويرها من خلال إنشاء بنية بيانات مثالية داخلها. تُستخدم بنية البيانات، وهي an array list of linked hash map، كمخزن للمسار الحل. توضح النتيجة أن وقت البحث عن مسار أقصر مثالي يتم تقليله مقارنة بخوارزمية Dijkstra الأصلية إذا كان المسار مساراً ضمنياً. نعتقد أن هذا التطبيق يوفر للباحثين تحليلًا دقيقًا لقيم الوقت المختلفة المستغرقة في البحث ويعطي نتائج قيّمة لأقصر طريق بين أي نقطتين داخل شبكات الطرق.
الكلمات المفتاحية: خوارزمية ديكسترا، خريطة قوقل، الهاش المتصلة، شبكات طرق، تطبيق ويب.

Introduction

In computer Sciences, routing maps is usually described as a graph data structure which represents the relationship between two of connected objects [1]. Objects are defined as a set of nodes (Vertices) and the connection between these objects is denoted as edges that link these nodes. Each edge is marked with a weight value describing the cost between the connected nodes. There are two types of graph; directed graph for which each node is directed by one way to any other node, undirected graph where all edges are bidirectional and it's possible to go to and back from the same way between two connected nodes. Google Maps is a large Web-based application that provides various services including information about geographical regions sites around the world. One of these important services is a route planner which offers shortest distance and time of directions between two nodes; source and destination. The application displays route or path in meters together with time cost. However, the given path is not the accurate shortest path

between the given nodes. That is because the application often uses the heuristic function to calculate the shortest time. This function stops after finding the first shortest path regardless of the other paths within a map.

Nowadays, researchers have improved existing algorithms or created new one to find the typical shortest path between nodes within a minimum time cost. These algorithms depend on the graph and path type. The most common existing algorithm is Dijkstra algorithm which is used for finding the shortest path between source and destination [2]. This algorithm is the most popular algorithm in the area of finding shortest path from single source to a node destination, or multiple nodes destination within a graph [3-5]. It can also be used to find the shortest route costs from the source to the destination by stopping the algorithm once the shortest route is set to the specific target. However, the Dijkstra algorithm has been under consideration until this time. Different data structure is used for supporting searching and storing the solution path within different graph data structure. This research adopts an improved Dijkstra algorithm which has been developed by us [6]. The improved algorithm uses Dijkstra algorithm with priority queue implemented by min heap to find the solution path (shortest path). The solution path is stored into data structure of array list contains of a linked hash map elements. We have tested the improved algorithm with different graphs sizes up to 10000 nodes and the results have shown that it is the best compared to the Dijkstra algorithm with priority queue data structure [6]. In order to test our improved algorithm using real data, we have developed a web application based on it. The web application is used for calculating the typical solution path of real roads networks extracted from a Google Map. The following section describes the most related works to our paper while section III explains the design and implementation of the web. Analysis and results are described in section IV. A discussion and conclusion of this paper is explained in section V.

I. Related work

Dijkstra algorithm has been considered by many researchers to improve the shortest path cost by minimizing the searching time (time complexity) using different data structure. Jain et. al. improve the Dijkstra algorithm by using priority queue and linked list [7]. It has been noticed that by using a graph represented by their adjacncey lists and the priority queue implemented as a min-heap, the time efficiency is in O(|E| log |V |), where V is the number of nodes and E is the number of edges which connect these nodes. if the priority queue is implemented using an advanced data structure called the Fibonacci heap, the time becomes O(|V| log V+E), and its imporved [8].

Time efficiency could be imporved by exploit the solution path to get the implicit paths if they are queried again. From the literature, most algorithms may provide this feature, but there is no such explanation or imporvement of it. A suitable data structure could improve the time efficiency for the whole algorithm if used propably for storing the solution path and then search for the implicit path. In [6], we have designed an algorithm based on Dijkstra algorithm by using a special data structure (linked hash map) to store the solution path. The designed algorithm is tested with different number of nodes using random graphs data structure. The results justify the effectiveness of the improved algorithm[6]. Moreover, we have decided to develop a web application to test the improved algorithm with real roads networks. The following section explains the design and implementation of the developed web.

II. Web application design

The web application has been designed and implemented using Java and Java Script languages. The web design languages such as html and CSS have also been used with tomcat server. The multithreading programming paradigm is adopted in the web design in order to compare between the improved algorithm and Dijkstra algorithm shortest path cost. In the designed web, DSA denotes the improved algorithm and DJX denotes the Dijkstra algorithm with priority queue. The results are displayed once the shortest path has been reached either by DSA or DJX. The web also describes whether the solution path is an implicit path or not. The solution path route is displayed within the road network on the map with the help of cytoscape library. The time cost is described using statistical charts graphics with the help of chartjs library. Figure 1 shows the main interface of the web.

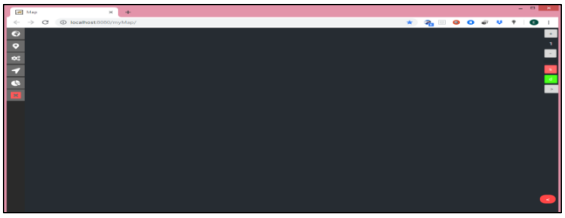

Fig. 1: The web main interface

The right tabs of the interface refer to the functions and operations of the web and left tabs refers to the setting and operations that should be applied to the chosen road map. Figure 2 describes the open street website tab.

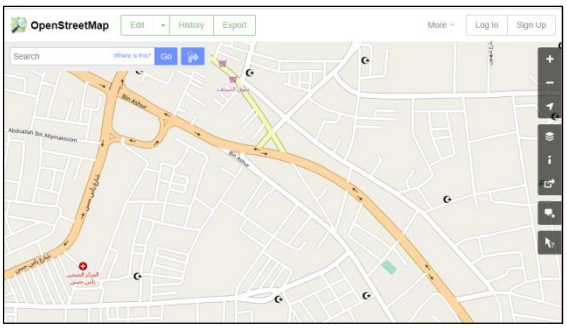

Fig. 2: The open street website interface

After choosing the town and the area from the map, the complete information about the chosen area is loaded as seen in figure 3.

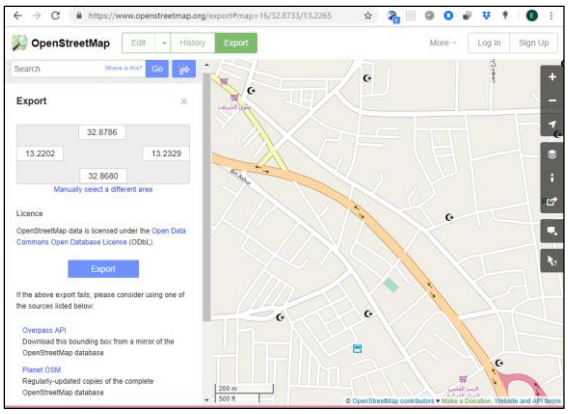

Fig. 3: The area information

The web application enables us to save the chosen roads map. It also enables us to choose the source and destination nodes to find the shortest path if available. The web displays all possible information about the shortest path including the specific time cost of both algorithms according to CPU of the device, size of data structure, and time averages. Information is displayed in different graphics. Figure 4 describes the different statistical graphics of shortest paths information.

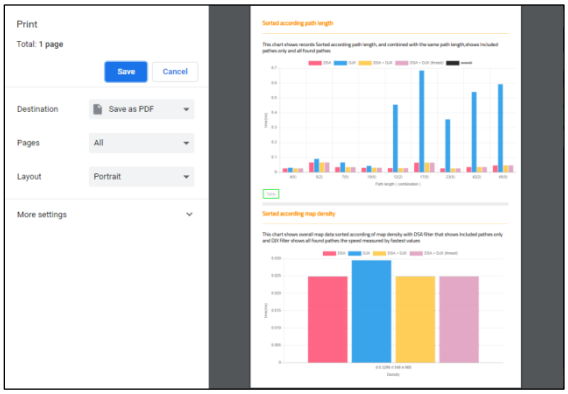

Fig. 4: The shortest path information

I. Analysis and Results

The time complexity of hash-map, linked-hash-map and array-list are different according to the kind of operations [6]. In this paper, n means the number of nodes (V), and m means the number of edges (E). The total time for putting the new solution path in the given data structure is:

$O(nlogn) + O(n) + O(nlogn) = O(n) + O(2nlogn) = O(2nlogn) = O(nlogn)$

In the other side, the total time complexity of the query from the data structure is:

$O(logn)+O(logn)+O(nlogn)+O(logn)+O(logn)$
$=O(4logn)+O(nlogn)$
$=O(logn)+O(nlogn)$
$=O((1+n)logn) =O(nlogn)$

We run the DJX and the DSA algorithms with different graphs type. We also run both of them with different number of nodes in order to get the shortest path between a given source node and destination from the chosen roads map. After the solution path is found, it's stored in the given data structure (array-list of linked-hash-map). Then, an inquiry for the implicit path is taken place. If the implicit path is found within the stored solution path in the given data structure, the time is calculated and recorded, else; the searching using DJX algorithm is started again and the total time is recorded. The averages of each recorded times are calculated. These operations are repeated many times with different number of nodes. The web application results emphasize and confirm the validity of the improved algorithm DSA. Figure 5 and 6 describe the running web results.

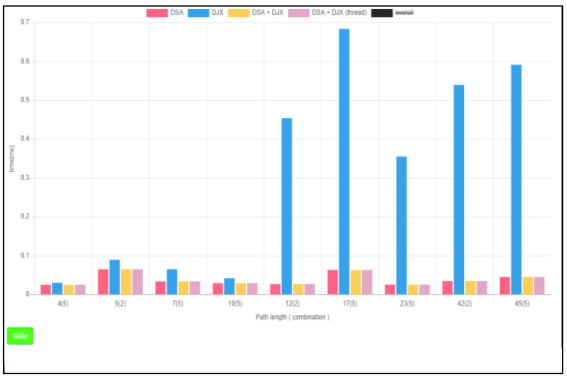

Fig. 5: The shortest path comparison

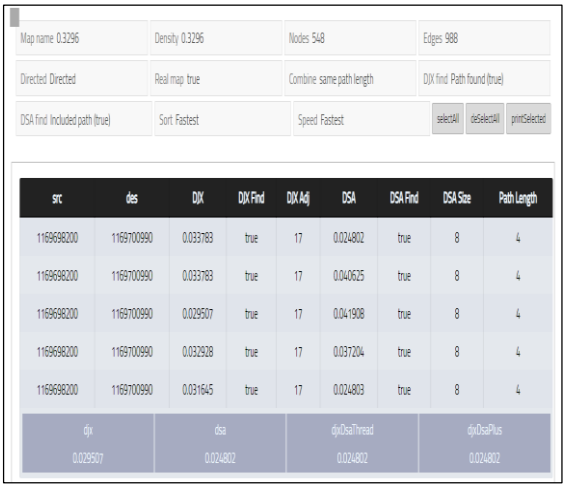

Fig. 6: The shortest path data information

I. Discussion and Conclusion

We have developed a web application which adopts the improved algorithm DSA [6] in order to consider and test a real data value of a case study of road map with one way road (directed graph) and two way road (undirected graph), with different number of nodes. We have justified that our improved algorithm DSA works better if the graph is undirected. The same case study has been given to DJX with priority queue implemented as a min-heap. The results of both algorithms have been recorded and analysed.

Comparisons between DSA and DJX have shown that DSA is almost the best. Although more data structures have been used within the proposed algorithm, however, the enlarged storage is available for all of the current devices, even for the smallest one. We argue that data storages aren't problem if the performance of the given algorithm is higher and success.

It has been observed that the searching time of DJX is almost the same as the time of DSA if the required path is not an implicit path within the solution path. Actually, the designed web application is able to run both DSA and DJX, and stopped once any of them has reached the exact solution path. This web application can adopt any improved algorithm in order to test the performance and searching time cost.

References

[1]- W., B., Douglas, Introduction to Graph Theory, Pernice Hall, 2001.

[2]- Dijkstra., E., W., (1959) "A note on Two Problems in Connexion with graphs", Numerische Mathematik, 1, 269-271.

[3]- Cormen, T., Leiserson, C., Rivest, R., & Stein, C., (2009) Introduction to Algorithms, 3rd. ed., MIT Press, London.

[4]- A., Levitin, Introduction to the Design and Analysis of Algorithms, 3rd ed., Pearson Education, Inc., Addison-Wesley. 2012.

[5]- Deng, Y., Chen, Y., Zhang, Y., & Mahadevan, S., (2012) "Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment," Applied Soft Computing, vol. 12, pp. 1231-1237.

[6]- Amarif , M., Alashoury, I., (2019), The Implicit Path Cost Optimization in Dijkstra Algorithm using Hash Map Data Structure., International conference on Computer Sciences & Infromatioon Technology COSIT2019, pp. 33–44. DOI: 10.5121/csit.2019.90204

[7]- Jain, A., Datta, U., & Joshi, N., (2016) "Implemented modification in Dijkstra‟ s Algorithm to find the shortest path for N nodes with constraint," International Journal of Scientific Engineering and Applied Science, vol. 2, pp. 420-426.

[8]- Cormen, T., Leiserson, C., Rivest, R., & Stein, C., (2009) Introduction to Algorithms, 3rd. ed., MIT Press, London.