Kafrelsheikh Journal of Information Sciences ISSN (Online): 2535-1478, ISSN (Print): 2537-0677 Volume 5, Issue 1, 2025, PP. 1–25 Date of publication 3 April 2025

# An Enhanced Model for Dijkstra Algorithm: Case Study Nearest Hospital in Greater Cairo-Egypt

Hytham M. Ismail[1], Mohamed El-Mekawy[2], Mona Nasr[3], and Mohamed Belal[4].

[1]Faculty of Computers and Artificial Intelligence, Beni-Suef University, Egypt.

[2]Department of Computer and Systems Sciences, Stockholm University, Sweden.

[3, 4] Faculty of Computers and Information, Helwan University, Egypt.

[1]*hytham.mahmoud@fcis.bsu.edu.eg*, [2]*moel@dsv.su.se*, [3]drmona_nasr@fci.helwan.edu.eg, [4]dr.mohamedbelal@gmail.com.

## ABSTRACT

The COVID-19 pandemic has increased the need for people to locate nearby healthcare facilities quickly, especially in densely populated countries like Egypt, where transportation networks are complex. The Dijkstra algorithm, commonly used to solve shortest path problems, traditionally focuses on a single destination. An enhancement for existing Dijkstra algorithm is required able to select the nearest location from multi-destinations. This study introduces an enhanced version of the Dijkstra algorithm - Enhanced Dijkstra Shortest Path Algorithm (EDSPA) - that accommodates multiple destination nodes by utilizing an endpoint list. This improvement significantly reduces execution time, enabling quicker identification of the nearest hospital and ensuring timely treatment for patients, particularly within the critical "golden hour." The proposed algorithm is adaptable to various geographic area zones with approximately the same execution time. This study provides the efficiency of the proposed algorithm by implementing the enhanced and existing Dijkstra algorithms using the same data sets and comparing the results.

**Keywords:** Shortest Path; Dijkstra's algorithm; nearest hospital; QGIS; Enhanced Dijkstra Algorithm

## 1. Introduction

The COVID-19 pandemic has highlighted the critical need for efficient health infrastructure and timely access to medical facilities. One of the main challenges facing health systems, especially in regions hit hard by the virus, is to ensure that patients, especially those who need urgent care, can reach hospitals as quickly as possible [1]. Shortest path algorithms are fundamental in fields as diverse as transport networks, communication systems, robotics and logistics. These algorithms are used to find the optimal path between two nodes in a graph, where nodes represent places (such as cities or hospitals) and edges represent connections or routes between these places. In the context of the COVID-19 pandemic, shortest path algorithms have gained attention for their ability to optimize routes for ambulance, providing health care and transporting patients to hospitals. The shortest route algorithms have emerged as a powerful tool to optimize routes and help individuals, medical personnel or distribution systems to find the nearest hospital in real time, especially in times of crisis such as pandemics [2]. There are many shortest path algorithms, Dijkstra's Algorithm, Bellman-Ford Algorithm [3], A* (A-star) Algorithm [4] and Floyd-Warshall Algorithm [5] are the most well-known shortest path algorithms.

Dijkstra's algorithm, proposed by Edsger W. Dijkstra, is an algorithm used to find the shortest path between nodes in a graph that can represent road networks [6]. Dijkstra's algorithm is widely used in network analysis to determine the most efficient path between two points in a graph [7]. This can be particularly useful for finding the fastest route to hospitals in an emergency. In the case of COVID-19, where rapid medical intervention is crucial, implementing algorithms can significantly improve the speed with which individuals reach healthcare facilities [8]. By modeling cities or regions as graphs, where nodes represent locations (including hospitals, clinics, and residential areas) and edges represent roads or transportation routes, Dijkstra's algorithm can be used to calculate the shortest path from a given location (such as a patient's home or ambulance station) to the nearest hospital.

Dijkstra's algorithm is considered the best algorithm to extract the shortest path for non-negative weight [9]. The Dijkstra's algorithm determines the least-cost path from a source point to the destination [6]. The pseudocode of the Dijkstra's algorithm is shown in (Figure 1) [10]. It shows how the algorithm finds the minimum distance path for the given start to the end point.

```
$function\ Dijkstra(Graph,Start,End)$:
    $for\ each\ vertex\ V\ in\ graph$: //initialize
        $dist[v]=infinity,\ previous[v]=undefined$;
    $dist[start]=0$;
    $Q=the\ set\ of\ all\ nodes\ in\ Graph$;
    $while\ Q\ is\ not\ empty:$
        $u=vertex\ in\ Q\ with\ smallest\ dist[]$;
        $if\ dist[u] = \emptyset:\ break$;
        $if\ u= end: break$;
        $remove\ u\ from\ Q$;
        $for\ each\ neighboring\ point\ v\ of\ u$:
            $alt=dist[u] + length(u,v)$;
            $if\ alt<dist[v]$:
                $dist[v]=alt,\ previous[v]=u$;
    // read the least distance path
    $E= empty\ sequence$;
    $u=End$;
    $while\ previous\ [u]\ is\ defined$:
        $insert\ u\ at\ the\ beginning\ of\ E, u=previous[u]$;
    $return\ E$
```

*Figure 1 Pseudocode for Dijkstra's algorithm [10]*

The complexity of Dijkstra's algorithm is as follows:

Let graph $G=(V, E);$ where $V$ represents vertices in the graph and $E$ represents the edges of the graph. It takes $O(V)$ time to build the initial priority queue of vertices $V$ [6]. Each of the following priority queue operations takes $O\ (log\ q)$ time where $q$ is the current queue size. Each vertex is dequeued exactly once, after taking the least-cost path from the source vertex. After dequeuing, each neighbor $v$ of vertex $u$ is tested to see if the path from the source to $v$ via $u$ has a lower cost than that of the current path from the source to $v$. If a lower cost path is obtained through $u$, then the cost of the path to $v$ is reduced and the priority of the vertex is changed in the queue. Thus, the path improvement test is performed a total of $O\ (E)$ times with a worst-case time of $O\ (log\ V)$ to update the vertex priority for each test. Therefore, the algorithm runs in time $O\ (E\ log\ V)$ [11].

$$Dijkstra\ Complexity = O(E\ log\ V); \qquad equ.\ 1$$

where E and V are the graph edges and vertices.

The first algorithm created by Dijkstra was designed to find the shortest route between two given nodes, but a more widespread variation can find the shortest paths from one source point to all other nodes [12]. (Huang, Yi, & Shi, 2013) provided improvement for Dijkstra algorithm incorporates a weight constraint function to solve data structure storage errors, such as large space and availability [13]. (Chadha & Garg, 2019) implemented various algorithms, Dijkstra, Bidirectional Dijkstra and A* algorithms, to calculate the optimal path between two nodes [14]. (Das, Ojha, Kramsapi, Baruah, & Dutta, 2019) performed proper analysis of the present road network, geo dataset of vertices and edges was created. Then origin and destination vertices were selected to extract the shortest route using Dijkstra algorithm [15]. Dijkstra's algorithm was applied to extract the shortest path to all nearest hospitals to an emergency location in case of an accident, then adapted a fuzzy logic algorithm to search among nearest hospitals based on many other factors i.e. the severity of the accident, the distance, the availability of medical equipment and etc. by (Gabriel, Lolade, Durodola, & Orimoloye, 2019) [2]. Although (Tang, Zhou, Geng, & Sun, 2019) also use Dijkstra algorithm on emergency rescue, the optimal route extracted from emergency rescue point to accident point [9]. Dijkstra algorithm also used in tourist applications, (Anam & Yunus, 2019) implement Android GIS-based information system to find the nearest tourist places [16]. In criminal cases, also extraction of shortest path is required, (Abd Al-Munaf, Abdulahmeed, & Hussein, 2020) use QGIS and the Dijkstra algorithm to extract the shortest path between the accident places and the nearest relevant authorities' location. [17]. Shortest path algorithms not applicable for road networks only, (Cadieux, Kalacska, Coomes, Tanaka, & Takasaki, 2020) applied Dijkstra algorithm on river networks [18]. (Al Bager A. & Al Samani A., 2020) explore comparison between Dijkstra and Bellfam-Ford algorithms, which proved Dijkstra is more efficient to extract shortest path than Bellfam-Ford algorithm [19].

Planning for Robot's geomagnetic navigation also demand applied of Dijkstra algorithm [20]. (Bhardwaj 2021) integrate vehicle route problems with Dijkstra algorithm [21]. Dijkstra algorithm also, used in optimizing route for waste collection by (Sahu, Sharma, Sharma, Choudhury, & Dewangan, 2023) [22]. An enhancement to Dijkstra algorithm was applied by node combination to reduce the memory usage as explored by (Tiong, Panganiban, Blanco, Regala, & Cortez, 2022) [23].

This study provides enhancement of Dijkstra's algorithm to find the nearest hospital during the COVID-19 pandemic from available hospitals within specific zone. Also, this article claims that enhancement algorithm requires less execution time to extract the shortest path than existing one.

## 2. Materials and Methods

### 2.1. Proposed Architecture and Methodologies

This study aims to provide a model for Enhanced Dijkstra Shortest Path Algorithm (EDSPA) for the sake of discovering the nearest hospital and extract the shortest path to this hospital in Greater Cairo. To provide the patient with the shortest path from the patient's current location to the nearest hospital site, a method with four analysis stages is proposed (see Figure 2).

In the first stage, Data Preparation, patient's current location, road network, and available hospitals are identified.

Then in the next stage, nearest hospitals are determined by getting all hospitals within patient location range from the previous stage, on exists hospitals map layers, using a buffer approach in the Quantum Geographic Information System (QGIS) open-source tool. The main goal of this stage is to identify the *N* (specified minimum number) number of nearest hospitals. First, by using buffer method, get all hospitals within a range of one kilometer from the patient location. If the number of hospitals within one kilometer range is less than the specified N, then increase range area by one kilometer and repeat buffering method. Finally, the nearest hospitals' location map layer is generated.

In the third stage, the proposed Enhanced Dijkstra Shortest Path Algorithm (EDSPA) and existing Dijkstra Algorithm are used to specify the nearest hospital and extract the shortest path from patient location to this hospital location. The following two sub-sections are used to explore how to use current and enhanced Dijkstra Shortest Path Algorithms. The proposed and existing algorithms are applied on nearest hospitals' location, patient's location and road network layers using Python language and QGIS tool.

Finally, in the last stage, the GIS map contains patient's location, the nearest hospital's location and the shortest path between them is provided to patient to arrive at a suitable time.

### 2.1.1. Dijkstra Shortest Path Algorithm

According to the previous discussion, the existed Dijkstra Algorithm can be used to extract the shortest path between two nodes as shown in (Figure 1). So, to extract the overall shortest path the Dijkstra algorithm should be executed for each available hospital.

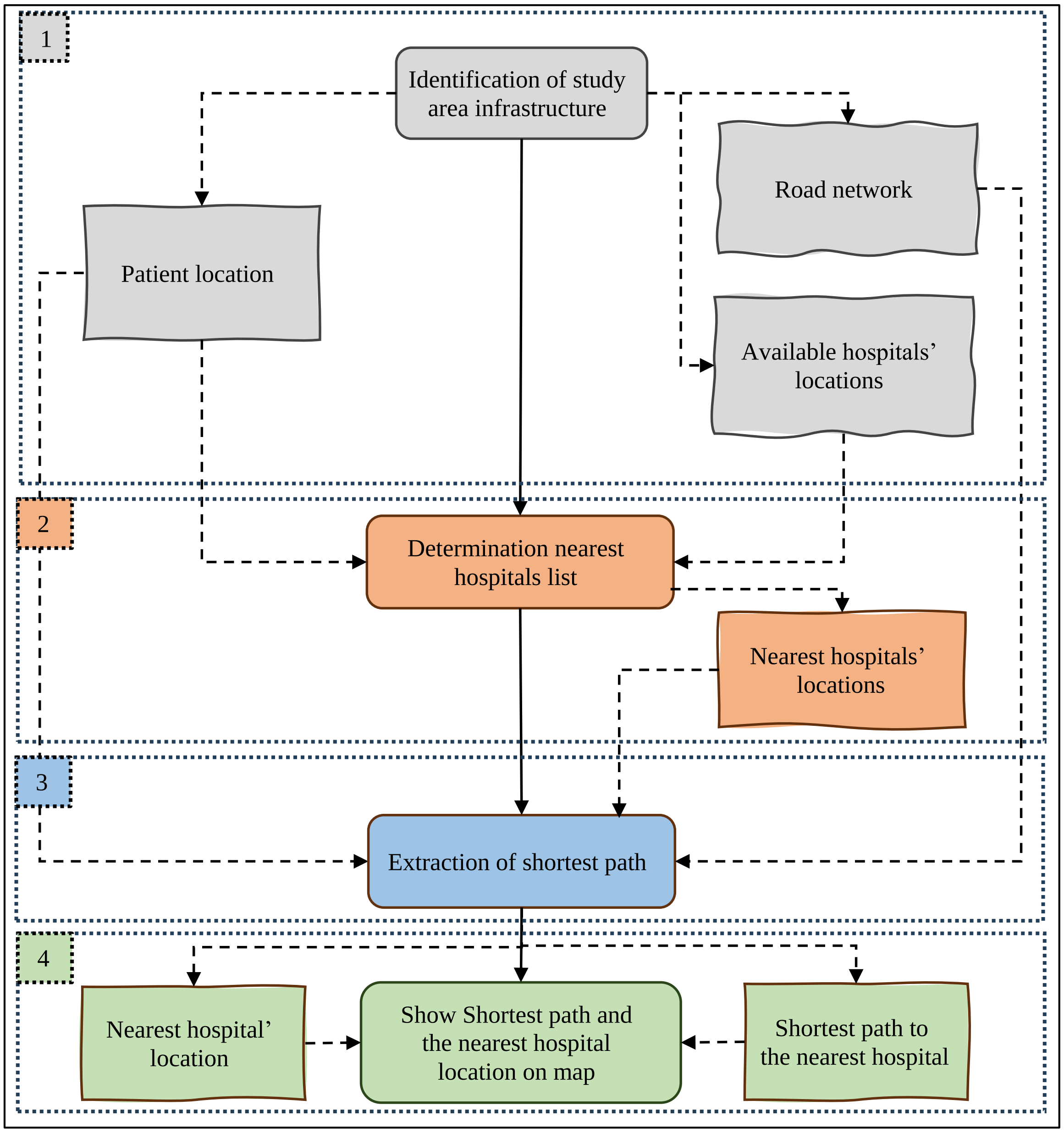

1
Identification of study area infrastructure
Road network
Patient location
Available hospitals' locations
2
Determination nearest hospitals list
Nearest hospitals' locations
3
Extraction of shortest path
4
Nearest hospital' location
Show Shortest path and the nearest hospital location on map
Shortest path to the nearest hospital

*Figure 2 Methodology Flowchart*

As shown in (Figure 3), to extract the overall shortest path, the Dijkstra algorithm extracts the shortest path between patient location and first hospital from nearest hospitals list. Then, the extraction shortest path is added to the shortest path list. Repeating these procedures for each hospital from the nearest hospitals list is required. Finally, comparing among the shortest path list is applied to get the overall shortest path and the nearest hospital associated to this shortest path.

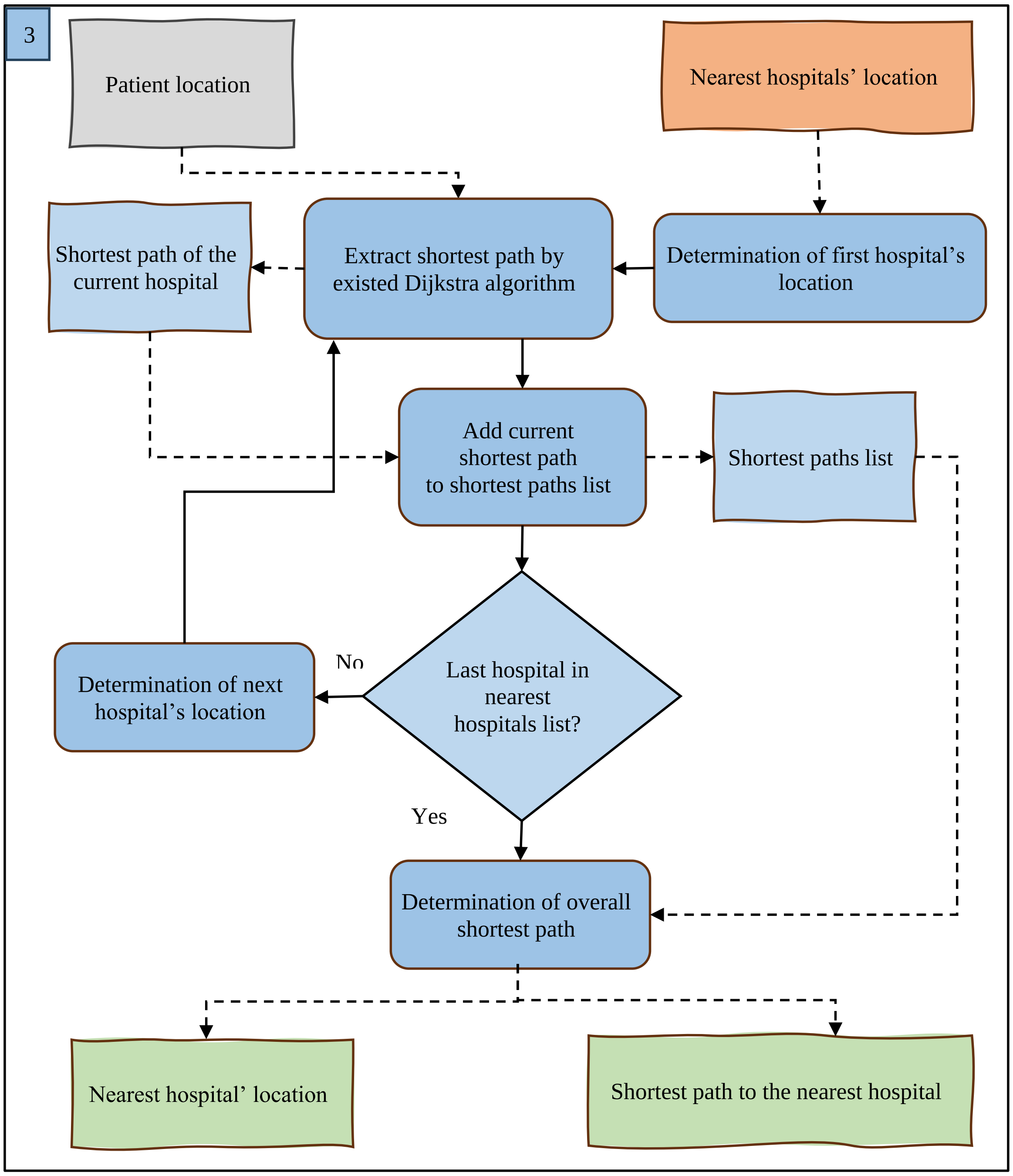

3
Patient location
Nearest hospitals' location
Shortest path of the current hospital
Extract shortest path by existed Dijkstra algorithm
Determination of first hospital's location
Add current shortest path to shortest paths list
Shortest paths list
Determination of next hospital's location
No
Last hospital in nearest hospitals list?
Yes
Determination of overall shortest path
Nearest hospital' location
Shortest path to the nearest hospital

*Figure 3 Extraction shortest path using existed Dijkstra Algorithm*

By using Dijkstra algorithm complexity as shown in (equ. 1) the complexity of the above model is:

$$Extraction\ \text{Complexity}\ using\ exists\ \text{Dijkstra}\ = O(N(E \log V));\qquad equ.\ 2$$

where E, V and N are the graph edges and vertices and the number of nearest hospitals.

This complexity is calculated by multiplying Dijkstra complexity with number of nearest hospitals, because the above model executes Dijkstra algorithm multiple times according to the number of hospitals.

### 2.1.2. Enhanced Dijkstra Shortest Path Algorithm (EDSPA)

```
function EDSPA (Graph,Start,EndList):
  for each vertex V in graph: //initialize
          dist[v]=infinity, previous[v]=undefined;
  dist[start]=0;
  Q=the set of all nodes in Graph;
  while Q is not empty:
          u=vertex in Q with smallest dist[];
          if dist[u] = Ø: break;
          if u in EndList: break;
          remove u from Q;
          for each neighboring point v of u:
                  alt=dist[u] + length(u,v);
                  if alt<dist[v]:
                          dist[v]=alt, previous[v]=u;
  // read the least distance path
  E= empty sequence;
  u=vertex in EndList with minimum dist;
  while previous [u] is defined:
          insert u at the beginning of E, u=previous[u];
  return E
```

*Figure 4 Pseudocode for Enhanced Dijkstra's algorithm*

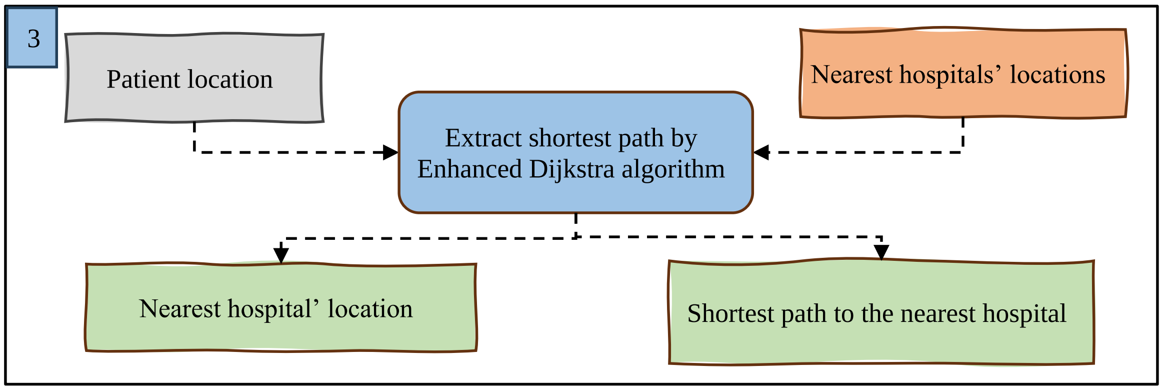

3
Patient location
Nearest hospitals' locations
Extract shortest path by Enhanced Dijkstra algorithm
Nearest hospital' location
Shortest path to the nearest hospital

*Figure 5 Extraction shortest path using Enhanced Dijkstra Algorithm*

As shown in (Figure 4) the iteration of nearest hospital list is executed within the while loop of vertices queue, so the number of vertices is multiplied by the number of nearest hospitals in complexity calculation as shown in (equ. 3). Also, because the proposed model executes Enhanced Dijkstra algorithm only once to get the overall shortest path, there is no need to multiple the algorithm complexity with number of hospitals to calculate the model complexity.

$$Extraction_{\ Complexity}\ using\ enhanced_{\ Dijkstra} = O(E\ log(NV)); \qquad equ.\ 3$$

where E, V and N are the graph edges and vertices and the number of nearest hospitals.

Comparing (equ. 2 and equ. 3), $O(E\log(NV)) < O(\mathrm{N}(E\log V))$, shows execution time of extraction shortest path using enhanced Dijkstra algorithm is dramatically less than execution time of extraction shortest path using existed Dijkstra algorithm.

## 2.2. Problem Definition and Case Study

In the context of epidemics, in critical health conditions such as heart disease, it is very important to choose the nearest hospital with the shortest path. Dijkstra's algorithm can take the shortest path between two points or all the shortest paths for all the given nodes. Using the shortest path between two points, the algorithm needs to be repeated for each available hospital, and the processing time is often too long. Also, it takes a long time to take all the shortest paths. This study proposes Enhanced Dijkstra Shortest Path Algorithm (EDSPA) to determine the nearest hospital and take the shortest path between the patient and the nearest hospital with the least processing time.

### 2.2.1. Study area

The Greater Cairo Area is the largest metropolitan area in Egypt (Chase, 2020), the largest urban area in Africa, the Middle East, and the Arab world, and is the sixth-largest metropolitan area in the world. It consists of all cities in the Cairo Governorate as well as Giza, Sheikh Zayed City in the Giza Governorate, and Shubra El Khima and Obour in the Qalyubia Governorate with a total estimated population of 20,296,000 people, an area of 2,010 km2, and a density of 10,099/km2 (Chase, 2020). It is allocated to 30°03'N and 31°22'E.

In addition to being the largest metropolitan area in Egypt, 20% of Egyptians (110,909,103 persons) live in Cairo. Furthermore, Greater Cairo has a complex collection of constraints and criteria. Because of this, Greater Cairo is considered a suitable case study area to implement the proposed methodology.

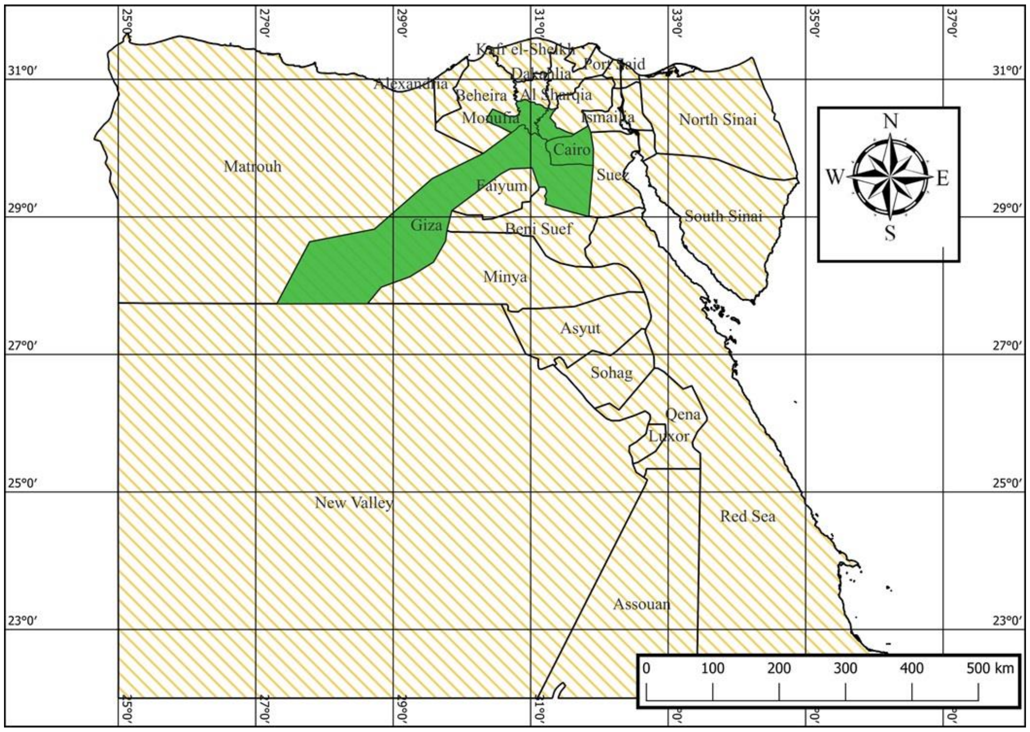


*Figure 6 Greater Cairo Study Area*

### 2.2.2. Data Source

OpenStreetMap is considered as a data source of vector layers which are used in this study. OpenStreetMap is a free, editable map of the whole world that is being built by volunteers largely from scratch and released with an open content [24].

- Data Source: OpenStreetMap
- Coordinate Reference System (CRS)
- Name: EPSG:4326 - WGS 84
- Units: Geographic (uses latitude and longitude for coordinates)
- Accuracy: Based on World Geodetic System 1984 ensemble (EPSG:6326), which has a limited accuracy of, at best, two meters.
- Extent:
    - 26.590, 27.197: 32.620, 31.116 for Area of Study map layers
    - 31.197, 30.016: 31.283, 30.071 for Selected Point Area map layers
- Scale:
    - 1:1400000 for Area of Study map layers
    - 1: 22393 for Selected Point Area map layers
- Date: updated to January 2021
- Datasets in vector layers and its description:

- Highway: line vector layer, contains different types of streets and roads (motorway, primary, secondary, and track).

The data sets which are used in this study are organized, managed, and validated by NextGIS [25].

Existing hospital: point vector layer, contains different types of hospitals (private, public, and international).

### 2.3. Enhanced Dijkstra Shortest Path Algorithm (EDSPA) Implementation

In this section, the proposed methodology is implemented in our case study, represented by shortest path for the nearest hospital in Greater Cairo on Egypt. The following sub-sections explain the implementation of the four stages of proposed EDSPA methodology.

#### 2.3.1. Data Preparation

In this sub-section, we determinate vector map layer for the study area using QGIS selection tool. The existing available hospitals layer, in the study area, were also established using QGIS intersection tool between all available hospitals and study area layers as shown in (Figure 7).

We then determined the greater Cairo Road network by extracting it from Egypt road network using study area vector layer as shown in (Figure 8).

Finally, we identified the patient location by selecting a random point within the study area and specifying this point coordinate ,30.04711 latitude and 31.23902 longitude, as shown in (Figure 9).

Zooming in for patient location and changing the extent of the map layer from 26.590, 27.197: 32.620, 31.116 to 31.197, 30.016: 31.283, 30.071, also the scale is raised from 1: 1400000 to 1: 22393 as shown in (Figure 9).

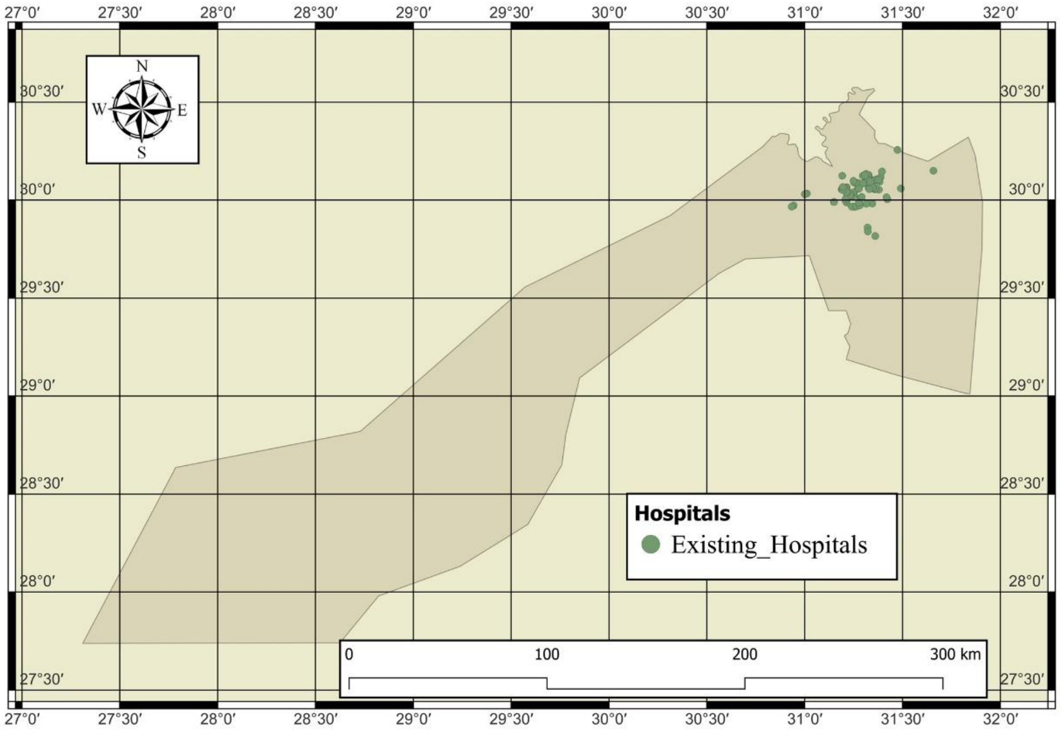


*Figure 7 Existed hospitals vector layer*

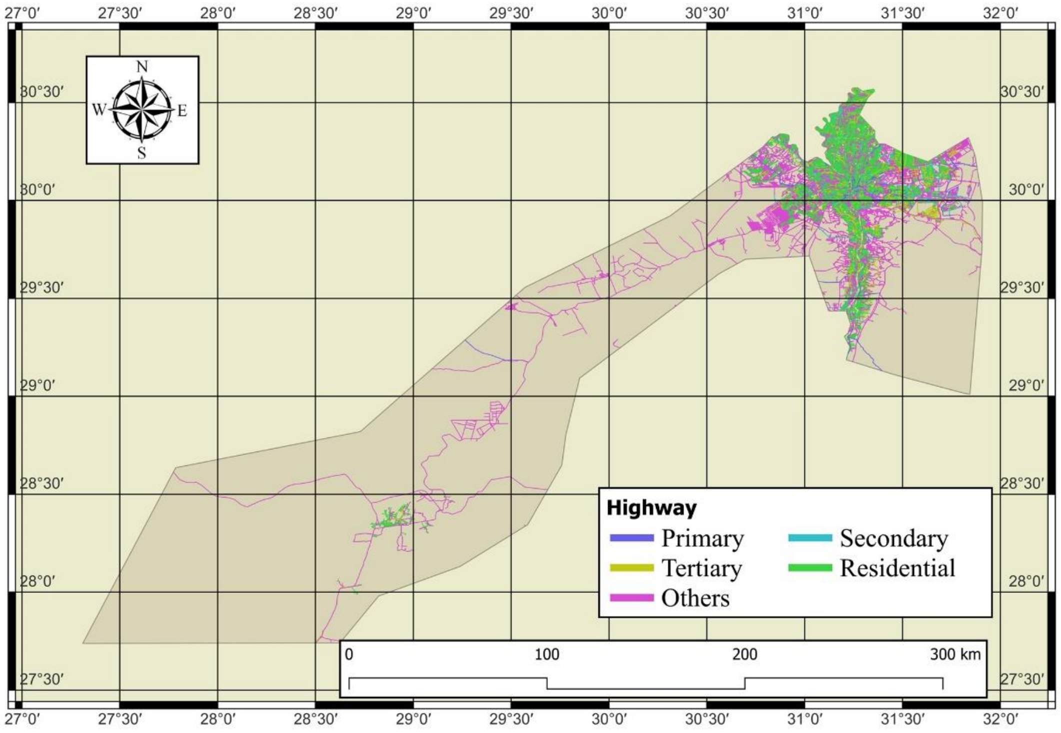


*Figure 8 Heighway vector layer*

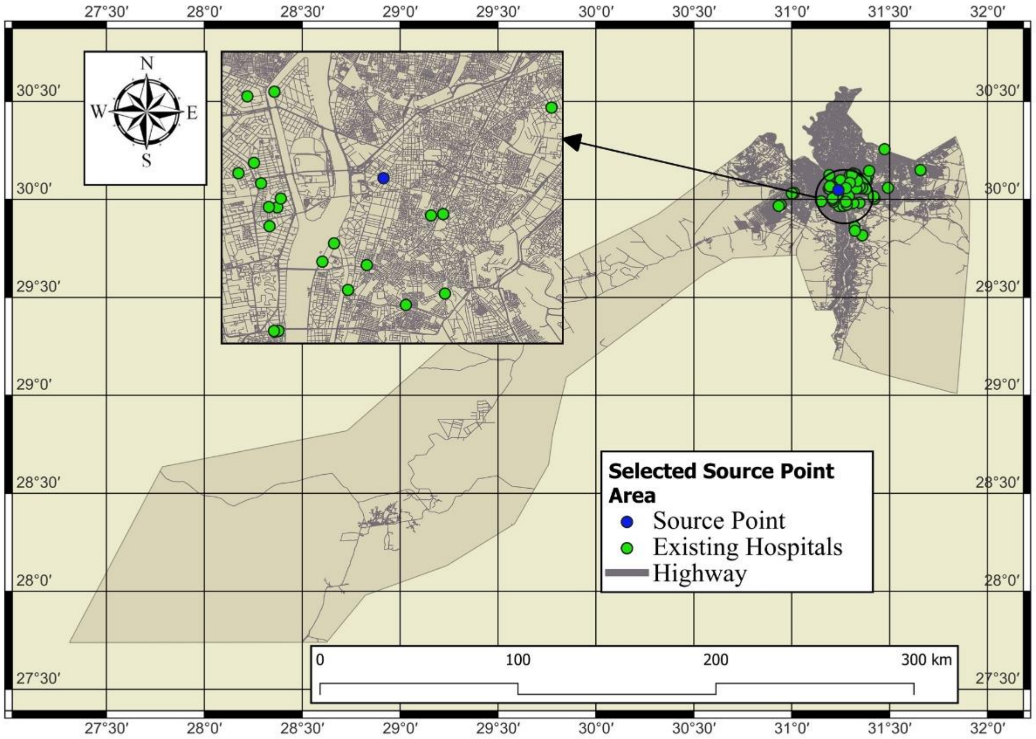


*Figure 9 Zoom in for Selected Source Point Area*

### 2.3.2. Nearest Hospitals Determination

This sub-section explains the second step on the proposed methodology, available nearest hospital determination. The nearest hospitals to the scene of the patient are identified by the buffer. Since the map is constructed with a projected coordinate system, Euclidean buffers are best suited to give the boundary accurately. Distance can be specified in linear units such as meters and kilometers and applied to the selected function. To find the nearest hospitals, a buffer distance of 1 km is first applied to check availability as shown in (Figure 10). If hospitals are not available in a buffer of 1 km, then the distance is increased by the radius of 1 km to get at least 3 hospitals [26]. The green points in Figure 10 represents the buffer zone of 2 km created around the patient location to identify the nearest hospital.

To study the effect of increasing the number of nearest hospitals on the existed and proposed algorithms, we increase buffer zone to 3 km and reidentify the nearest hospitals as shown in (Figure 9), this new buffer zone increase the nearest hospitals number from 4 hospitals to 15 hospitals as shown in (Figure 10).

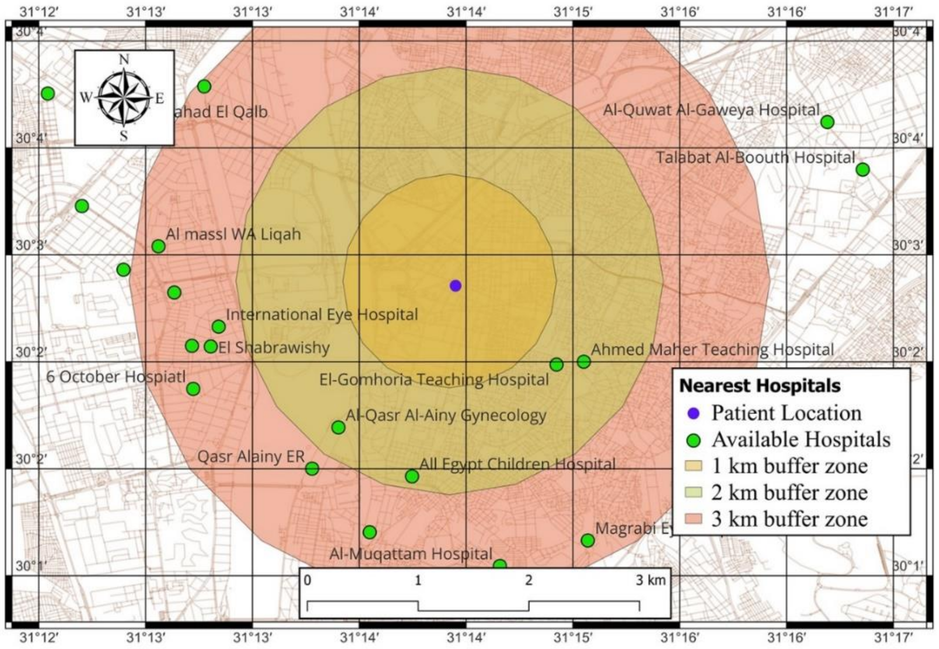


*Figure 10 Nearest Hospitals*

### 2.3.3. Shortest Path Extraction

Dijkstra's algorithm calculates the shortest path from a source node to all other nodes in the graph. However, it does not explicitly store the path, but only the minimum distances between the source and each node. Extracting the actual shortest path from the calculated results requires some additional steps. The simplest way to derive the shortest path is to keep the predecessor nodes while running Dijkstra's algorithm. On this study, there are multiple target nodes (available hospitals), so we first extract the shortest path for each target node then compare among these paths to get the overall shortest path, using classic Dijkstra algorithm. While by using the proposed enhanced Dijkstra algorithm, we extract directly the overall shortest path without compromising among multiple target nodes. In the following sub-sections, the shortest path is extracted using classic and enhanced Dijkstra algorithm within 2 kilometers buffer zone from patient location, 4 hospitals available.

#### 2.3.3.1. Existing Dijkstra Algorithm for 2 km buffer zone nearest hospitals

The existing Dijkstra Algorithm extracts the shortest path between one start and one target nodes. Also, existing Dijkstra can extract all shortest paths from source to all other nodes. The area of study (Grater Cairo Road network) consists of 1881161 vertices nodes, which makes extracting all the shortest paths the worst option. So, when applying the existing algorithm, the shortest path for each available hospital is extracted separately then comparing among them to show the nearest hospital.

For extracting the shortest path from patient location to “*Kasr Al Ainy Gynecology Hospital*” there are 12802 vertices are examined by Dijkstra Algorithm as shown in (Figure 11).

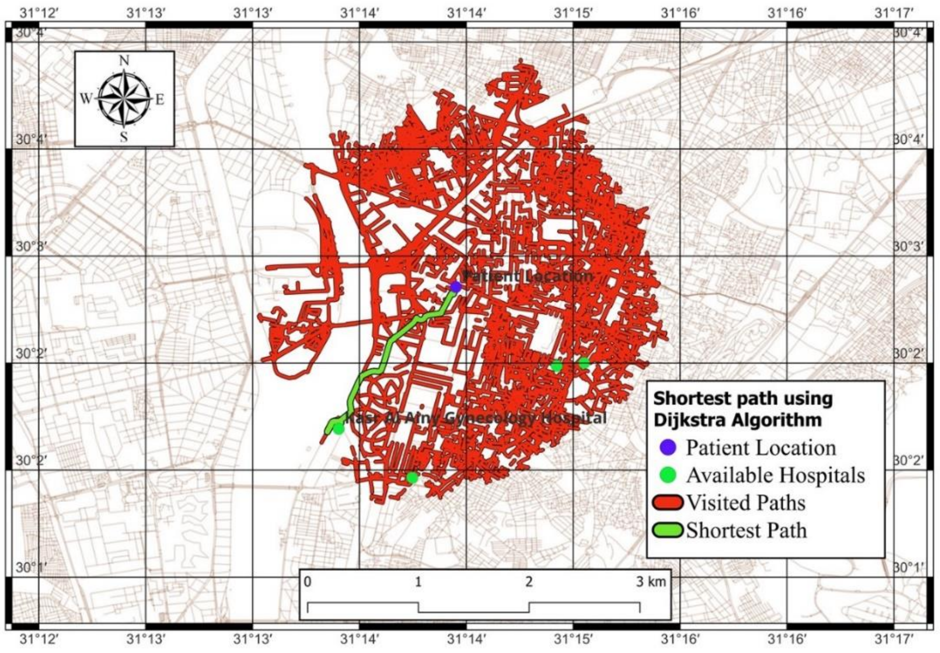


*Figure 11 Shortest path to Kasr Al Ainy Gynecology Hospital using Dijkstra Algorithm*

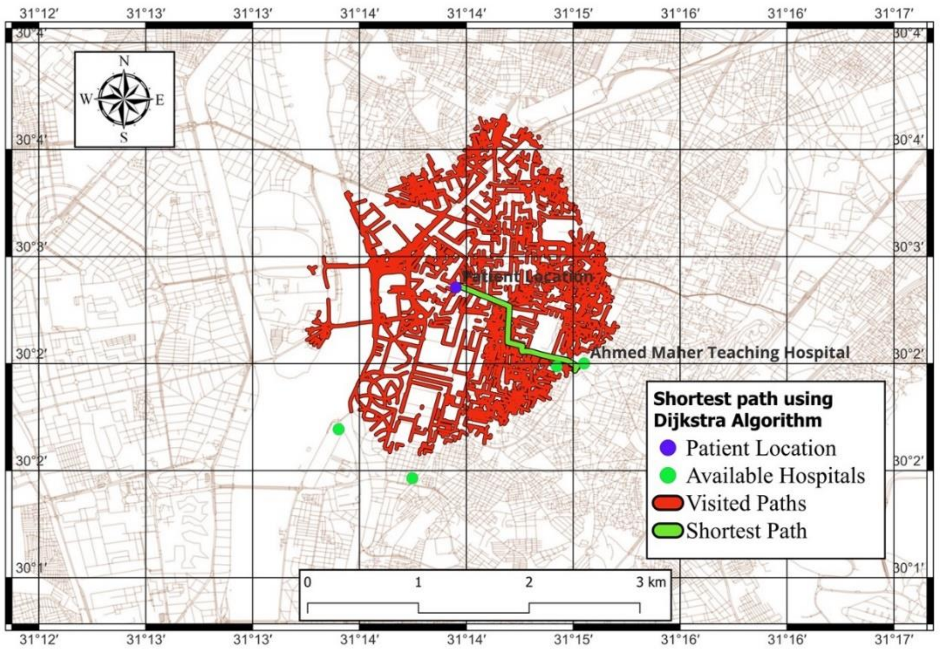


*Figure 12 Shortest path to Ahmed Maher Teaching Hospital using Dijkstra Algorithm*

For “*Ahmed Maher Teaching Hospital*” Dijkstra Algorithm pass with 5923 vertices to extract shortest path from patient location to this hospital as shown in (Figure 12)

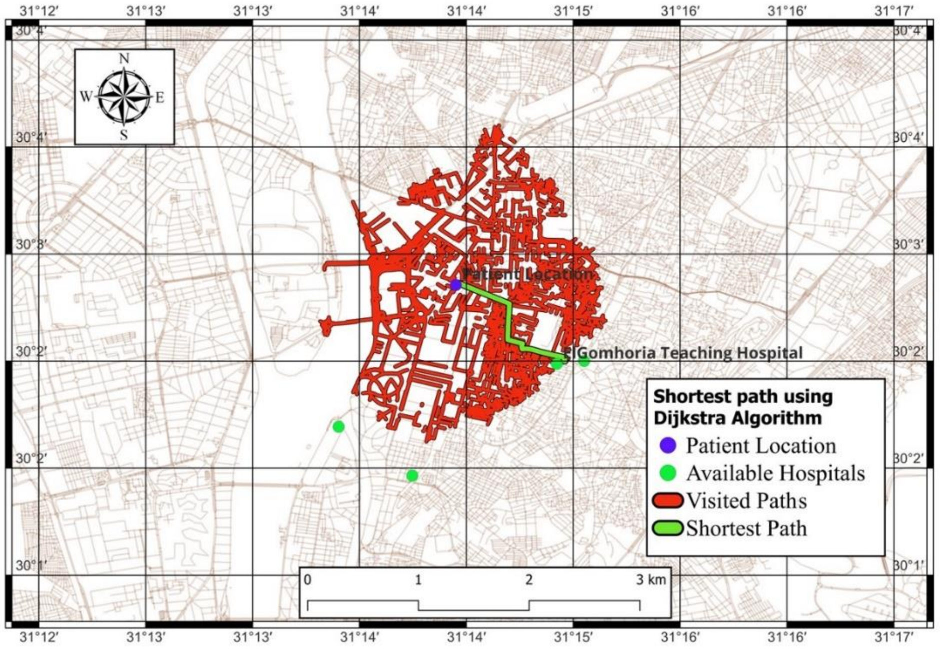


*Figure 13 Shortest path to ElGomhoria Teaching Hospital using Dijkstra Algorithm*

For extracting the shortest path from patient location to “*ElGomhoria Teaching Hospital*” there are 4702 vertices are examined by Dijkstra Algorithm as shown in (Figure 13)

The last available hospital within 2 kilometer “*Abo El Reesh Pediatric Hospital*” demands 8791 visited vertices to extract the shortest path from patient location to this as shown in (Figure 14).

As the previous discussion the total number of vertices need to checked using the existing Dijkstra algorithm is $32218 = 12802 + 5923 + 4702 + 8791$ for 2 kilometers buffer zone.

$$Eximined\ vertices = \sum_{i=1}^{n} vn_i \qquad equ.\ 4$$

Finally, we compare among the extracted shortest paths to extract the overall shortest path as shown in (Figure 15).

As shown in (Figure 20), the nearest hospital is “*ElGomhoria Teaching Hospital*” with path length ***1.55 kilometers***, and extraction time ***161 milliseconds***.

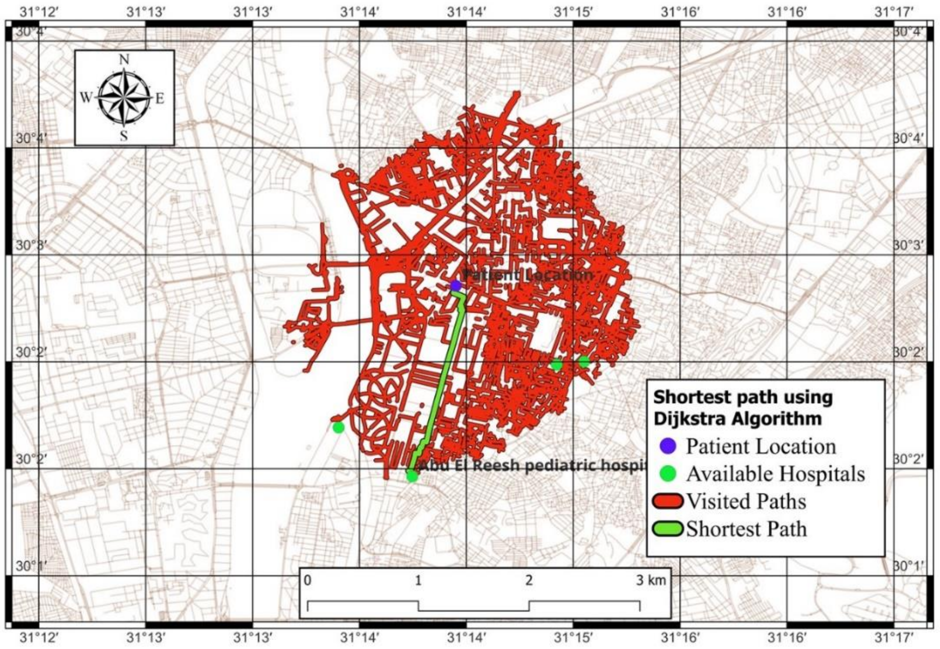


*Figure 14 Shortest path to Abo El Reesh Pediatric Hospital using Dijkstra Algorithm*

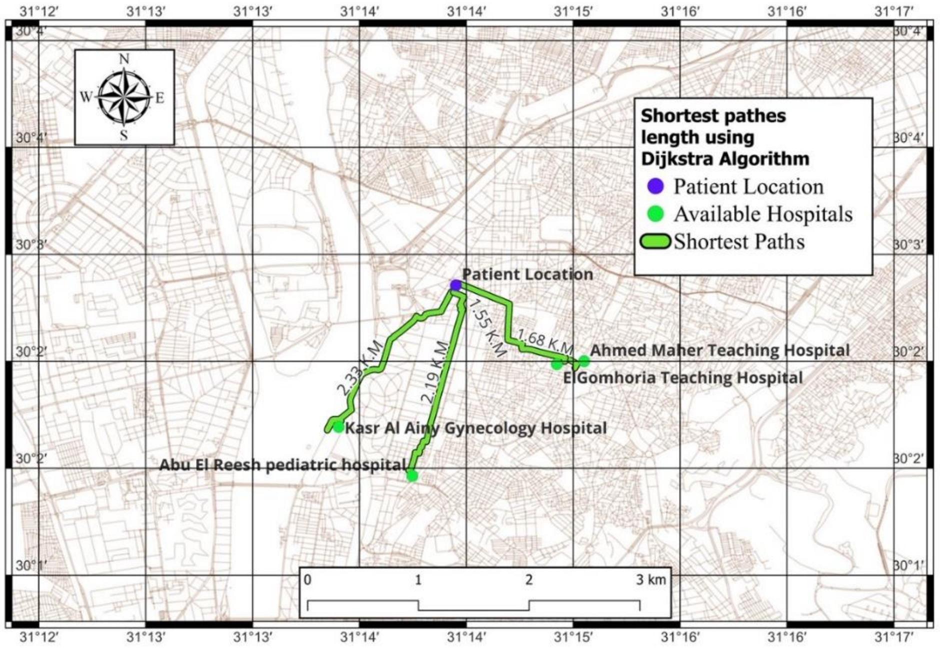

Shortest pathes length using Dijkstra Algorithm
Patient Location
Available Hospitals
Shortest Paths
1.55 K.M
1.68 K.M
2.33 K.M
2.19 K.M
Ahmed Maher Teaching Hospital
ElGomhoria Teaching Hospital
Kasr Al Ainy Gynecology Hospital
Abu El Reesh pediatric hospital
0 1 2 3 km

*Figure 15 Shortest Paths within 2 kilometers buffer using Dijkstra Algorithm*

### 2.3.3.2. Enhanced Dijkstra Algorithm (EDSPA) for 2 km buffer zone nearest hospitals

The proposed EDSPA algorithm extracts the overall shortest path directly, that is because the proposed algorithm works on multiple destinations node. The proposed algorithm demands to visit only 4702 vertices as shown in (Figure 16) to extract the shortest path for the nearest hospital.

As shown in (Figure 20), the nearest hospital is "*ElGomhoria Teaching Hospital*" with path length ***1.55 kilometers***, and extraction time ***26 milliseconds***.

According to the previous sub-sections, although the existed and proposed algorithms get the same results, the proposed algorithm execution time is dramatically shorter than the existing algorithm, proposed algorithm saves ***86%*** of execution time.

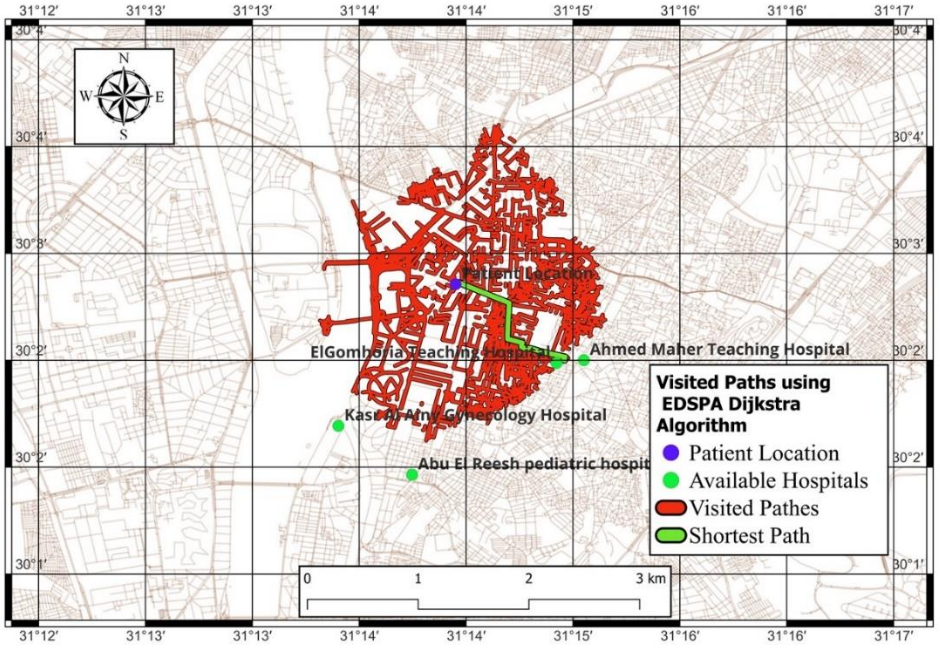

Patient Location
ElGomhoria Teaching Hospital
Ahmed Maher Teaching Hospital
Kasr Al-Ainy Gynecology Hospital
Abu El Reesh pediatric hospit
Visited Paths using EDSPA Dijkstra Algorithm
● Patient Location
● Available Hospitals
Visited Pathes
Shortest Path
0 1 2 3 km
N S E W

*Figure 16 Visited paths within 2 kilometers buffer using EDSPA Dijkstra Algorithm*

For study how the scalability of buffer zone and available hospitals effects on shortest path extraction time, we increase buffer zone to 3 kilometers with 15 available hospitals.

#### 2.3.3.3. Existing Dijkstra Algorithm for 3 km buffer zone nearest hospitals

For extracting all shortest paths from patient location to the nearest hospital from 15 available hospitals located within 3 kilometers buffer zone there are 310964 vertices examined by Dijkstra Algorithm as shown in (Figure 17).

Then a comparison among these paths is applied to extract the overall shortest path as shown in (Figure 18).

As shown in (Figure 20), the nearest hospital is "*ElGomhoria Teaching Hospital*" with path length ***1.55 kilometers***, and extraction time ***1726 milliseconds (1.726 second)***.

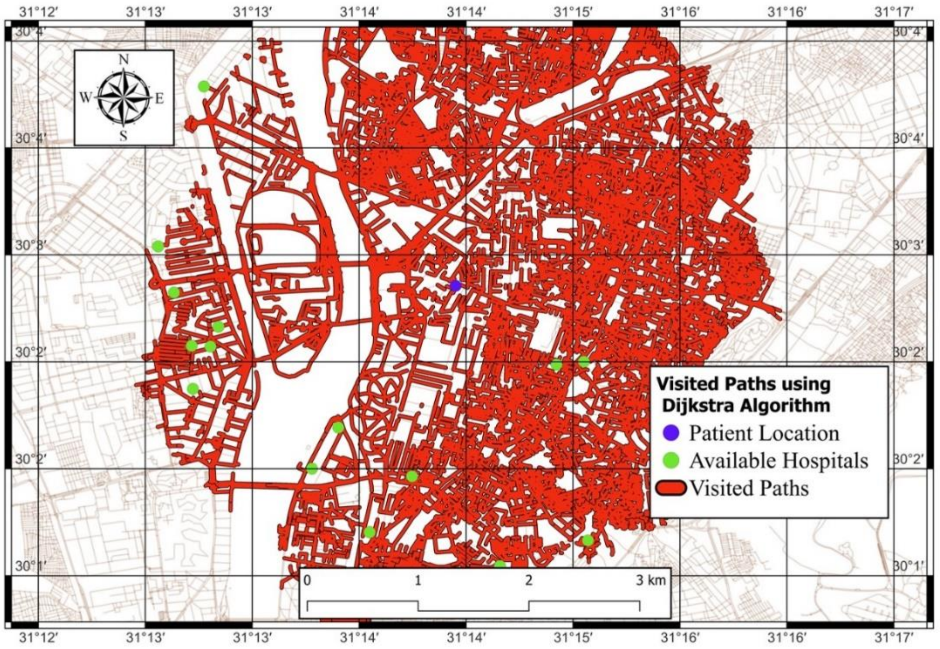


*Figure 17 Visited paths within 3 kilometers buffer using Dijkstra Algorithm*

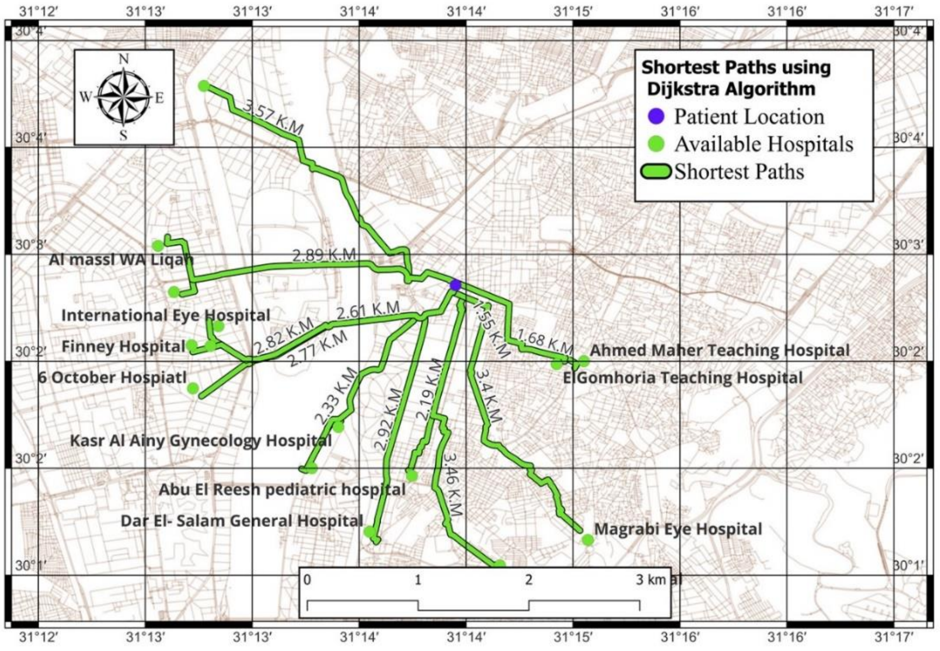
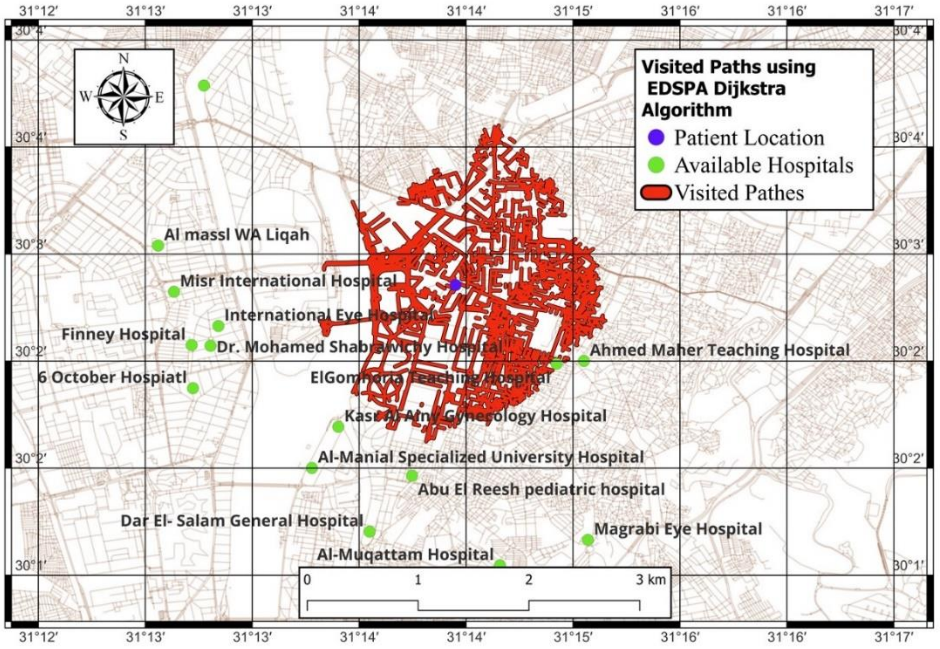


*Figure 18 Shortest paths within 3 kilometers buffer using Dijkstra Algorithm*

#### 2.3.3.4. Enhanced Dijkstra Algorithm (EDSPA) for 3 km buffer zone nearest hospitals

For extracting the overall shortest path from patient location to the nearest hospital from 15 available hospitals located within 3 kilometers buffer zone there are 4702 vertices examined by the proposed EDSPA algorithm as shown in (Figure 19).

As shown in (Figure 20), the nearest hospital is “*ElGomhoria Teaching Hospital*” with path length ***1.55 kilometers***, and extraction time ***31 milliseconds***.



Visited Paths using EDSPA Dijkstra Algorithm
Patient Location
Available Hospitals
Visited Pathes
Al massl WA Liqah
Misr International Hospital
International Eye Hospital
Finney Hospital
Dr. Mohamed Shabrawichy Hospital
6 October Hospiatl
ElGomhoria Teaching Hospital
Ahmed Maher Teaching Hospital
Kasr Al Ainy Gynecology Hospital
Al-Manial Specialized University Hospital
Abu El Reesh pediatric hospital
Dar El- Salam General Hospital
Magrabi Eye Hospital
Al-Muqattam Hospital
0 1 2 3 km
N S E W

*Figure 19 Visited paths within 3 kilometers buffer using EDSPA Dijkstra Algorithm*

## 3. Results and Discussion

According to the previous two sub-sections, although the existed and proposed algorithms get the same results, the proposed algorithm execution time is dramatically shorter than the existing algorithm, proposed algorithm saves 98% of execution time.

Also, by comparing results of applying the same algorithm with different number of available hospitals, Dijkstra algorithm requires multiple execution time, approximately 1500 milliseconds, for extra available hospitals. While the proposed algorithm requires slightly more time, 5 milliseconds, for extra available hospitals.

According to the above discussion and as comparison in (Table 1), both existed and enhanced algorithms extracted the same shortest path for the same nearest hospital. Although, the existing algorithm requires longer time than enhanced algorithm, especially when the number of available hospitals is increased. This agrees with the complexity of each algorithm. For the enhanced algorithm, the number of vertices visited to extract the shortest path is the same even though the number of available hospitals is increased. On the other hand, applying the existing algorithm requires to increase the number of visited vertices according to the number of available hospitals.

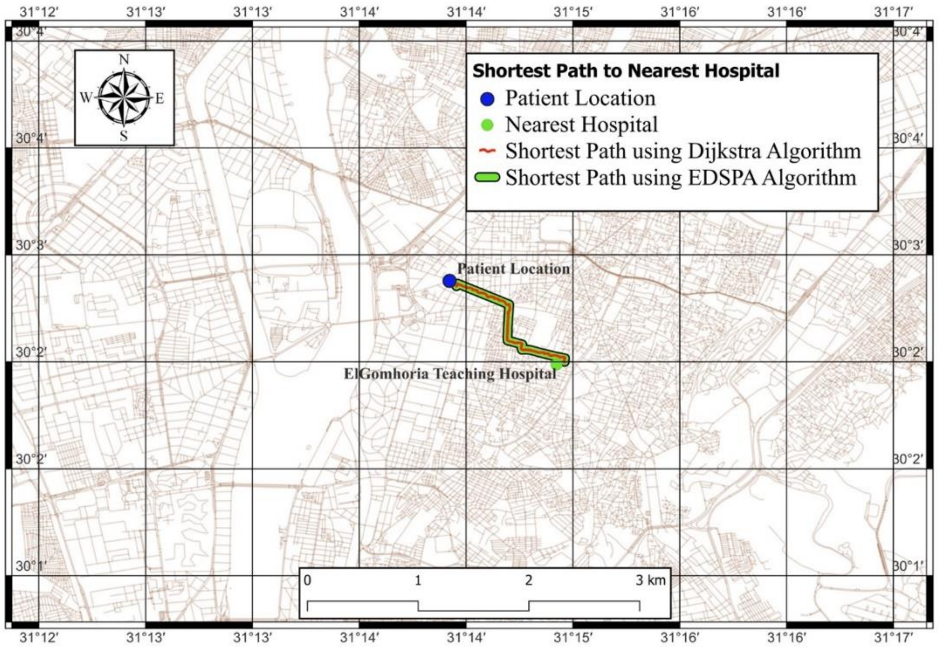

Shortest Path to Nearest Hospital
● Patient Location
● Nearest Hospital
Shortest Path using Dijkstra Algorithm
Shortest Path using EDSPA Algorithm
Patient Location
ElGomhoria Teaching Hospital
0
1
2
3 km

*Figure 20 Shortest path to Nearest Hospital*

*Table 1 Comparison between existed and enhanced Dijkstra Algorithms*

| | **Existing Dijkstra Algorithm** | | **Enhanced Dijkstra Algorithm** | |
|---|---|---|---|---|
| | **2 km. zone** | **3 km. zone** | **2 km. zone** | **3 km. zone** |
| **Complexity** | $O(\mathrm{N}(E\ log\ V))$ | | $O(E\ log(NV))$ | |
| **Available Hospitals** | 4 | 15 | 4 | 15 |
| **Visited vertices** | 32218 | 310964 | 4702 | 4702 |
| **Execution Time** | 161 milliseconds | 1725 milliseconds | 26 milliseconds | 31 milliseconds |
| **Shortest Distance** | 1.55 kilometers | | | |
| **Nearest hospital** | ElGomhoria Teaching Hospital | | | |

Another patient location is reidentified with coordinate 30.04711 latitude and 31.23902 longitude.

*Table 2 Comparison between existed and enhanced Dijkstra Algorithms with another patient location*

| | **Existing Dijkstra Algorithm** | | **Enhanced Dijkstra Algorithm** | |
|---|---|---|---|---|
| | **2 km. zone** | **3 km. zone** | **2 km. zone** | **3 km. zone** |
| **Complexity** | $O(\mathrm{N}(E\ log\ V))$ | | $O(E\ log(NV))$ | |
| **Available Hospitals** | 8 | 16 | 8 | 16 |
| **Visited vertices** | 50915 | 180173 | 849 | 849 |
| **Execution Time** | 282 milliseconds | 980 milliseconds | 5 milliseconds | 6 milliseconds |
| **Shortest Distance** | 0. 679 kilometers | | | |
| **Nearest hospital** | Maahad El Qalb | | | |

According to the comparison in (Table 2), which represents implementation results of execution both existed and enhanced algorithms, the extracted shortest path for the nearest hospital is same. Comparing Table 1 and Table 2 provides that enhanced algorithm execution time depends mainly on distance of the overall shortest path, while existing algorithm depends on both distance of the overall shortest path and the number of available hospitals.

In relation to the previous research, this study fills the gap found in previous studies by extracting the shortest path between one source location and the nearest destination from a list of locations. The enhanced Dijkstra algorithm is provided in this study to be capable of extracting the shortest path for nearest location without extracting paths for other available locations. The previous research provided improvements for Dijkstra algorithm within two models, the shortest path between one source and one destination locations or all other locations.

(Gabriel, Lolade, Durodola, & Orimoloye, 2019) which is the most study common to our study, used Dijkstra algorithm to extract the shortest path for all nearest hospitals to the accident location then used Fuzzy logic to recommend suitable hospitals out of list of nearest hospitals [2]. Because of using different datasets, to compare the shortest path time extracting, the current study uses Dijkstra algorithm for extracting the shortest path for all nearest hospitals. Then the enhanced algorithm is applied to extract the shortest path using the same patient location and available hospitals list. The compassion results in Table 1 and Table 2 show that the enhanced algorithm is significantly reduce the shortest path extracting time, and also extracting time for enhanced algorithm depends on the overall shortest path, and does not depends on the number of nearest hospitals. When there are **4** hospitals in the nearest hospital list the enhanced algorithm extracts the shortest path within **26** milliseconds by visiting **4702** intermediate vertices, while extracting the shortest path for all nearest hospitals requires **161** milliseconds and visiting **50915** intermediate vertices. By changing the patient location and with **16** hospitals in the nearest hospital list the enhanced algorithm extracts the shortest path within **6** milliseconds by visiting **849** intermediate vertices, while extracting the shortest path for all nearest hospitals requires **980** milliseconds and visiting **180173** intermediate vertices.

## 4. Conclusion

In conclusion, driving the shortest path to the nearest hospital plays a vital role in improving healthcare delivery, especially in pandemic and emergency situations where time is of the essence. By using advanced path-finding algorithms, such as Dijkstra, we can accurately calculate the most efficient route for a patient to reach a healthcare facility. In relation to the previous research, only two models of Dijkstra algorithm were applied in the previous studies, start to end nodes and start to all other nodes. Start to end nodes model is used to extract the shortest path between the start and the end node only. Although, starting node to all other nodes is used to extract all shortest paths between the start and each other nodes on road network. According to this research, extracting the overall shortest path for all available hospitals is required. Repeating starting to end node Dijkstra algorithm model is used to extract all shortest paths between the patient location and the available hospitals. Then comparing among these shortest paths is applied to extract the overall shortest path. Although this method is effective, it requires long execution time especially when increasing the number of available hospitals, ***1726 milliseconds*** for ***15 available hospitals.*** This study fills the gap found in previous studies by enhancing the Dijkstra algorithm to be able deal with single start and specific target nodes list. The proposed (EDSPA) model is focused on the target nodes list instead of single target node, which leads to dramatically decreasing the execution time. By using the proposed model, the same shortest path from the classic Dijkstra algorithm is extracted with saving more than 85% of execution time, ***31 milliseconds*** for ***15 available hospitals.*** In this study, we successfully decrease the extracting time of deriving the shortest path to the nearest hospital using enhanced Dijkstra algorithm. Saving this execution time enables us to identify the most optimal routes, ensuring that people can reach medical facilities as quickly as possible in the event of an emergency. The ability to decrease the time of extracting the shortest path not only improves access to healthcare, but also plays a critical role in optimizing emergency response systems, potentially saving lives by reducing travel times. For patients, especially those suffering from medical emergencies, rapid access to nearby hospitals can mean the difference between life and death. In conclusion, driving the shortest route to the nearest hospital represents not only a technical achievement in terms of algorithmic design, but also an important contribution to public health and safety. By continuously improving these systems, we can provide faster and more reliable access to emergency care, improving health outcomes, and potentially saving lives.

In future studies, the integration of real-time data (such as current traffic flow, road closures, and weather conditions) will improve accuracy and adaptability. This will enable users to receive up-to-date information about the best routes available, adjusting for unexpected delays or obstacles. Other factors, such as hospital availability, capacity and specialization, ensuring that patients are not only directed to the nearest hospital, but also to the one that best complements their specific needs, can be included in future studies. Future improvements may include machine learning-based models that predict traffic conditions and hospital availability, as well as advanced geographic information systems (GIS) that include factors such as road quality, accident hotspots, or weather conditions. In addition, the integration of multimodal transportation options (such as public transportation or air ambulances) can also optimize the routing system in urban and rural areas.

## References

[1] M. Yazdani and M. Haghani, "Optimisation-based integrated decision model for ambulance routing in response to pandemic outbreaks," *Progress in Disaster Science*, vol. 18, 2023.

[2] T. Gabriel, B. Lolade, Durodola and S. M. Orimoloye, "Shortest Route Analysis for Road Accident Emergency using Dijkstra Algorithm and Fuzzy Logic," *International Journal of Computer Science and Mobile Computing,* vol. 8, no. 12, pp. 64-73, 2019.

[3] R. Bellman, "ON A ROUTING PROBLEM," *Quarterly of Applied Mathematics,* vol. 16, pp. 87-90, 1958.

[4] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics,* vol. 4, no. 2, pp. 100-107, 1968.

[5] R. W. Floyd, "Algorithm 97: Shortest path," *Commun. ACM,* vol. 5, 1962.

[6] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik,* vol. 1, no. 1, pp. 269 - 271, 1959.

[7] D. Fan and P. Shi, "Improvement of Dijkstra's algorithm and its application in route planning," in *Seventh International Conference on Fuzzy Systems and Knowledge Discovery*, 2010.

[8] Jason, M. Siever, A. Valentino, K. M. Suryaningrum and R. Yunanda, "Dijkstra's algorithm to find the nearest vaccine location," *Procedia Computer Science*, vol. 216, pp. 5-12, 2023.

[9] Z. Tang, S. Zhou, B. Geng and J. Sun, "Research on Railway Emergency Rescue Path Selection Based on GIS," in *International Conference on Oil & Gas Engineering and Geological Science*, 2019.

[10] M. Sushma and V. Reddy, "Finding an optimal path with hospital information system using GIS-based Network analysis," *WSEAS Transations on Information Science and Applications,* vol. 18, 2021.

[11] M. Barbehenn, "A note on the complexity of Dijkstra's algorithm for graphs with weighted vertices," *IEEE Transactions on Computers,* vol. 47, no. 2, 1998.

[12] R. Nasiboglu, "Dijkstra solution algorithm considering fuzzy accessibility degree for patch optimization problem," *Applied Soft Computing,* vol. 130, 2022.

[13] Y. Huang, Q. Yi and M. Shi, "An Improved Dijkstra Shortest Path Algorithm," in *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering*, China, 2013.

[14] C. Chadha and S. Garg, "Shortest Path Analysis on Geospatial Data Using PgRouting," in *International Conference on Innovative Computing and Communications,* Singapore, 2019.

[15] D. Das, A. K. Ojha, H. Kramsapi, P. P. Baruah and M. K. Dutta, "Road network analysis of Guwahati city using GIS," *SN Applied Sciences,* vol. 1, no. 8, 2019.

[16] K. Anam and M. Yunus, "Android GIS-Based Information System Applying Dijkstra Algorithm For Finding The Nearest Tourist Spots in Sumenep District," *International Journal of Computer, Network Security and Information System,* vol. 1, no. 1, pp. 1-5, 2019.

[17] A. Y. Abd Al-Munaf, A. A. Abdulahmeed and K. Q. Hussein, "Accessing the Best Path Using Dijkstra Algorithm & GIS for Mobile Cloud Systems," in *3rd International Conference on Engineering Technology and its Applications,* 2020.

[18] N. Cadieux, M. Kalacska, O. T. Coomes, M. Tanaka and Y. Takasaki, "A Python Algorithm for Shortest-Path River Network Distance Calculations Considering River Flow Direction," *Data,* vol. 5, no. 1, 2020.

[19] A. B. Al Bager A. and A. Al Samani A., "Using GIS to Determine the Shortest Distance in the Searching Engines," *International Journal of Advance Research in Computer Science and Management Studies,* vol. 8, no. 9, pp. 1-7, 2020.

[20] Y. Xu, G. Guan, Q. Song, C. Jiang and L. Wang, "Heuristic and random search algorithm in optimization of route planning for Robot's geomagnetic navigation," *Computer Communications,* vol. 154, pp. 12-17, 2020.

[21] H. Bhardwaj, "Integration of A Vehicle Route Problem with A Geographical Information System," *International Research Journal of Modernization in Engineering Technology and Science,* vol. 3, no. 12.

[22] M. Sahu, P. Sharma, H. K. Sharma, T. Choudhury and B. K. Dewangan, "Route Optimization for Waste Collection," in *Emerging Technologies in Data Mining and Information Security*, Singapore, Springer Nature Singapore, 2023, pp. 605-613.

[23] A. Z. Tiong, C. J. Panganiban, M. C. Blanco, R. Regala and D. M. Cortez, "Enhancement of Dijkstra Algorithm for Finding Optimal Path," vol. 102, no. 1, pp. 164-170, 2022.

[24] M. S. N. Fitri, O. Marena, O. A. Hisam, M. Y. M. Hafiz and A. K. N. Izzati, "Suitability of Open Street Map (OSM) for 1:50000 Topographic Map," in *8th International Conference on Geomatics and Geospatial Technology*, 2022.

[25] "Spatial data for your project," 1 January 2021. [Online]. Available: https://nextgis.com/datasets/.

[26] A. J. Mahariba, R. A. Uthra and a. R. G. Brunet, "Estimation of Shortest Route with Minimum Travel Time Using GIS and MSSTT Algorithm," in *Lecture Notes in Civil Engineering,* vol. 191, Springer, Singapore, 2022.