Dijkstra’s Algorithm to Find Shortest Path of Tourist Destination in Bali

A Fitriansyah\textsuperscript{1a}, N W Parwati\textsuperscript{1b}, D R Wardhani\textsuperscript{1c} and N Kustian\textsuperscript{1d}

\textsuperscript{1}Program Studi Informatika, Universitas Indraprasta PGRI, Jl. Raya Tengah No.80, Kel. Gedong, Kec. Pasar Rebo, Jakarta Timur, 13760, Indonesia

\textsuperscript{a}aswin.fitriansyah@gmail.com; \textsuperscript{b}wayan.parwati@gmail.com; \textsuperscript{c}lilo4cherry@gmail.com; \textsuperscript{d}kustiannunu@gmail.com

Abstract. Bali is one of many small island in Indonesia and referred as “The Island of Gods”. Bali has varieties of tourist destination e.g. art villages, ecotourism, historical building, spiritual tourism and temples. Dijkstra’s algorithm is an algorithm that used to be solution in finding shortest path problem. It can be use to find the shortest route between one tourist destination and all other tourist destinations. Hence, we use dijkstra’s algorithm to find shortest path of tourist destination in Bali. There are top ten tourist destinations that has been used in this paper Tanah Lot, Uluwatu, Ulun Danu Beratan, Kebun Raya Eka Karya, Penelokan Batur, Tirta Empul, Taman Ayun, Bali Safari and Marine Park, Goa Gajah and Bali Zoo Park.

1. Introduction

Indonesia is country that has a lot of attractions. One of best attractions that never deserted local and international tourist is Bali. Bali is one of many small island in Indonesia and referred as “The Island of Gods”. These island has varieties of tourist destinations e.g. art villages, ecotourism, historical building, spiritual tourism and temple. Trip Advisor as the largest travel site in the world has crowned Bali as the most traveler’s choice destination in the world. In relatively small area of 5,636.66 km\textsuperscript{2}, Bali provides many kinds of attraction compared to other island. Hence it’s difficult for tourist to visit all of the attraction efficiently, therefor Dijkstra algorithm is used to determine shortest path to visit ten most visited attraction in Bali. As one of the main tourist destinations, there are domestic or international tourist comes to visit Bali. For the local people that has been living in Bali for years may common or know in detail about the location of tourist attraction in Bali. There are top ten of tourist destinations according Bali Government Tourism Office, as shows on figure 1.
Figure 1. Number of visitor in 2015 (www.disparda.baliprov.go.id/)

Zhan and Noon consider amount of empirical findings with respect to performance and found number of different algorithm. The computation of shortest path still an important task in many network and transportation related analysis [1]. Purwanto, Purwitasari and wibowo used Dijkstra, Floyd and two queues algorithm to find shortest path problem in surabaya and the calculation are implemented into web based and WAP based system. This research shows that Dijkstra algorithm has fastest execution time than Floyd and two Queues algorithm [2]. Afrianto found that shortest route on public transportation can be determined by Dijkstra algorithm. Calculation that is chosen from several alternative manually has the same result as Dijkstra algorithm.

Dijkstra algorithm was conceived by computer scientist Edsger Dijkstra in 1956 is an algorithm to find the shortest paths between nodes in graph [3]. Dijkstra algorithm is an iterative algorithm that find the shortest path from source vertex to all other vertices in graph and it is greedy algorithm that rely on optimum solution at each iteration until destination vertex visited [4]. This algorithm applied on some applications e.g. Computer Networking, Map and Other Navigation Services. Ahmat studied extensively in association with the complex communication networks. The study described basic concepts of graph theory and their relation to communication networks. The study also presented some optimization problems that are related to routing protocols and network monitoring and showed that many of the optimization problems are NP-Complete or NP-Hard. Finally, it described some of the common tools used to generate network topologies based on graph theory [5]. Meghanathan [6] reviewed Dijkstra’s algorithm and Bellman-Ford algorithm for finding the shortest path in a graph. He concluded that the time complexity of Dijkstra’s algorithm is $O(|E|\log |V|)$ while the time complexity of the Bellman-Ford algorithm is $O(|V||E|)$. Sommer [7] investigated shortest path query processing in networks both from a theoretical and a practical point of view. An experimental study was performed using road transportation network. The study revealed a simple and general method based on Voronoi duals to efficiently support the shortest path queries in undirected graphs with very low pre-processing overheads and competitive query times, at the cost of exactness. This method was proved to be effective on a variety of graph types while remaining a reasonable alternative to existing exact methods specifically designed for transportation networks. Li et al [8] proposed an efficient algorithm named Li-Qi (LQ) for the SSSP problem with the objective of finding a simple path of the smallest total weights from a specific initial or source vertex to every other vertex within the graph. This algorithm is formed from the ideas of the queue and relaxation; the vertices may be queued several times, and furthermore, only the source vertex and relaxed vertices are being queued.
2. Method
This research performed a quantitative approach to find shortest path of tourist destination in Bali using Dijkstra algorithm. Dijkstra algorithm will be implemented in a system using java programming language. Netbeans is an integrated development environment (IDE) for java that allows applications to be developed from a set of modular software components called modules. It provides builders Graphic User Interface (GUI), a text or code editor, a compiler or interpreter and a debugger.

The developed system’s input is a directional graph the weighted or referred to an array of some of the sights (Weighted Directed Graph)/G and a Vertex (Summit)/S in G and V is the set of all Vertices (Nodes) in a Graph G. Each side of this graph is a pair of Vertices (U, V) to symbolize the relationship from vertex U to vertex V the set of all edges E. Weights of all the sides are calculated with the function:

\[ w: E \rightarrow [0, \infty) \]  

So \( W(U, V) \) is a non-negative distance from Vertex U to Vertex V.

The picture below shows layout of the top 10 tourist destinations on the island of Bali:

![Figure 2. Layout of 10 top tourist destinations on the island of Bali](image)

Proposed system is application that applied Dijkstra algorithm to find shortest path of tourist destination in Bali. Steps of Dijkstra algorithm are as follows:

Step 1. Make sure that there is no negative edges and initialize distance to source node (vertex) as zero and all other distance to infinity;

![Figure 3. Figure of step 1 of Dijkstra algorithm](image)
Step 2. Update all nodes adjacent to the current node;

![Figure 4](image1.png) **Figure 4.** Figure of step 2 of Dijkstra algorithm

Step 3. Choose the closest nodes (minimum distance) as the next current node.

![Figure 5](image2.png) **Figure 5.** Figure of step 3 of Dijkstra algorithm

Step 4. Repeat step 2 and step 3 until destination node visited

3. **Result and discussion**

3.1 **Dijkstra computation**

Consider the starting node is D and end node is B, the distance from D to D is 0 and distances to all other nodes from D are unknown, and it should be set to the highest value ∞ (infinity).

3.1.1 **1st iteration of Dijkstra computation**

| Nodes | A | B | C | D | E | F | G | H | I | J |
|-------|---|---|---|---|---|---|---|---|---|---|
| Shortest Distance from D | ∞ | ∞ | ∞ | 0 | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ |
| Previous Node | | | | | | | | | | |
3.1.2 2nd iteration of Dijkstra computation
Start visiting the unvisited nodes that adjacent node D and has smallest distance (weight). Since the current node is D and the unvisited adjacent node are C, G and I. Calculate the distance of each adjacent nodes from the start node. The distance from D to C is $0 + 3.4 = 3.4$, distance from D to G is $0 + 34.7 = 34.7$, distance from D to I is $0 + 45.1 = 45.1$. Next, update the shortest distance edges from D. Since the current node is D, update the previous node with this current node D and node D will never revisited. Set the shortest distance as next current node. In this case is C.

| Nodes | A | B | C | D | E | F | G | H | I | J |
|-------|---|---|---|---|---|---|---|---|---|---|
| Shortest Distance from D | $\infty$ | $\infty$ | 3.4 | 0 | $\infty$ | $\infty$ | 34.7 | $\infty$ | 45.1 | $\infty$ |
| Previous Node | D | D | D |

Table 3. 2nd iteration of Dijkstra computation

3.1.3 3rd iteration of Dijkstra computation
Now, current node is C. Examines nodes that adjacent to C there are E, F and I. For the current node, calculate the distance of each nodes from the start node D. The distance of E from D is the distance of D to C that already calculated plus the distance f from D to E that is 59.9. Hence the distance of E from D is $3.4 + 59.9 = 63.3$. The distance of F from D is distance from D to C plus distance from C to F, that is $3.4 + 59.3 = 62.7$. The distance from D to I is the one that already written on previous table. That is 45.1 or the distance from D to C plus distance from C to I that is $3.4 + 45.4 = 48.8$, so the shortest distance from D to I is 45.1.

| Nodes | A | B | C | D | E | F | G | H | I | J |
|-------|---|---|---|---|---|---|---|---|---|---|
| Shortest Distance from D | $\infty$ | $\infty$ | 3.4 | 0 | 63.3 | 62.7 | 34.7 | $\infty$ | 45.1 | $\infty$ |
| Previous Node | D | C | C | D |

Table 4. 3rd iteration of Dijkstra computation

3.1.4 4th iteration of Dijkstra computation
Repeated the above steps and updated the table until destination node is visited.

| Nodes | A | B | C | D | E | F | G | H | I | J |
|-------|---|---|---|---|---|---|---|---|---|---|
| Shortest Distance from D | 52.9 | $\infty$ | 3.4 | 0 | 63.3 | 62.7 | 34.7 | $\infty$ | 45.1 | 49.7 |
| Previous Node | G | D | C | C | D |

Table 5. 4th iteration of Dijkstra computation

3.1.5 5th iteration of Dijkstra computation

| Nodes | A | B | C | D | E | F | G | H | I | J |
|-------|---|---|---|---|---|---|---|---|---|---|
| Shortest Distance from D | 52.9 | $\infty$ | 3.4 | 0 | 63.3 | 62.7 | 34.7 | 58.1 | 45.1 | 49.7 |
| Previous Node | G | D | C | C | D | I | D | G |

Table 6. 5th iteration of Dijkstra computation

3.1.6 6th iteration of Dijkstra computation

| Nodes | A | B | C | D | E | F | G | H | I | J |
|-------|---|---|---|---|---|---|---|---|---|---|
| Shortest Distance from D | 52.9 | 93.6 | 3.4 | 0 | 63.3 | 62.7 | 34.7 | 58.1 | 45.1 | 49.7 |
| Previous Node | G | J | D | C | C | D | I | D | G |

Table 7. 6th iteration of Dijkstra computation

In this iteration node B is visited, hence the iteration end. And the path is D-G-J-B with 93.6km distance.
3.2 System design
Java language builds up CRUD like every other language. Java has packages that help you build large-scale software use exceptions to make code more reliable and easier to maintain manage memory automatically with garbage collection discover how the JVM provides portability and security [9, 10]. In this paper we propose system using Dijkstra algorithm to find shortest path of tourist destination in Bali. The proposed system shows graph and chosen path from origin node to destination node.

```
public class Dijkstra {
    private Random random = new Random();
    public static String String(String s) {
        return s;
    }
}
```

![Figure 6. Codes of Dijkstra algorithm](image)

In this paper we propose system using Dijkstra algorithm to find shortest path of tourist destination in Bali. The proposed system shows graph and chosen path from origin node to destination node.

```
while (!queue.isEmpty()) {
    Vertex u = queue.poll();
    // Put each un visited vertex
    for (Vertex v : u.getNeighbors()) {
        if (v.getDistance() > u.getDistance() + weight) {
            v.setDistance(u.getDistance() + weight);
            queue.add(v);
        }
        v.setPrevious(u);
        queue.offer(v);
    }
}
```

![Figure 7. User interface of shortest path system](image)

In figure 7. Shows that origin node is A (Tanahlot) and final node or destination node is F (Tirta empul) and it resulted A-G-I-F as a shortest path. In this application user (tourist/traveler) may input the desires origin node and destination node from ten most visited attraction in Bali.

4. Conclusion
This paper proposed Dijkstra algorithm to find shortest path of ten tourist destination in Bali; and it successfully implemented using java programming language. Proposed system presents graph and shortest path from the origin node to destination node. In future works, this algorithm might be implemented in other platform (android) with consideration of other factor (e.g. traffic and other infrastructure problems), and might provides all tourist destinations in Bali.
Acknowledgments
The authors would like to thank Kemenristek DIKTI Kopertis Wilayah III for financial support of this research.

References
[1] Zhan F B and Noon C E 1998 Shortest Path Algorithms: An Evaluation Using Real Road Networks Transp. Sci. 32 1 pp 65–73
[2] Purwananto Y, Purwitasari D and Wibowo W A 2005 Implementasi dan analisis algoritma pencarian rute terpendek di Kota Surabaya Jurnal Penelitian dan Pengembangan TELEKOMUNIKASI 10 2 pp 94-101
[3] Misa T J 2010 An interview with Edsger W. Dijkstra Commun. ACM 53 8 p 41
[4] Chen J 2003 Dijkstra’s shortest path algorithm J. Formaliz. Math. 15 2 pp 1–9
[5] Ahmat K A 2009 Graph Theory and Optimization Problems for Very Large Networks pp 1–5
[6] Meghanathan N 2015 Use of eigenvector centrality to detect graph isomorphism Computer Science & Information Technology (CS & IT) pp 01–09
[7] Sommer C 2016 All-pairs approximate shortest paths and distance oracle preprocessing 43rd Int. Colloq. Autom. Lang. Program. 55 pp 1-13
[8] Li T, Qi L and Ruan D 2008 An efficient algorithm for the Single-Source Shortest Path Problem in graph theory Proc. of 2008 3rd International Conference on Intelligent System and Knowledge Engineering pp 152–157
[9] Waldo J 2010 Java: The good parts Style DeKalb IL First Edition pp 192
[10] Hayes B 1997 The Java programming language/the Java tutorial/the Java language specification/the Java virtual machine specification/and others Am. Sci. 85 4 pp 389–390