Shortest path applied on tourist destination in Bali: a short review

Ni Wayan Parwati Septiani*, Dyah Rhetno Wardhani, Nunu Kustian and Aswin Fitriansyah
Universitas Indraprasta PGRI

*ni_wayanparwati@unindra.ac.id

Abstract. Bali is a tourist destination known as the Island of Gods. Largest trusted site TripAdvisor has crowned Bali as “The World Best Destination” chosen by International and local tourist beat other larger country in the world. Variety of tourist destination in Bali has confused tourist from any other country or any other city to find an efficient route to excurses all Bali’s tourist destinations. Ants Colony algorithm and Dijkstra algorithm are two algorithms that used to solve an optimizations problem such as finding the shortest path to have an efficient tour. Both algorithms will help travel agent, international or local tourist to have a convenience travel. This paper presents computation using ants colony algorithms and Dijkstra algorithms to find the shortest path over ten greatest tourist destination in Bali (Data from Bali tourism office). This paper shows that ant colony algorithm computation more complicated than Dijkstra, but it has a better result.

1. Introduction
Tourist destination is a place for tourist to visit and stay, could be a country, state, region and city, usually due to its cultural or natural value. Bali is ranked as the best world’s tourist destination to visit. According tourism office of Bali province, there are top ten best destination in Bali i.e. Tanah Lot, Uluwatu, Ulun Danu Beratan, Kebun Raya Eka Karya, Penelokan Batur, Tirta Empul, Taman Ayun, Bali Safari Park, Goa Gajah, Bali Zoo Park.

Ant algorithm is an heuristic algorithm that use to optimization issues. Ant algorithm is mimicking ant’s behaviour in finding their food. Ants use pheromone in finding their food from the food source to its nest [1], [2]. Pheromone is a chemical item that produces by ants that trigger behaviour of another ants. The more pheromone substances left on path, hence the other ants will follow the same path. On the other hand Dijkstra’ Algorithm was conceived by Edger W. Dijkstra in 1956 is an optimization algorithm for finding the shortest paths between nodes in a graph [3]–[5].

Therefore to have an efficient and effective travel experience we try to applied ant algorithm and Dijkstra ‘algorithm to find the shortest path of top ten tourist destination in Bali.

2. Methodology
Mount of research regarding shortest path problem was founded in many literatures. Several articles have been published that successfully overcome shortest path problem. Sun et al. use the advantage of Dijkstra algorithm and applied to tourism path search and the experimental shows that Dijkstra algorithm can accomplish a good result in efficiency [6]. Gallo and Pallottino considered the shortest path problem
from a computational point of view, focus of their study is on implementation of the different structures used in the algorithms [7]. Bazarra and Goode formulate dual of travelling salesman problem solved by subgradient optimization technique [8]. Bazarra’s formulation extends in natural way the dual problem of Held and Karp to the nonsymmetric case, a branch and bound scheme with stepped fathoming is then used to find optimal and suboptimal tours [8]. Zakzouk et al. was presented an ant colony optimization approach to solve the shortest path problem, especially with fuzzy constraints. The proposed algorithm are including following sequential steps as follows, first is to determine the number of possible path from the source to target, second is to calculates the probability of each possible paths, third is calculates the expected number of ants through each path of possible path then calculates the new trail of each weight component for each path of possible paths, that leads to the final step to calculate the average trail of each path [9]. Mac-Min Ant System (MMAS) is ant colony optimization algorithm derived from ant system [10]. MMAS differs from ant system in several important aspects, whose usefulness are demonstrated by means of an experimental study. The results on the traveling salesman problem and quadratic assignment problem shows that MMAS is currently among the best performing algorithm [10].

2.1. Ant Colony Algorithm
Ant colony optimization (ACO) takes inspiration from the behaviour of ant species. Ants left their chemical product called pheromone on their path in order to mark some favourable path that should be followed by other ants [11]. ACO use similar mechanism to solve optimization problems. The application of ant colony algorithm includes traveling salesman problem, quadratic assignment, job-shop scheduling, vehicle routing, graph colouring, shortest common super sequence. Steps of ant colony algorithm for traveling salesman problem are as the following case.

Given \( n \) cities, aiming to find the shortest path going through all cities and visiting each exactly once. There are some definition that use on ant algorithm computation i.e. \( d_{ij} \) is euclidean distance from city \( i \) to city \( j \); \( m \) is the number of ants; \( \tau_{ij} (t) \) is the intensity of pheromone; \( \eta_{ij} \) is visibility (heuristic information) expressed by \( 1/d_{ij} \); \( (1-\rho) \) evaporation factor, \( \rho \) is constant for the whole optimization process; tabu \( k \) is dynamically growing vector of cities that have already been visited by \( k \)th ant; Ant System iteration each ant adds one city to the built route; Ant System Cycle composed of \( n \) iteration during which all ants complete their routes.

**Pheromone deposition of ant system**

\[
\tau_{ij} = \rho \cdot \tau_{ij}(t) + \Delta \tau_{ij}
\]

\[
\Delta \tau_{ij} = \sum_k \Delta \tau_{ij}^k
\]

\[
\Delta \tau_{ij}^k = \begin{cases} 
Q \cdot (1-k^\text{th ant used the edge (1-}\) & \\
0, & \text{otherwise}
\end{cases}
\]

Where,

- \( \Delta \tau_{ij}^k \) is the amount of pheromone deposited on the edge \((i,j)\) by \( k \)th ant within a time interval \((t,t+n)\)
- \( Q \) is a constant
- \( L_k \) is the length of route constructed by \( k \)th ant
- \( \rho \) must be smaller than 1, otherwise pheromone would accumulate unboundedly (recommended is 0.5)
- \( \tau_{ij}(0) \) is set to small positive values

**Ant system probabilistic decision making**

Probability of adding a link \( i-j \) (where \( j \in (N - \text{tabu}_k) \)) into the route

\[
P_{ij}^k(t) = \begin{cases} 
\frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_l[\tau_{il}(t)]^\alpha [\eta_{ij}]^\beta}, & \text{if } j \in (N-\text{tabu}_k) \\
0, & \text{otherwise}
\end{cases}
\]

where,

- \( l \in (N - \text{tabu}_k) \)
- \( \alpha, \beta \) define relative importance of the pheromone and the visibility.

The probability is a compromise between visibility (that prefer closer cities to more distant ones) and intensity of pheromone that prefers more frequently used edges. The ant system cycle are as follows:
a. Initialize time \( t=0 \); number of cycle \( NC=0 \); pheromone \( \tau_{ij}(t) = c \) and initial positioning of \( m \) ants to \( n \) cities;

b. Initialize tabu list;

c. Each ant iteratively builds its route; calculate length of the routes \( L_k \) for all \( k \in (1, \ldots, m) \);
   update the shortest route found and calculate \( \Delta \tau_{ij}^k \) and update \( \tau_{ij}(t+n) \)

d. Increment discrete time \( t = t + n \), \( NC = NC + 1 \)

e. If \( (NC < NC_{max}) \) then go to step 2, else stop.

2.2. Dijkstra Algorithm

Dijkstra’s algorithm can be applied to either directed or undirected graph, without negative edge. It is an algorithm for finding the shortest paths between nodes in a graph and it conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later \([3], [12]\). Process of Dijkstra algorithm is using greedy principles aims to find the shortest path by choosing minimum weighted path and use it to the solution set. Dijkstra’ algorithm solve optimization problem such shortest path, it finds shortest path from a given node \( S \) (starting node) to all other nodes in graph. Dijkstra’ algorithm starts by assigning initial value for the distance from node \( S \) and to every other node in the graph. It operates in some steps, for every steps the distance values is improved and the shortest distance from node \( S \) to another node is determined.

Dijkstra algorithm steps:

**Step 1.** Initialize the distance value to node \( S \) and labeled it as permanent \((0, p)\), assign to every node a distance value of \( \infty \) and labeled it as temporary \((\infty, t)\), and designated the node \( S \) as the current node.

**Step 2.** Update distance value and current node, let \( i \) become the index of the current node. Find the set \( J \) of nodes with temporary labels that can be reached from the current node \( i \) by a link \((i, j)\). update the distance values of those nodes. For each \( j \in J \), the distance value \( d_j \) of node \( j \) is updated as follows
\[
\text{new } d_j = \min \{d_i, d_j + c_{ij}\},
\]
where \( c_{ij} \) is the cost of link \((i, j)\), as given in the graph/network problem. Determine a node \( j \) that has the smallest distance value \( d_j \) among all nodes \( j \in J \), find \( j^* \) such that
\[
\min_{j \in J} d_j = d_{j^*},
\]
and change the label of node \( j^* \) to permanent and designate this node as current node.

**Step 3.** If all node that can be reached from node \( S \) have been permanently labeled then stop the process, and if we cannot reach any temporary labeled node from the current node, then all the temporary labels become permanent and the process is done. Otherwise go to step 2.

3. Result and Discussion

There are ten most visited tourist destination according Bali Tourism Office. The following table (Table 1.) is distance of ten tourist destination.

**Table 1. Distance of ten most visited tourist destination in Bali**

|   | A   | B   | C   | D   | E   | F   | G   | H   | I   | J   |
|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| A | 0   | 42.5| 51.2| 50.7| 64.3| 47.2| 18.2| 41  | 33.3| 27.1|
| B | 42.5| 0   | 79  | 78.6| 87.4| 70.4| 46.2| 49.8| 57.5| 43.9|
| C | 51.2| 79  | 0   | 3.4 | 59.9| 59.3| 35.1| 59.5| 45.4| 47.9|
| D | 50.7| 78.6| 3.4 | 0   | 59.8| 57.5| 34.7| 59.2| 45.1| 47.6|
| E | 64.3| 87.4| 59.9| 59.8| 0   | 17.1| 48.4| 38.9| 31  | 41.9|
| F | 47.2| 70.4| 59.3| 57.5| 17.1| 0   | 31  | 26  | 14  | 25  |
| G | 18.2| 46.2| 35.1| 34.7| 48.4| 31  | 0   | 28  | 18  | 15  |
| H | 41  | 49.8| 59.5| 59.2| 38.9| 26  | 28  | 0   | 13  | 14  |
| I | 33.3| 57.5| 45.4| 45.1| 31  | 14  | 18  | 13  | 0   | 11  |
| J | 27.1| 43.9| 47.9| 47.6| 41.9| 25  | 15  | 14  | 11  | 0   |
Notation:
A : Tanah Lot  
B : Uluwatu  
C : Ulun Danu Beratan  
D : Kebun Raya Eka Karya  
E : Penelokan Batur  
F : Tirta Empul  
G : Taman Ayun  
H : Bali Safari Park  
I : Goa Gajah  
J : Bali Zoo Park

3.1. Implementation of Ant Algorithm
Following steps are process of finding shortest path using Ant Colony algorithm.

**Step 1.** Initialize the following parameter:
- \( \alpha = 1 \)
- \( \beta = 1 \)
- \( \rho = 0.5 \)
- \( M = 10 \)
- \( \tau_{ij} (awal) = 0.01 \)
- \( N = 10 \)

**Step 2.** Find nodes/tourist destination visibility \( \eta_{ij} = \frac{1}{d_{ij}} \), where is distance between \( i \) and \( j \).

**Step 3.** Find the next tourist destination by calculate the probability of ant \( i \) at the starting point choose the next \( j \) point. For example the probability of (A,B) is \( \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum[\tau_{ik}']^\alpha \cdot [\eta_{ik}']^\beta} = \frac{0.01^1 \cdot 0.002532941^1}{0.00245708} = 0.0957 \)

**Step 4.** iterate the above steps until all objects (tourist destination) are visited.

**Step 5.** calculate the length of the path of each ant \( L_k \).

| Table 2. Cumulative probability |
|-------------------------------|
| \( A \) | \( B \) | \( C \) | \( D \) | \( E \) | \( F \) | \( G \) | \( H \) | \( I \) | \( J \) |
|---|---|---|---|---|---|---|---|---|---|
| A | 0 | 0.09573 | 0.175194 | 0.255441 | 0.318715 | 0.404913 | 0.628459 | 0.727691 | 0.84987 | 1 |
| B | 0.150394 | 0.150394 | 0.231302 | 0.312621 | 0.385753 | 0.476545 | 0.614894 | 0.743242 | 0.854403 | 1 |
| C | 0.04359 | 0.071841 | 0.121933 | 0.197343 | 0.272879 | 0.370305 | 0.530362 | 0.639842 | 0.892194 | 1 |
| D | 0.043873 | 0.072172 | 0.121933 | 0.204797 | 0.27879 | 0.370305 | 0.530362 | 0.639842 | 0.892194 | 1 |
| E | 0.068283 | 0.114064 | 0.168414 | 0.224466 | 0.412943 | 0.51691 | 0.64087 | 0.871081 | 1 |
| F | 0.159375 | 0.222158 | 0.304797 | 0.388388 | 0.448318 | 0.541887 | 0.64548 | 0.806626 | 1 |
| G | 0.074724 | 0.136244 | 0.187735 | 0.239468 | 0.318244 | 0.436079 | 0.545496 | 0.545496 | 0.781165 | 1 |
| H | 0.071723 | 0.11326 | 0.165867 | 0.218824 | 0.295869 | 0.466467 | 0.599154 | 0.782875 | 1 |
| I | 0.093552 | 0.151303 | 0.204232 | 0.257494 | 0.318001 | 0.419412 | 0.58843 | 0.769521 | 1 |

To determine the next destination, examine parameter of comparison of exploitation to the exploration by triggering random number \( q \) and \( q_0 \).

**Step 4.** iterate the above steps until all objects (tourist destination) are visited.

**Step 5.** calculate the length of the path of each ant \( L_k \).

| Table 3. Route length of each ant cycle |
|---|---|---|---|---|---|---|---|---|
| S1 | A | C | I | D | B | G | J | H | F | E | 338.6 | 0.002953 |
| S2 | B | E | C | G | H | F | D | I | J | A | 377.1 | 0.002652 |
| S3 | C | A | D | H | G | I | F | E | J | B | 324 | 0.003086 |
| S4 | D | E | C | I | J | G | A | H | F | B | 346.7 | 0.002884 |
| S5 | E | J | F | H | I | G | A | C | D | B | 293.3 | 0.003409 |
| S6 | F | E | B | I | H | G | J | A | D | C | 299.2 | 0.003342 |
The above table 3. shows the best route/path is Panelokan Batur, Bali Zoo Park, Tirta Empul, Bali Safari and Marine Park, Goa Gajah, Taman Ayun, Tanah Lot, Ulun Danu Beratan, Kebun Raya, with 293.3 km distance.

3.2. Implementation of Dijkstra’ Algorithm

The following picture is a graph that represented ten most visited tourist destination in Bali.

Figure 1. Weighted directed graph of ten most visit tourist destination in Bali

The aim is to find the shortest path from node D (Kebun Raya Eka Karya) to node B uluwatu. The process are as the following steps:

**Step 1.** Node D is designated as the current node (0, p) and every other node is (∞, t)

**Step 2.** Update the distance values of nodes C, I and G since those three nodes are connected to D.

\[
\begin{align*}
    d_C &= \min\{\infty, 0 + 3.4\} = 3.4; \\
    d_I &= \min\{\infty, 0 + 45.1\} = 45.1; \\
    d_G &= \min\{\infty, 0 + 34.7\} = 34.7.
\end{align*}
\]

Thus among nodes C, I and G, Node C has the smallest distance value. Hence node C becomes permanent node and the other nodes remains temporary. Now node C is the current node. Iterate this step until all nodes are visited

![Graph of ten most visited tourist destination in Bali](image_url)

**Table 4.** Iteration of Dijkstra ‘process

| Visited | Current | A     | B     | C     | D     | E     | F     | G     | H     | I     | J     |
|---------|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| [1]     | -       | (∞,t) | (∞,t) | (∞,t) | (∞,t) | (∞,t) | (∞,t) | (∞,t) | (∞,t) | (∞,t) | (∞,t) |
| [D]     | D       | (∞,t) | (∞,t) | (3.4,p) | (∞,t) | (∞,t) | (34.7,t) | (∞,t) | (45.1,t) | (∞,t) | (∞,t) |
| [DC]    | C       | (∞,t) | (∞,t) | (∞,t) | (63.3,t) | (∞,t) | (62.7,t) | (∞,t) | (45.1,t) | (∞,t) | (∞,t) |
| [DCG]   | G       | (52.9,t) | (∞,t) | (63.3,t) | (62.7,t) | (∞,t) | (45.1,t) | (49.7,t) | (93.6,p) | (93.6,t) | (93.6,t) |
| [DCGI]  | I       | (52.9,t) | (∞,t) | (63.3,t) | (59.1,t) | (58.1,t) | (49.7,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) |
| [DCGIJ] | J       | (52.9,p) | (93.6,t) | (63.3,t) | (59.1,t) | (58.1,t) | (58.1,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) |
| [DCGJA] | A       | (93.6,t) | (63.3,t) | (59.1,t) | (58.1,t) | (58.1,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) |
| [DCGIAH] | H       | (93.6,t) | (63.3,t) | (59.1,t) | (58.1,t) | (58.1,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) |
| [DCGIAHF] | F       | (93.6,t) | (63.3,t) | (59.1,t) | (58.1,t) | (58.1,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) |
| [DCGIAHE] | E       | (93.6,t) | (63.3,t) | (59.1,t) | (58.1,t) | (58.1,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) | (93.6,t) |
Step 3. done the process and from Table 4. shows that the shortest path from node D to node B is 93.6 km and the path is Kebun Raya Eka Karya, Taman Ayun Bali Zoo Park and Uluwatu.

4. Conclusion
By using ant colony algorithm and Dijkstra algorithm can help tourist and travel agent in determining an efficient tour package. Ant algorithm and Dijkstra algorithm are applied on ten most visited tourist destination. Dijkstra algorithm has simpler step than ant algorithm. By using Dijkstra algorithm start node and destination node are required and not all nodes are visited, while by using ant algorithm resulted the shortest path and all nodes must be visited.

Acknowledgement
The authors would like to thank Kemenristek DIKTI Kopertis Wilayah III for financial support of this research.

References
[1] Salami N Al, 2009 Ant Colony Optimization Algorithm UbiCC J. 4, 3 p. 823–826.
[2] Dorigo M and Stützle T, 2004 Ant Colony Optimization.
[3] Dijkstra E W, 1959 A note on two problems in connexion with graphs Numer. Math. 1, 1 p. 269–271.
[4] Chen J, 2003 Dijkstra’s shortest path algorithm J. Formaliz. Math. 15, 2 p. 1–9.
[5] Bento L M S Boccardo D R Machado R C S Miyazawa F K Pereira de Sá V G and Szwarcfiter J L, 2017, Dijkstra graphs, Discrete Applied Mathematics.
[6] Sun Y Liu Y and Xiao K, 2012, Shortest travel path searching system based on dijkstra algorithmm, Applied Mechanics and Materials, 198–199. p. 1395–1398.
[7] Gallo G and Pallottino S, 1988 Shortest path algorithms Ann. Oper. Res. 13, 1 p. 1–79.
[8] Bazaraa M S and Goode J J, 1977 The traveling salesman problem: A duality approach Math. Program. 13, 1 p. 221–237.
[9] Zakzouk A A A Zaher H M and El-Deen R A Z, 2010 An ant colony optimization approach for solving shortest path problem with fuzzy constraints Informatics Syst. (INFOS), 2010 7th Int. Conf.
[10] Stützle T and Hoos H H, 2000 MAX–MIN Ant System Futur. Gener. Comput. Syst. 16 p. 889–914.
[11] Dorigo M and Blum C, 2005 Ant colony optimization theory, A survey Theor. Comput. Sci. 344, 5 p. 243–278.
[12] Misa T J, 2010 An interview with Edsger W. Dijkstra Commun. ACM 53, 8 p. 41.