Digital Zoo management

Md.Tanvir Hossain[†],Md.Shahbaz Sadik*, Nzmus Sakib Patwary[‡]

2015-2-60-113 , 2015-2-60-067 , 2015-2-60-092

[†*‡]Department of Computer Science and Engineering, East West University, Dhaka-1212, Bangladesh.

Email: [†]th4pallab@gmail.com , *sadiksojib@gmail.com,[‡] Nazmus.ewu@gmail.com

***Abstract*—In this project, we discussed the shortest path solution to all pair shortest problem based on Warshall algorithm as resolving the basic concepts. We can travel a destination point by different routes in zoo. This can be time consuming if we do not travel trough the best route. This project aims to determine path distance from the source node and calculating the shortest path on the whole zoo. This project includes the modification of main algorithm which has been implemented in the prototype development. This study discussed the emphasis on all pair shortest path at the location of specific studies. So, by this study we have proposed a practical algorithm for the shortest path problem to find the shortest route .The proposed algorithm is also compared with the others algorithm to prove its efficiency.**

Key-words : shortest path, Dijkstra's algorithm, Warshal algorithm ,0-1 knapsack algorithm etc.

1. Introduction[1]

In modern life , we trying to optimized the problem to solve in best way. Finding shortest path is big challenge in modern life .There have many algorithm to find out shortest path. The computation of a shortest path between different locations appears to be a key problem in the road networks. We usually explore every possible solution in finding an optimum path, but not every solution can produce the best shortest path. Every direction in a graph should have a cost to be calculated. This shortest path problem is a way to find a new route or path in a graph with a minimum sum of weight traveled through the direction. This shortest path problem can be solved by using an algorithm of finding the best edge path between vertices in a graph. There are several variations of algorithm that can be used to determine the node that followed the direction given graph. Variations of the shortest path can be distinguished from single-source objective, pair path and generalization. A pair of shortest path is finding the shortest path for two points of nodes. All pair of shortest path is a technique to find the shortest path among all directed nodes. Single-source shortest path is finding the shortest form traveled, starting from a certain node to all other nodes in the graph. When we try to find out largest area shortest path then we can be uses this kind of algorithm. So, We need this type of project because, due to the development of this algorithm it is possible to determine the fastest route in zoo and dispatch an emergency vehicle like ambulance, fire service etc.

2.

3. Background Study

A. Google searching **:** We have used Google search to study our project. We construct a road map in zoo with weighted and undirected graph . Then we have study about warshall algorithm to find out distance from one location to another location in zoo.

B. Floyd-Warshall Algorithm **:** Floyd–Warshall algorithm is an algorithm for finding all pair shortest paths in a weighted graph with positive or negative edge weights.

C. Dijkstra's Algorithm **:** Dijkstra's algorithm is an algorithm for finding a single source shortest path in a weighted graph with non-negative edges weights.

D. 0-1 Knapsack Algorithm : The knapsack problem or rucksack problem is a problem in combinatorial optimization: Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.

E. NP-Heard problem **:** At first we try to find Travelling Sell's man problem but it is a NP-heard problem**.** But it has not any polynomial solution ,so that didn't find any solution of that problem. At last we applying Floyd-warshall algorithm to find out shortest path in zoo.

F. C++ Language **:** Then we implement Floyd-Warshall algorithm to construct our project by using C++ Language.

4. Implemented work

Floyd-Warshall algorithm is an algorithm for finding the shortest paths between all pair of node which is most similar to Dijkstra's algorithm. It maintains the class of search algorithm. This algorithm can be implemented both directed and in- directed graph. The graph can be negative weighted. Floyd-Warshall algorithm uses the dynamic programming to solve the all pair shortest path problem. Below are the detailed steps used in

Floyd Warshall algorithm to find the all pair shortest path from a single source vertex to all other vertices in the given graph.

A. Discuss Floyd Warshall algorithm :

Algorithm [2]

1)a set *sptSet* (shortest path tree set) should be created that keeps track of vertices included in shortest path tree, i.e., whose minimum distance from source is calculated and finalized. Initially, this set is empty.

2) Then we have to assign a distance value to all vertices in the input graph and Initialize all distance values as INFINITE. Distance value should be assigned as 0 for the source vertex so that it is picked first.

3) While *sptSet* doesn't include all vertices

….**a)** We have to pick a vertex u which is not there in *sptSet*and has minimum distance value.

….**b)** then we will include u to *sptSet*.

….**c)** After that we will update distance value of all adjacent vertices of u. To update the distance value, all adjacent vertices should be iterated . For every adjacent vertex v, if sum of distance value of u (from source) and weight of edge u-v, is less than the distance value of v, the distance value of v will be updated.

A . Pseudo code of Warshall algorithm[3]
1. $D \leftarrow W$ // initialize D array to W [ ] 2. $P \leftarrow 0$ // initialize P array to [0] 3. for $k \leftarrow 1$ to n 4. do for $i \leftarrow 1$ to n 5. do for $j \leftarrow 1$ to n 6. if ($D[i, j] > D[i, k] + D[k, j]$) 7. then $D[i, j] \leftarrow D[i, k] + D[k, j]$ 8. $P[i, j] \leftarrow k$;

Example :

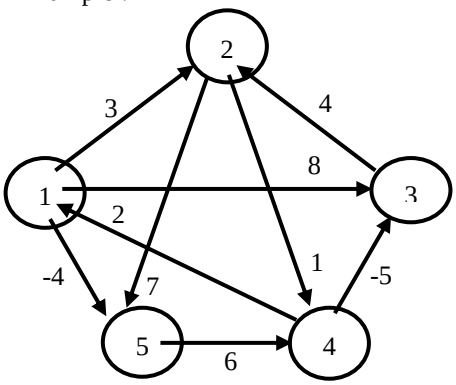

Fig 1 :- Before applying Floyd Warshall Algorithm

In every step gives a shortest path and it work will be continue until vertex -1 times. Let discuss about example

$D^{(0)} = W$

	1	2	3	4	5
1	0	3	8	∞	-4
2	∞	0	∞	1	7
3	∞	4	0	∞	∞
4	2	∞	-5	0	∞
5	∞	∞	∞	6	0

$D^{(1)}$

0	3	8	∞	-4
∞	0	∞	1	7
∞	4	0	∞	∞
2	5	-5	0	-2
∞	∞	∞	6	0

$D^{(2)}$

0	3	8	4	-4
∞	0	∞	1	7
∞	4	0	5	11
2	5	-5	0	-2
∞	∞	∞	6	0

$D^{(3)}$

0	3	8	4	-4
∞	0	∞	1	7
∞	4	0	5	11
2	-1	-5	0	-2
∞	∞	∞	6	0

$D^{(4)}$

0	3	-1	4	-4
3	0	-4	1	-1
7	4	0	5	3
2	-1	-5	0	-2
8	5	1	6	0

In D(4) we have all pair shortest path which is done by warshall algorithm. In this operation we can decided that ,if a graph has v vertex ,then loop continue v-1 times to find out all pair shortest path.

```
============== Welcome to The Digital Zoo Management ==============

        1. Check All The Animals & Interests
        2. Set a Time Consuming Travel Plan
        3. Set a Trouble Consuming Travel Plan
        4. Set a Full Day Travel Plan
        5. Set a Limited Time Travel Plan
        6. Set a Work Plan For The Cleaners
        7. Set a Work Plan For The Food Servers
        8. Find an Specific Location That Your Are looking For

Please Enter Your Choice: 8

Please Enter Your Nearest Cage Name: Bird Home

Please Enter Your Destination Cage Name: Peacock

Please Follow This Path: Birds Home -> Flamingo -> Peacock
```

Fig : Finding all pair zoo cage using warshall

B. Discuss about Dijkstra's Algorithm

For Dijkstra's algorithm we have find out one-source shortest path .

Pseudo code of Dijkstra's algorithm[4]

```
1.function Dijkstra(Graph, source):
2.create vertex set Q
3.for each vertex v in Graph:
4.  dist[v] ← INFINITY
5.  prev[v] ← UNDEFINED
6.  add v to Q
7.  dist[source] ← 0
8.  while Q is not empty:
9.    u ← vertex in Q with min dist[u]
10.   remove u from Q
11.   for each neighbor v of u:
12.       alt ← dist[u] + length(u, v)
13.       if alt < dist[v]:
14.       dist[v] ← alt
15.       prev[v] ← u
16.       return dist[], prev[]
```

Example :

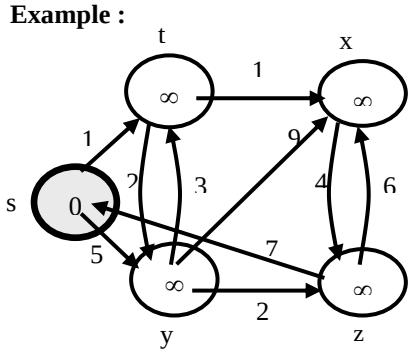

Fig : Before applying Dijkstra's algorithm

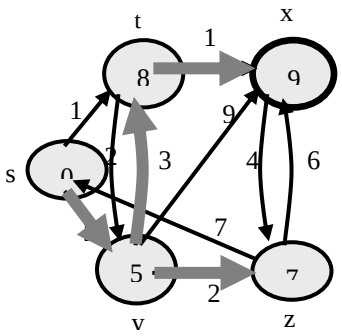

Fig : After applying Dijkstra's algorithm

```
============== Welcome to The Digital Zoo Management ==============

        1. Check All The Animals & Interests
        2. Set a Time Consuming Travel Plan
        3. Set a Trouble Consuming Travel Plan
        4. Set a Full Day Travel Plan
        5. Set a Limited Time Travel Plan
        6. Set a Work Plan For The Cleaners
        7. Set a Work Plan For The Food Servers
        8. Find an Specific Location That Your Are looking For

Please Enter Your Choice: 2
node to node path 1 4
Birds Home -> Peacock
Distance : 6
```

Fig : Finding shortest path between two zoo cage for time consuming ,using Dijkstra's

```
============== Welcome to The Digital Zoo Management ==============

        1. Check All The Animals & Interests
        2. Set a Time Consuming Travel Plan
        3. Set a Trouble Consuming Travel Plan
        4. Set a Full Day Travel Plan
        5. Set a Limited Time Travel Plan
        6. Set a Work Plan For The Cleaners
        7. Set a Work Plan For The Food Servers
        8. Find an Specific Location That Your Are looking For

Please Enter Your Choice: 3
node to node path 1 4
Birds Home -> Flamingo -> Peacock
distance  7
```

Fig : Finding shortest path between two zoo cage for trouble consuming travel ,using Dijkstra's algorithm

C. Discuss about 0-1 Knapsack Algorithm :

0-1 Knapsack Algorithm

```
// Input:
 2 // Values (stored in array v)
 3 // Weights (stored in array w)
 4 // Number of distinct items (n)
 5 // Knapsack capacity (W)
 6
 7 for j from 0 to W do:
 8     m[0, j] := 0
 9
10 for i from 1 to n do:
11     for j from 0 to W do:
12         if w[i] > j then:
13             m[i, j] := m[i-1, j]
14         else:
15         m[i, j] := max(m[i-1, j], m[i-1, j-w[i]] + v[i])
```

We have use this with dynamic programming.

Two key ingredients of optimization problems that lead to a dynamic programming solution: 1) Optimal substructure: an optimal solution to the problem contains within it optimal solutions to subproblems.

2) Overlapping subproblems: same subproblem will be visited again and again (i.e., subproblems share sub subproblems).

4.Complexity Analysis & Comparison

Floyd–Warshall algorithm works with 3 nested loop and that's why in worst time complexity is $O(|V|^3)$ where V means vertex.

On the other hand, With min-priority queue Dijkstra's algorithm is worst time complexity is O(ElogV) where E means Edges and V means vertex.

The Floyd–Warshall algorithm is a good choice for computing paths between all pairs of vertices in dense graphs, in which most or all pairs of vertices are connected by edges. For sparse graphs with non-negative edge weights, a better choice is to use Dijkstra's algorithm from each possible starting vertex, since the running time of repeated Dijkstra ($O(|E||V|+|V|^2\log|V|)$ using fibonacci heaps) is better than the $O(|V|^3)$ running time of the Floyd–Warshall algorithm.

Time complexity of knapsack algorithm is O(n*W) where n is number of distinct item and W is knapsack capacity .

In our program we have used two algorithm ,so our time complexity in worst case is $\max(|V|^3,(ElogV),(n*W))$.so the time complexity is $O(|V|^3)$.

5. Conclusion

In this project Dijkstra's algorithm is used to find out the shortest path between source node to end node. If visitor in zoo and want to go one node to another node (that's mean one place to another place),then we used Floyd warshall algorithm. Because it gives all pair shortest path .The proposed algorithms can limit the search in a sub-graph based on the given nodes of the distance between the two nodes. As a result, the calculation for the shortest path has been simplified. So, we need this type of project because, due to the development of this algorithm it is possible to determine the shortest route in zoo.

6. Future work

Here we have some limitation such as time limitation and we have work with small graph. But in future we hope to design a good software with a big project like Transit node routing , Reach-based pruning.

Acknowledgment

We thank Md. Shamsujjoha sir to give us this project and for helping us in theoretical part of this project.

REFERENCES

1. [3]*Floyd, Robert W. (June 1962). "Algorithm 97: Shortest Path". Communications of the ACM. **5** (6): 345. doi:10.1145/367766.368168*
2. [3]*Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L. (1990). Introduction to Algorithms (1st ed.). MIT Press and McGraw-Hill. ISBN 0-262-03141-8.* See in particular Section 26.2, "The Floyd–Warshall algorithm", pp. 558–565 and Section 26.4, "A general framework for solving path problems in directed graphs", pp. 570–576..
3. [1] Hongmei Wang; Ming Hu; Wei Xiao, "A new public transportation data model and shortest-path algorithms," Informatics in Control, Automation and Robotics (CAR), 2010 2nd International Asia Conference on, vol.1, no., pp.456,459, 6-7 March 2010.
4. [2] and [4] Introduction to Algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein