DATA STRUCTURES

All pair Shortest Path

Floyd Warshall Algorithm

Session Objectives

- To learn about all pair shortest path algorithm using Floyd Warshall algorithm

Session Outcomes

- At the end of this session, participants will be able to
  – Understand all pair shortest path algorithm
  – Work on Floyd Warshall Algorithm

Agenda

- Floyd Warshall Algorithm
- Example

All pair Shortest Path (Floyd Warshall) Algorithm

Dr. B. Bharathi
SSNCE

Floyd Warshall Algorithm

The All-Pairs Shortest Path Problem asks to find the length of the shortest path between any pair of vertices in G.

Floyd Warshall algorithm is an All-Pairs Shortest Path Problem

Floyd Warshall Algorithm

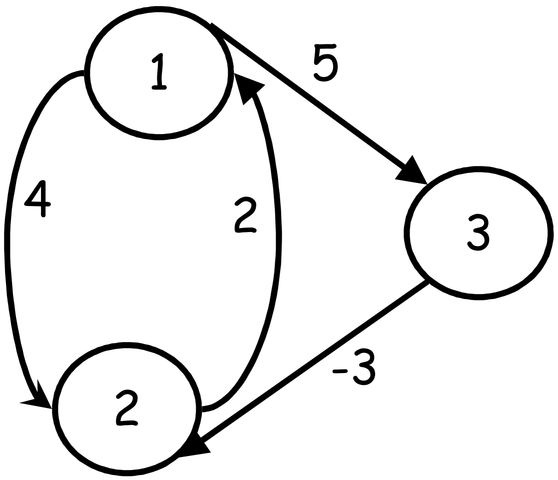
- D – Distance Matrix
- Pred – Predecessor Matrix with the intermediate information

Floyd Warshall Algorithm

```
Floyd_Warshall(int n, int W[1..n, 1..n]) {
 array d[1..n, 1..n]
  for i = 1 to n do {                                   // initialize
  for j = 1 to n do {
     d[i,j] = W[i,j]
   pred[i,j] = null}}
 for k = 1 to n do                            // use intermediates{1..k}
 for i = 1 to n do                            // ...from i
 for j = 1 to n do                            // ...to j
  if (d[i,k] + d[k,j]) < d[i,j]){
   d[i,j] = d[i,k] + d[k,j]                   // new shorter path length

   pred[i,j] = k      }                       // matrix of final distances
 return d                                     // new path is through k
 }
```

Example

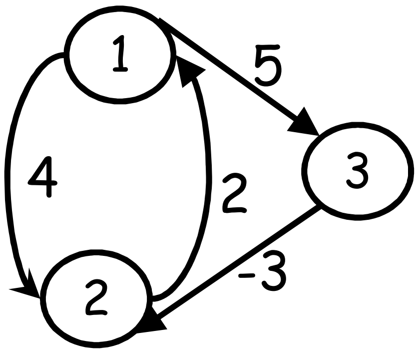


$W = D^0 =$

	1	2	3
1	0	4	5
2	2	0	∞
3	∞	-3	0

P =

	1	2	3
1	0	0	0
2	0	0	0
3	0	0	0



D^0 =

	1	2	3
1	0	4	5
2	2	0	∞
3	∞	-3	0

k = 1
Vertex 1 can be intermediate node

D^1 =

	1	2	3
1	0	4	5
2	2	0	7
3	∞	-3	0

$D^1[2,3] = \min(D^0[2,3], D^0[2,1]+D^0[1,3])$

$= \min(\infty, 7)$

$= 7$

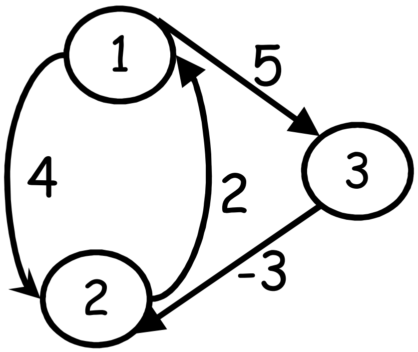
P =

	1	2	3
1	0	0	0
2	0	0	1
3	0	0	0

$D^1[3,2] = \min(D^0[3,2], D^0[3,1]+D^0[1,2])$

$= \min(-3, \infty)$

$= -3$



D^1 =

	1	2	3
1	0	4	5
2	2	0	7
3	∞	-3	0

k = 2
Vertices 1, 2 can be intermediate

D^2 =

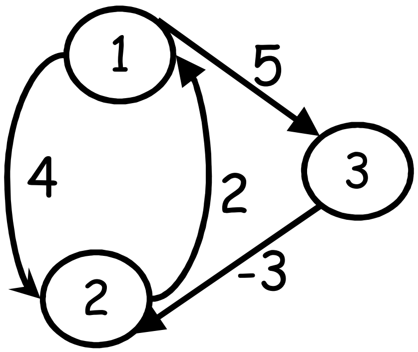
	1	2	3
1	0	4	5
2	2	0	7
3	-1	-3	0

$D^2[1,3] = \min(D^1[1,3], D^1[1,2]+D^1[2,3])$
$= \min (5, 4+7)$
$= 5$

P =

	1	2	3
1	0	0	0
2	0	0	1
3	2	0	0

$D^2[3,1] = \min(D^1[3,1], D^1[3,2]+D^1[2,1])$
$= \min (\propto, -3+2)$
$= -1$



D^2 =

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

k = 3
Vertices 1, 2, 3 can be intermediate

D^3 =

	1	2	3
1	0	2	5
2	2	0	7
3	-1	-3	0

$D^3[1,2] = \min(D^2[1,2], D^2[1,3]+D^2[3,2])$
$= \min(4, 5+(-3))$
$= 2$

P =

	1	2	3
1	0	3	0
2	0	0	1
3	2	0	0

$D^3[2,1] = \min(D^2[2,1], D^2[2,3]+D^2[3,1])$
$= \min(2, 7+(-1))$
$= 2$

Distance & Predecessor Matrix Updations

$D^{(0)}$ - Initial

0	4	7
1	0	2
6	∞	0

Pre. Matrix

0	0	0
0	0	0
0	0	0

$D^{(1)}$ – Intermediate Vertex

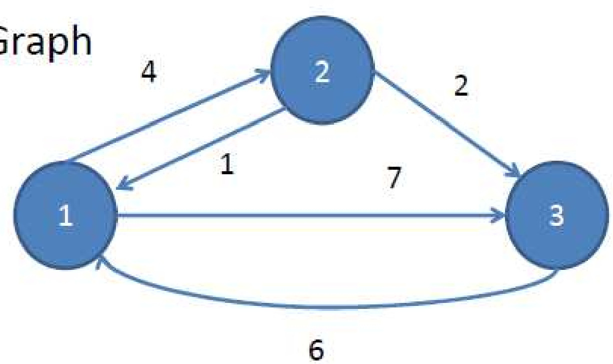
0	4	7
1	0	2
6	10	0

Pre. Matrix

0	0	0
0	0	0
0	1	0

Considering Intermediate **Vertex 1**

Sample Graph



D(3,1)+D(1,2)<D(3,2)

10 < ∞

Distance & Predecessor Matrix Updations

$D^{(2)}$ – Intermediate Vertex

0	4	6
1	0	2
6	10	0

Pre. Matrix

0	0	2
0	0	0
0	1	0

$D^{(3)}$ – Intermediate Vertex

0	4	6
1	0	2
6	10	0

Pre. Matrix

0	0	2
0	0	0
0	1	0

How to find shortest path between any pair of vertices?

Algorithm:

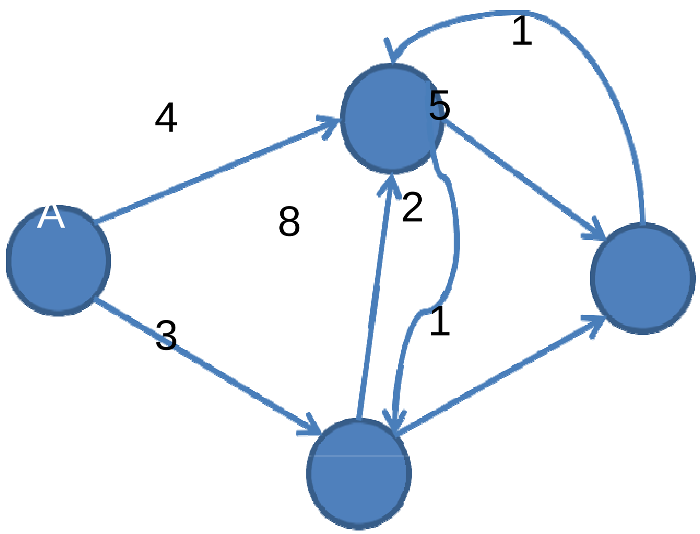
```
Path(i,j)
{
   if pred[i,j] =null                    // path is a single edge
    output(i,j)
   else
   {                                     // path goes through
   pred
        Path(i, pred[i,j]); // print path from i to pred
        Path(pred[i,j], j); // print path from pred to j
   }
}
```

Example – Shortest path from vertex 2 to vertex 3

2..3	path(2,3)	pred(2,3) =4
2..4..3	path(2,4)	pred(2,4) =5
2..5..4..3	path(2,5)	pred(2,5) = nil output(2,5)
25..4..3	path(5,4)	pred(5,4) =nil output(5,4)
254..3	path(4,3)	pred(4,3)=nil output(4,3)

So Path is 2->5->4->3

Excercise



Summary

- All pair shortest path
- Floyd Warshall Algorithm
- Example