Data Structures and Algorithm

COMP2421

Graph and MST Algorithms revision

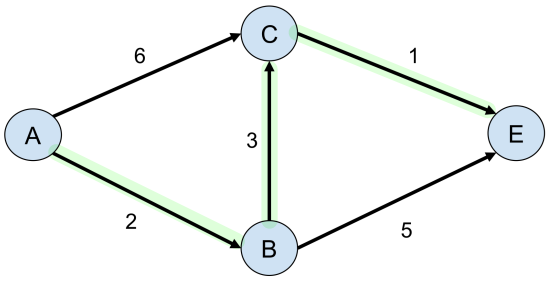
By: Jibreel Bornat

أنتَ مسؤولٌ فقط عنِ السَعي،لَيس النَتيجةَ.
فالنَتيجة إختَص بِها ربُّ الخَير
الَذي لايأتِي إلابالخَير.

① Dijkstra Algorithm :-

1- Choose a source Vertex
2- make a Chart that contain all vertices
3- assign the source to ZERO and the rest ∞
4- start from the less edge weight
5- if the new values are less from the ones in the Chart, Change them
6- Continue to all vertices until the destination

	I_0	I_1	I_2	I_3
A	0	0	0	0
B	∞	2	2	2
C	∞	6	5	5
D	∞	∞	7	6



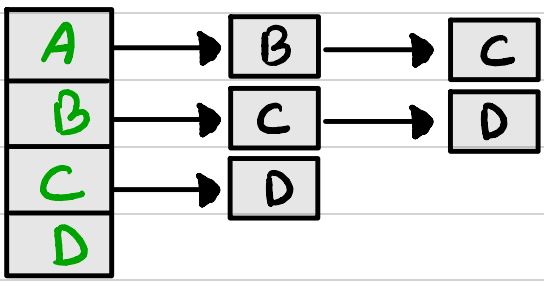
② Adjacency Matrix :-

	A	B	C	D
A	0	2	6	0
B	0	0	3	5
C	0	0	0	1
D	0	0	0	0

1- you see any Direct Connection between two Vertices and mark them in the Chart

2- the rest you fill it with Zeros

③ Adjacency List :-



1- you see any Direct Connection between two Vertices and add it in the list

④ Bellman Ford :-

1- Choose a source Vertex
2- make a Chart that contain all vertices
3- assign the source to ZERO and the rest ∞
4- you have to iterate on all of them V - 1 times
5- you start from the source vertex and see it's edges and what does it Change
* A → B → C → D for 4-1 = 3 times
6- Stop when : - you reach the limit (V-1)
- No Change from the previous iterate

⑤ Topological Sort :-

1- make a list that has all the number of edges
2- start putting the vertices according to the number of ENTERING edges
3- Choose the less Vertex with entering edges
4- Remove it and it's edges
5- Repeat the process until vertices are done

① | 0 | r, |
|---|---|
| 1 | s |
| 2 | t, u, v, w |

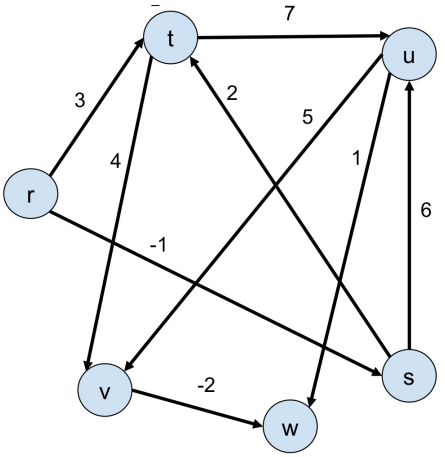
② | 0 | s |
|---|---|
| 1 | t |
| 2 | u, v, w |

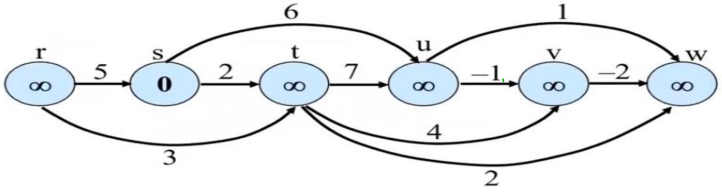
③ | 0 | t |
|---|---|
| 1 | u |
| 2 | v, w |

④ | 0 | u |
|---|---|
| 1 | v |
| 2 | w |

⑤ | 0 | v |
|---|---|
| 1 | w |
| 2 | |

⑥ | 0 | w |
|---|---|
| 1 | |
| 2 | |

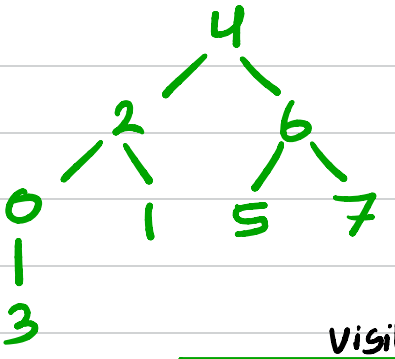




#

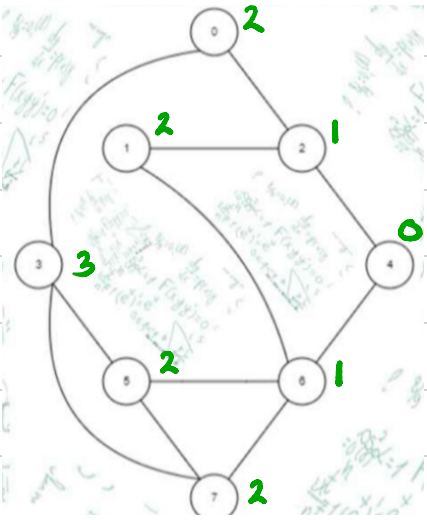
⑥ Breadth First Search (BFS) :-

1- Start from the required vertex and add it to the visited tree then assign it to ZERO
2- go to its adjacent and assign them to ONE then add them to the tree and mark them visited
3- do the same and keep incriminating each level by 1 , don't add the visited vertices



visited queue

4	2	6	5	7	0	1	3



⑦ Prim's Algorithm :-

1- Choose the start point then go to the shortest path
2- mark the vertex as visited , you end up with two vertices , Choose the shortest path from both
3- keep Choosing the shortest paths until you reach the last vertex , make sure there is no redundancy

⑧ Kruskal Algorithm :-

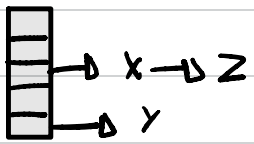
1- assign all the paths with their wight (A $\overset{4}{—}$ B)
2- sort them from lower to higher edges
3- connect them from low to high
4- Avoid to put existing vertices

"هو اختارني لهذا الطريق ، فإمَّا سيُيَسِّرُه لي أو سيُعينُني عليه"

⑨ Hash :-

① seperate Chain :-

make array and linked list



② Open addressing :-

1- Linear hashing :

$h(k) = (k + i) \% size$

2- Quadratic hashing :

$h(k) = (k + i^2) \% size$

3- Double hashing :

$h(k) = h_1(k) + i\, h_2(k)$

$(k \% size) + i * (R - k \% R)$

* R : prime < size

4- String hashing :

he gives you the key or by finding the ASCII

$S[0] * 32^1 + S[1] * 32^{i-1} + \dots$

example : ALI

$= A * 32^2 + L * 32^1 + I * 32^0$