Spanning Tree

- Any tree consisting solely of edges in *G* and including all vertices in *G* is called a *spanning tree*.
- Spanning tree can be obtained by using either a depth-first or a breath-first search.
- When a nontree edge (v, w) is introduced into any spanning tree T, a cycle is formed.
- A spanning tree is a minimal subgraph, *G'*, of *G* such that V(*G'*) = V(*G*), and *G'* is connected. (Minimal subgraph is defined as one with the fewest number of edges).
- Any connected graph with n vertices must have at least n-1 edges, and all connected graphs with n – 1 edges are trees. Therefore, a spanning tree has n – 1 edges.

Spanning Tree

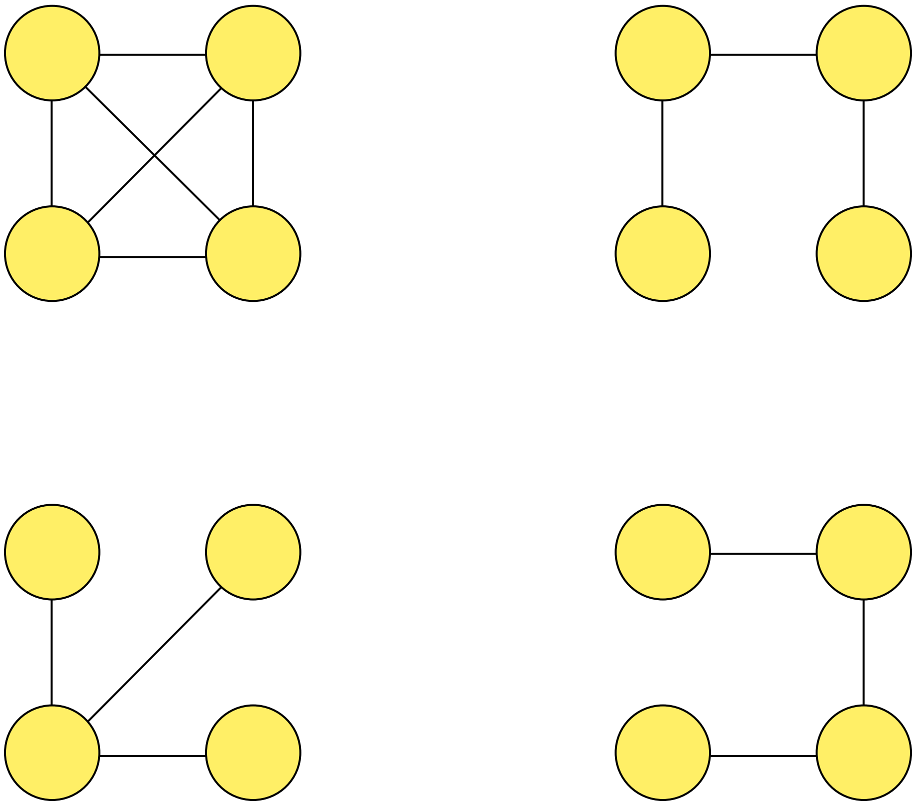
When G is connected, DFS or BFS is applied, then the edges is partitioned into T and N

T: edges used during traversal, also called tree edges

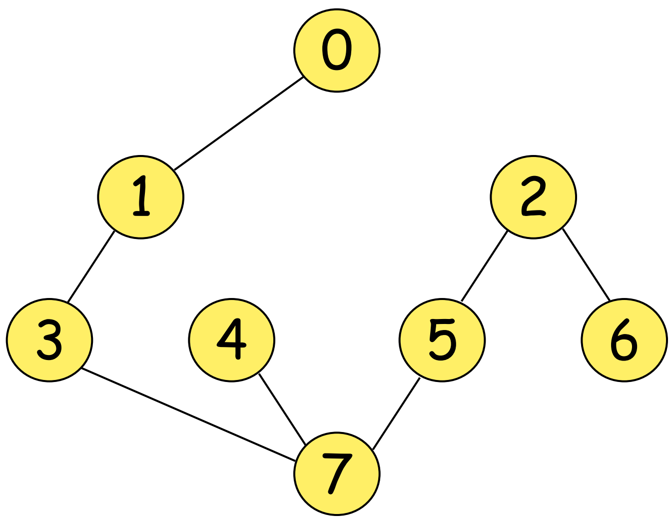
N: nontree edges

Spanning tree: all vertices + T

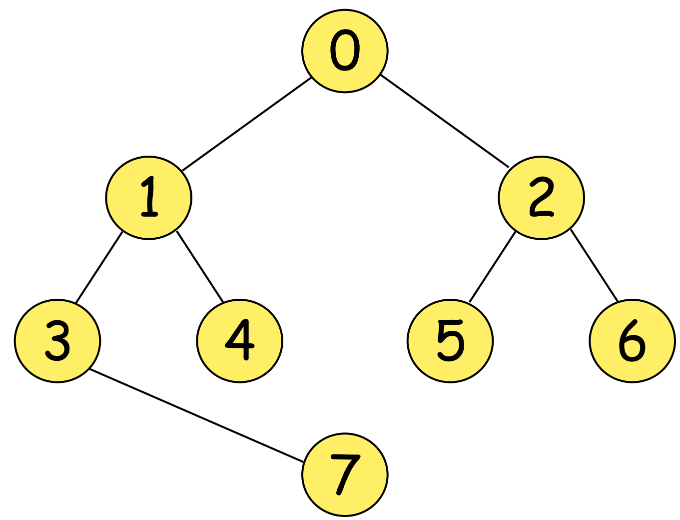
A Complete Graph and Three of Its Spanning Trees



Depth-First and Breath-First Spanning Trees



(a) DFS (0) spanning tree

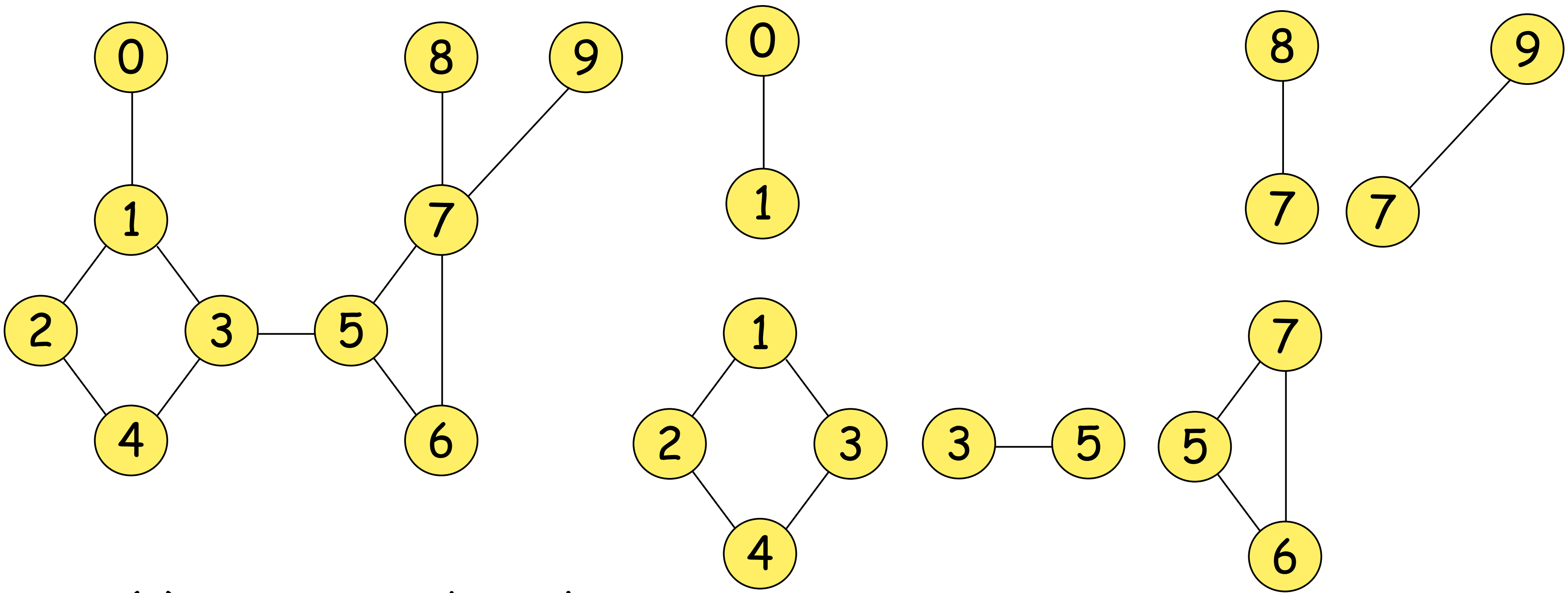


(b) BFS (0) spanning tree

Biconnected Components

- Definition: A vertex v of G is an articulation point iff the deletion of v, together with the deletion of all edges incident to v, leaves behind a graph that has at least two connected components.
- Definition: A biconnected graph is a connected graph that has no articulation points.
- Definition: A biconnected component of a connected graph G is a maximal biconnected subgraph H of G. By maximal, we mean that G contains no other subgraph that is both biconnected and properly contains H.

A Connected Graph and Its Biconnected Components



(a) A connected graph

(b) Its biconnected components

Maximal without articulation point

Efficiency of Algorithm

- Algorithm efficiency is equal to the function of number of elements to be processed.
- We must know efficiency of loop

Linear loop

- Example

```
i=1
  Loop(i<=10)
    ...
  i=i+1
```

Logarithmic loop

Example 1

i=1

Loop(i<1000)

....

i=i*2

Example 2

i=1000

Loop(i>=1)

...

i=i/2

Logarithmic loop (continued)

Iteration	Value of i(multiplication)	Iteration	Value of i(Division)
1	1	1	1000
2	2	2	500
3	4	3	250
4	8	4	125
5	16	5	62
6	32	6	31
7	64	7	15
8	128	8	7
9	256	9	3
10	512	10	1
Exit	1024	Exit	0

Nested loop

- Iteration=Outer loop iteration* Inner loop iteration
- Three types of nested loop

➢ Linear Logarithmic

➢ Dependent Quadratic

➢ Quadratic

Linear logarithmic

- Example

```
i=1
    loop(i<=10)
        j=1
            loop(j<=10)
            ...
            j=j*2
    i=i*2
```

Dependent Quadratic

- Example

```
i=1
   loop(i<=10)
   j=1
        loop(j<=i)
        ...
        j=j+1
   i=i+1
```

Quadratic

- Example

```
i=1
   loop(i<=10)
   j=1
        loop(j<=10)
        ...
        j=j+1
        i=i+1
```

Example1

Statement	s/e	frequency	Total steps
Algorithm Sum(a, n)	0	-	0
{	0	-	0
s=0.0;	1	1	1
for i:=1 to n do	1	n+1	n+1
s=s +a[i];	1	n	n
return s;	1	1	1
}	0	-	0
Total			2n+3

Application

- Network flow
- Bridge Block problem
- Cluster

Scope of research

- **Rapid protein side-chain prediction**

Assignment

Q.1) What is bi-connected graph? Give an example of the bi-connected component.

Q.2)What is articulation point?

Q.3What is efficiency of following algorithm

```
Unsigned int fibonacci (Unsigned int n)
{
    int previous=-1;
    int result=1;
    for(unsigned int i=0;i<=n;++i)
    {
    int sum=result+previous;
    previous=result;
    result=sum;
    }
    return sum;
}
```