MH1301 Discrete Mathematics

Tutorial 11

Academic Year 2025/2026, Semester 2

Quantitative Research Society @NTU

August 17, 2026

Overview

This tutorial develops the basic theory of trees, forests, complete bipartite trees, and binary trees. The main themes are:

- classification of trees with a vertex of very large degree;
- equivalent characterisations of trees using connectivity, cycles, and bridges;
- identifying which complete bipartite graphs are trees;
- edge-count formulae for trees, forests, and full binary trees;
- construction and counting formulae for complete binary trees.

Question map.

- Questions 1–3: non-isomorphic trees with one vertex of degree $n-1$, $n-2$, or $n-3$.
- Question 4: a bridge-deletion characterisation of trees.
- Question 5: complete bipartite graphs that are trees.
- Questions 6–7: edge counts in trees, full binary trees, and forests.
- Question 8: complete binary trees of height h.

For academic reference only. This document is provided for educational purposes and does not constitute an official question paper, answer key, or any formally authorised academic material. Its content is not guaranteed to be complete or accurate.

Contents

Question 1 (Trees with a vertex of degree $n-1$) **4**
Problem . . . 4
Solution . . . 4
Method 1: Direct degree-neighbourhood classification . . . 4
Method 2: Edge-count extremal argument . . . 4

Question 2 (Trees with a vertex of degree $n-2$) **5**
Problem . . . 5
Solution . . . 5
Method 1: Edge-budget classification . . . 5
Method 2: Branch-length partition . . . 5

Question 3 (Trees with a vertex of degree $n-3$) **7**
Problem . . . 7
Solution . . . 7
Method 1: Component classification after deleting the high-degree vertex . . . 7
Method 2: Branch-size partitions and branch shapes . . . 8

Question 4 (Bridge-deletion characterisation of trees) **9**
Problem . . . 9
Solution . . . 9
Method 1: Cycle contradiction and bridge property . . . 9
Method 2: Unique-path characterisation . . . 10

Question 5 (Complete bipartite graphs that are trees) **11**
Problem . . . 11
Solution . . . 11
Method 1: Cycle obstruction . . . 11
Method 2: Edge-count characterisation . . . 11

Question 6 (Edges in trees and full binary trees) **13**
Problem . . . 13
Solution . . . 13
Method 1: Standard tree and child-edge formulae . . . 13
Method 2: Internal-leaf count and total-vertex count . . . 13

Question 7 (Edges in a forest) **15**
Problem . . . 15
Solution . . . 15
Method 1: Sum over tree components . . . 15
Method 2: Component-count invariant . . . 15

QRS@NTU notes.org

Question 8 (Complete binary trees) **16**
Problem 16
Solution 16
Method 1: Level-by-level construction and counting 16
Method 2: Recursive recurrence 17

Unofficial — QRS@NTU — qrsntu.org

Question 1 (Trees with a vertex of degree $n-1$)

Problem

Assume that $n \geq 7$. How many non-isomorphic trees with n vertices are there in which one vertex has degree $n-1$?

Solution

Method 1: Direct degree-neighbourhood classification

Let T be a tree on n vertices. Suppose some vertex $v \in V(T)$ has

$$\deg(v) = n-1.$$

Since T has exactly n vertices, the vertex v is adjacent to every other vertex.

Thus every edge incident with v is of the form

$$vv_i, \qquad v_i \in V(T) \setminus \{v\}.$$

There are already $n-1$ such edges. A tree on n vertices has exactly $n-1$ edges, so there can be no further edges among the vertices in $V(T) \setminus \{v\}$.

Therefore

$$T \cong K_{1,n-1},$$

the star on n vertices. Hence there is exactly one isomorphism type:

$$\boxed{1}.$$

Method 2: Edge-count extremal argument

Theorem/criterion used. Every tree on n vertices has exactly $n-1$ edges.

If a vertex v has degree $n-1$, then the edges incident with v already account for

$$n-1$$

distinct edges. Since a tree on n vertices has exactly $n-1$ edges, these are all the edges of the tree.

Hence the tree is completely determined:

$$E(T) = \{vv_i : v_i \in V(T) \setminus \{v\}\}.$$

This is the star $K_{1,n-1}$. Therefore all such trees are isomorphic, and the number of non-isomorphic trees is

$$\boxed{1}.$$

Question 2 (Trees with a vertex of degree $n-2$)

Problem

Assume that $n \geq 7$. How many non-isomorphic trees with n vertices are there in which one vertex has degree $n-2$?

Solution

Method 1: Edge-budget classification

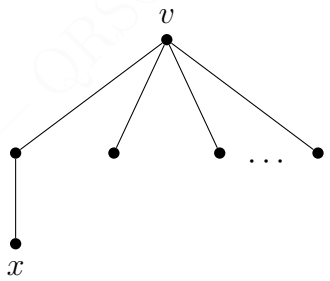
Let T be a tree on n vertices. Suppose some vertex v has

$$\deg(v) = n-2.$$

Then v is adjacent to exactly $n-2$ of the other $n-1$ vertices. Hence there is exactly one vertex, say x, not adjacent to v.

The $n-2$ edges incident with v already occur in T. Since T has exactly $n-1$ edges, there is exactly one remaining edge. The vertex x must be connected to the rest of the graph because T is connected. Since x is not adjacent to v, this remaining edge must connect x to one of the neighbours of v.

All neighbours of v are symmetric under isomorphism. Therefore every such tree is isomorphic to a star with one pendant edge subdivided once:



Thus there is exactly one non-isomorphic tree of this type:

$$\boxed{1}.$$

Method 2: Branch-length partition

Root the tree at the vertex v of degree $n-2$. Since v has $n-2$ incident edges, it has $n-2$ branches starting from it.

There are

$$n-1$$

vertices other than v. These are distributed among $n-2$ branches. Each branch contains at least one vertex. Hence the branch-size multiset is a partition of $n-1$ into $n-2$ positive parts:

$$n-1 = 2+1+\cdots+1.$$

Thus exactly one branch has length 2, and all other branches have length 1. This determines the rooted tree up to isomorphism, and hence determines the unrooted tree up to isomorphism.

Therefore the number of non-isomorphic trees is

$$\boxed{1}.$$

Unofficial — QRS@NTU — qrsntu.org

Question 3 (Trees with a vertex of degree $n-3$)

Problem

Assume that $n \geq 7$. How many non-isomorphic trees with n vertices are there in which one vertex has degree $n-3$?

Solution

Method 1: Component classification after deleting the high-degree vertex

Let T be a tree on n vertices. Suppose some vertex v has

$$\deg(v) = n - 3.$$

Deleting v from a tree separates the tree into exactly $\deg(v)$ components. Therefore

$$T - v$$

is a forest with

$$n - 1$$

vertices and

$$n - 3$$

components.

A forest with N vertices and c components has $N - c$ edges. Hence

$$|E(T - v)| = (n - 1) - (n - 3) = 2.$$

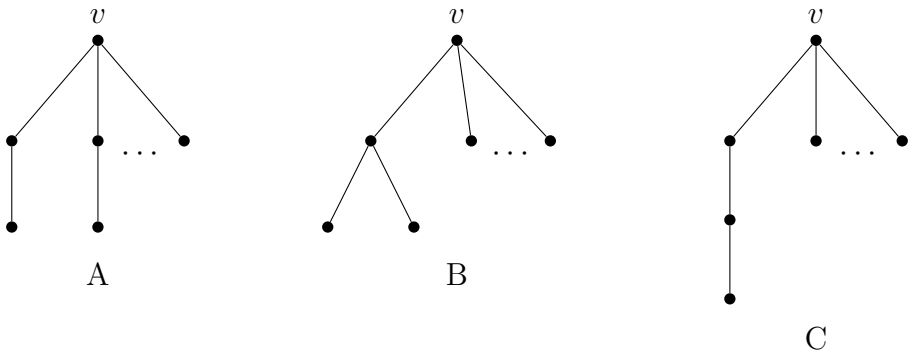
Thus the classification reduces to classifying a forest on $n-1$ vertices with exactly 2 edges and $n-3$ components, where the $n-3$ components correspond to the $n-3$ branches from v.

There are exactly three possible component types involving the two additional edges:

$$K_2 \sqcup K_2, \qquad K_{1,2}, \qquad P_3.$$

Equivalently, the tree has one of the following three branch patterns at v:

A: two branches of length 2 B: one split branch C: one branch of length 3



A B C

These three types are pairwise non-isomorphic:

- Type A has two vertices at distance 2 from v and none at distance 3 from v.
- Type B has a neighbour of v of degree 3.
- Type C has one vertex at distance 3 from v.

Thus there are exactly three non-isomorphic trees:

$$\boxed{3}.$$

Method 2: Branch-size partitions and branch shapes

Root the tree at a vertex v of degree $n-3$. Then there are $n-3$ branches from v, and these branches contain all

$$n-1$$

vertices other than v.

Thus the branch sizes form a partition of $n-1$ into $n-3$ positive integers. Equivalently, starting from

$$1+1+\cdots+1$$

with $n-3$ parts, we must distribute two additional vertices among the branches. There are two possible size patterns:

$$3,1,1,\ldots,1 \qquad \text{or} \qquad 2,2,1,\ldots,1.$$

For the pattern

$$2,2,1,\ldots,1,$$

both branches of size 2 are forced to be paths of length 2. This gives Type A.

For the pattern

$$3,1,1,\ldots,1,$$

there are two rooted tree shapes on a branch of size 3:

$$\text{a path of length 3} \qquad \text{or} \qquad \text{one neighbour with two children.}$$

These give Type C and Type B respectively.

Hence the total number of rooted branch shapes, and therefore the total number of non-isomorphic unrooted trees, is

$$1+2=3.$$

Therefore

$$\boxed{3}.$$

Question 4 (Bridge-deletion characterisation of trees)

Problem

(Ex 11.1.8). Prove that a simple graph is a tree if and only if it is connected, but the deletion of any of its edges produces a graph that is not connected.

Solution

Method 1: Cycle contradiction and bridge property

Theorem/criterion used. A tree is a connected simple graph with no circuits. An edge is a bridge if deleting it disconnects the graph.

We prove both directions.

Forward direction. Suppose G is a tree. Let $e = uv \in E(G)$. Consider $G - e$.

If $G - e$ were connected, then there would still be a path from u to v in $G - e$. Let this path be

$$u = x_0, x_1, \ldots, x_k = v.$$

Then in G, the path together with the edge uv forms a circuit:

$$u = x_0, x_1, \ldots, x_k = v, u.$$

This contradicts the fact that a tree has no circuits. Hence $G - e$ is disconnected. Since e was arbitrary, deleting any edge of G disconnects the graph.

Reverse direction. Suppose G is connected and deleting any edge disconnects G. We prove that G has no circuit.

Assume for contradiction that G contains a circuit

$$C = v_1 v_2 \cdots v_k v_1.$$

Delete the edge

$$e = v_1 v_2.$$

The vertices v_1 and v_2 remain connected in $G - e$ by the path

$$v_1, v_k, v_{k-1}, \ldots, v_2.$$

Moreover, any path in G that used the deleted edge $v_1 v_2$ can replace that edge by the remaining part of the circuit. Thus deleting e does not disconnect G, contradicting the hypothesis.

Therefore G is connected and has no circuit. Hence G is a tree.

Thus

G is a tree if and only if G is connected and deleting any edge disconnects G.

Unofficial — QRS@NTU — qrs.ntu.edu.sg

Method 2: Unique-path characterisation

Theorem/criterion used. A connected simple graph is a tree if and only if there is exactly one simple path between every two distinct vertices.

Forward direction. Suppose G is a tree. Let $e = uv$ be an edge. If $G - e$ were connected, then u and v would still be joined by a path in $G - e$. Together with the edge uv, this would give two distinct simple paths between u and v in G: the single-edge path uv and the path in $G - e$. This contradicts the unique-path characterisation. Hence $G - e$ is disconnected.

Reverse direction. Suppose G is connected and deleting any edge disconnects G. We show that any two distinct vertices have a unique simple path.

Since G is connected, at least one path exists between any two vertices. If there were two distinct simple paths between some vertices x and y, then their union would contain a circuit. Choose an edge e on that circuit. Deleting e would not disconnect the graph, because the endpoints of e remain connected by the rest of the circuit, and all other paths using e may be rerouted around the circuit. This contradicts the hypothesis.

Hence every pair of distinct vertices has exactly one simple path between them. Therefore G is a tree. Thus

G is a tree exactly when it is connected and every edge is a bridge.

Unofficial — QRS@NTU

Question 5 (Complete bipartite graphs that are trees)

Problem

(Ex 11.1.10). Which complete bipartite graphs $K_{m,n}$, where m and n are positive integers, are trees?

Solution

Method 1: Cycle obstruction

The graph $K_{m,n}$ has bipartition

$$A \sqcup B, \qquad |A| = m, \qquad |B| = n,$$

and every vertex in A is adjacent to every vertex in B.

If

$$m \geq 2 \qquad \text{and} \qquad n \geq 2,$$

choose distinct vertices

$$a_1, a_2 \in A, \qquad b_1, b_2 \in B.$$

Since the graph is complete bipartite,

$$a_1b_1,\ b_1a_2,\ a_2b_2,\ b_2a_1$$

are all edges. Hence

$$a_1 - b_1 - a_2 - b_2 - a_1$$

is a 4-cycle. Therefore $K_{m,n}$ is not a tree.

If $m = 1$, then $K_{1,n}$ is a star: it is connected and has no cycle. Hence it is a tree. Similarly, if $n = 1$, then $K_{m,1}$ is a star and is a tree.

Therefore

$$\boxed{K_{m,n} \text{ is a tree if and only if } m = 1 \text{ or } n = 1.}$$

Method 2: Edge-count characterisation

Theorem/criterion used. A connected simple graph with N vertices is a tree if and only if it has $N - 1$ edges.

The graph $K_{m,n}$ is connected for all positive integers m, n. It has

$$N = m + n$$

vertices and

$$|E(K_{m,n})| = mn$$

edges.

For $K_{m,n}$ to be a tree, it is necessary and sufficient that

$$mn = (m + n) - 1.$$

Rearranging,

$$mn - m - n + 1 = 0,$$

so

$$(m-1)(n-1) = 0.$$

Since m, n are positive integers, this holds exactly when

$$m = 1 \qquad \text{or} \qquad n = 1.$$

Therefore

$$\boxed{K_{m,n} \text{ is a tree exactly for } m = 1 \text{ or } n = 1.}$$

Unofficial — QRS@NTU — qrsntu.org

Question 6 (Edges in trees and full binary trees)

Problem

(Ex. 11.1.11, 11.1.13).

(a) How many edges does a tree with 10000 vertices have?

(b) How many edges does a full binary tree with 10000 internal vertices have?

Solution

Method 1: Standard tree and child-edge formulae

(a) **Theorem/criterion used.** A tree with n vertices has exactly $n-1$ edges.

Here

$$n = 10000.$$

Therefore the number of edges is

$$n - 1 = 10000 - 1 = 9999.$$

Hence

$$\boxed{9999}.$$

(b) In a full binary tree, every internal vertex has exactly two children. If there are

$$I = 10000$$

internal vertices, then the number of parent-child edges is

$$2I = 2(10000) = 20000.$$

Every edge in a rooted tree is exactly one parent-child edge. Therefore the full binary tree has

$$\boxed{20000}$$

edges.

Method 2: Internal-leaf count and total-vertex count

(a) A tree with 10000 vertices has

$$10000 - 1 = 9999$$

edges, so

$$\boxed{9999}.$$

(b) **Theorem/criterion used.** In a full binary tree with I internal vertices, the number of leaves is

$$L = I + 1.$$

Thus, for $I = 10000$,

$$L = 10001.$$

The total number of vertices is

$$N = I + L = 10000 + 10001 = 20001.$$

Since every tree with N vertices has $N - 1$ edges,

$$|E| = 20001 - 1 = 20000.$$

Therefore

$$\boxed{20000}.$$

Unofficial — QRS@NTU — qrsntu.org

Question 7 (Edges in a forest)

Problem

(Ex 11.1.23). How many edges are there in a forest of t trees containing a total of n vertices?

Solution

Method 1: Sum over tree components

Let the forest consist of t tree components

$$T_1, T_2, \ldots, T_t.$$

Let

$$|V(T_i)| = n_i \qquad (1 \le i \le t).$$

Then

$$n_1 + n_2 + \cdots + n_t = n.$$

Since each T_i is a tree, it has

$$n_i - 1$$

edges. Therefore the total number of edges in the forest is

$$\sum_{i=1}^{t}(n_i - 1) = \left(\sum_{i=1}^{t} n_i\right) - t = n - t.$$

Thus

$$\boxed{n - t}.$$

Method 2: Component-count invariant

Theorem/criterion used. For a forest F with n vertices and c connected components,

$$|E(F)| = n - c.$$

Here the forest has exactly t connected components, because it is a forest of t trees. Hence

$$|E(F)| = n - t.$$

Therefore

$$\boxed{n - t}.$$

Unofficial — qrsntu.org

Question 8 (Complete binary trees)

Problem

(Ex 11.1.19, 11.1.20). A complete binary tree is a full binary tree in which every leaf is at the same level.

(a) Construct a complete binary tree of height 4.

(b) How many vertices and how many leaves does a complete binary tree of height h have?

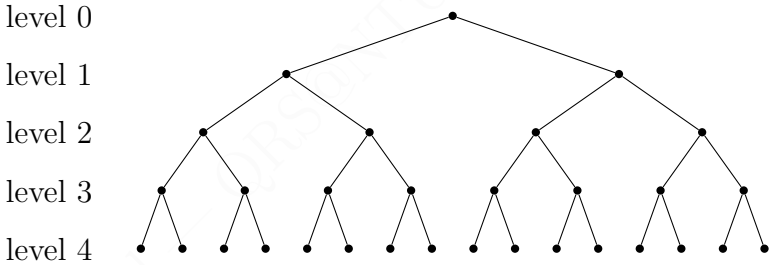
Solution

Method 1: Level-by-level construction and counting

(a) A complete binary tree of height 4 has levels

$$0, 1, 2, 3, 4,$$

with every internal vertex having exactly two children and every leaf lying at level 4. One such tree is:



(b) At level k, a complete binary tree has

$$2^k$$

vertices. Therefore the total number of vertices from level 0 to level h is

$$2^0 + 2^1 + 2^2 + \cdots + 2^h.$$

This is a finite geometric series:

$$2^0 + 2^1 + \cdots + 2^h = \frac{2^{h+1} - 1}{2 - 1} = 2^{h+1} - 1.$$

The leaves are exactly the vertices at level h, so the number of leaves is

$$2^h.$$

Hence

$$\boxed{\text{vertices } = 2^{h+1} - 1, \qquad \text{leaves } = 2^h.}$$

Method 2: Recursive recurrence

Let

$$V_h = \text{number of vertices in a complete binary tree of height } h,$$

and

$$L_h = \text{number of leaves in a complete binary tree of height } h.$$

For $h = 0$, the tree consists of a single root, so

$$V_0 = 1, \qquad L_0 = 1.$$

For $h \geq 1$, the root has two complete binary subtrees of height $h - 1$. Therefore

$$V_h = 1 + 2V_{h-1}, \qquad L_h = 2L_{h-1}.$$

Solving the leaf recurrence gives

$$L_h = 2^h L_0 = 2^h.$$

Solving the vertex recurrence gives

$$V_h = 1 + 2 + 2^2 + \cdots + 2^h = 2^{h+1} - 1.$$

Thus a complete binary tree of height h has

$$\boxed{2^{h+1} - 1 \text{ vertices and } 2^h \text{ leaves.}}$$

Unofficial — QRS@NTU — qrsntu.org