07-21 Class: Graph Theory, Tree Properties, and Euler’s Formula

> Date & Time: 2025-07-21 09:35:21
> Location: [Insert Location]
> Course Name: [Enter Course Name]

Keywords

`Graph Theory` `Tree Properties` `Graph Coloring`

Key Learnings

1. Equivalence of Tree Definitions in Graph Theory: This knowledge point explains that several statements about trees are equivalent. The statements include: (1) G is a tree, (2) every two vertices of G are connected by a unique path, (3) every two vertices of G are joined by a unique path, (4) G is connected and n = e + 1, and (5) G is acyclic and n = e + 1. The proofs show that statement 1 implies statement 2, statement 2 implies statement 3, statement 3 implies statement 4, statement 4 implies statement 5, and statement 5 implies statement 1.
2. Inductive Proof for Trees: n = e + 1: This knowledge point provides a detailed explanation of proving that for any graph where every two nodes are joined by a unique path, the relationship n = e + 1 holds, using mathematical induction.
3. Tree Property: Existence of at Least Two Endpoints (Leaves): This knowledge point shows that every tree has at least two vertices of degree 1 (endpoints) and uses the degree sum property to derive a contradiction if this is not the case.
4. Euler’s Formula for Planar Graphs: This knowledge point discusses Euler’s formula, which relates the number of vertices (n), edges (e), and faces (f) in a planar graph via the formula n - e + f = 2. The explanation uses spanning trees and dual graphs as key components in the proof.
5. Planarity and Edge Bounds in Simple Planar Graphs: This knowledge point covers the definition of a planar graph and explains that a simple planar graph with n > 2 vertices must have at least one vertex of degree at most 5. It also states that such a graph has at most 3n - 6 edges, with the proof for the degree property shown by contradiction.
6. Degree Sum Formula in Graph Theory: This knowledge point introduces the degree sum formula which states that in any graph, the sum of the vertex degrees equals

twice the number of edges (2e). This is a fundamental property used in several proofs and examples.

7. Graph Coloring in Graph Theory and Compiler Register Allocation: This knowledge point covers the concept of graph coloring, where no two adjacent vertices share the same color. It includes both theoretical and practical aspects, such as the application of graph coloring in optimally allocating registers in compilers.
8. Proof by Induction for Six-Coloring Planar Graphs: This section explains the inductive proof that every planar graph can be colored using no more than six colors. The process involves removing a vertex with a degree of at most 5 and then demonstrating that the graph can be extended to include this vertex with an available color.
9. Euler's Formula and Graph Properties: This point covers important numerical relationships in graph theory, including the fact that the sum of vertex degrees equals twice the number of edges and that for trees the number of vertices equals the number of edges plus one. These properties underpin Euler's formula for planar graphs.
10. Graph Representation Techniques: Adjacency Matrix vs. Adjacency List: This section explains two primary methods for representing graphs in computer memory: the adjacency matrix and the adjacency list. The choice of representation depends on whether the graph is dense or sparse.
11. Historical and Practical Aspects of Graph Coloring: This knowledge point discusses the historical relevance of graph coloring, particularly its importance in map-making and the economic considerations of colored inks and dyes. It also connects the historical motivations to modern applications such as register allocation.

Explanations

1. Equivalence of Tree Definitions in Graph Theory

- **Key Points**
  - Each statement about trees (being a tree, unique path between any two vertices, connected with $n = e + 1$, and acyclic with $n = e + 1$) is equivalent.
  - The unique path property ensures there is no cycle; if two nodes had two different paths, a cycle would form.
  - A contradiction is reached by assuming G is a tree with two nodes not connected by a unique path.
- **Explanation**
  The instructor started by assuming that G is a tree and then used proof by contradiction. By considering two nodes that would supposedly have two different paths, it became clear that such a situation leads to a cycle. Labeling these nodes as 1 and 2, the presence of two distinct paths between them contradicts the

definition of a tree. Therefore, it is concluded that every two nodes are joined by a unique path, establishing the equivalence of the different statements regarding trees.

- **Cycle Formation Example**

  > The example involves nodes labeled 1 and 2, where if there exists one path from 1 to 2 and an alternative second path from 1 to 2, then traversing these forms a cycle. This cycle contradicts the tree property which requires a unique path between any two nodes, thus forming a cycle.

  i. Assume there exists an alternative second path between node 1 and node 2.
  ii. The existence of two distinct paths creates a loop or cycle.
  iii. This demonstrates by contradiction that if a graph is a tree, there can be only one unique path connecting any two nodes.

2. Inductive Proof for Trees: n = e + 1

- **Key Points**
  - Base cases: For n = 1 the result is trivial; for n = 2 there is exactly one edge; for n = 3, only one tree configuration exists where a unique path connects every pair of nodes.
  - Inductive step: Assume the result for graphs with fewer than n nodes, then partition the graph by removing an edge (X,Y) into two components G1 and G2, where X is in G1 and Y is in G2.
  - Using the induction hypothesis, if G1 has n1 nodes and E1 edges then n1 = E1 + 1, and similarly for G2, then combining the subgraphs gives n = E + 1 overall.
- **Explanation**
  The proof starts with small cases. For n = 2, there is only one edge ensuring the unique path property. For a larger graph with n nodes, the graph is divided into two components G1 and G2 by removing an edge (X,Y), where X is in G1 and Y is in G2. The induction hypothesis is applied to subgraphs that have fewer nodes, and by carefully counting the nodes and edges (including the removed edge (X,Y)), it is shown that the overall graph satisfies n = E + 1.
- **Base Case Verification for n = 2 and n = 3**

  > For n = 2, vertices v1 and v2 are connected by one edge. For n = 3, a unique spanning tree exists with three nodes where every vertex pair is connected by a unique path.

  i. n = 2: With only two vertices, only one edge can exist for a unique path.
  ii. n = 3: Any additional edge would introduce an alternative path between some vertices, violating uniqueness.

3. Tree Property: Existence of at Least Two Endpoints (Leaves)

- **Key Points**
  - Utilizes the fact that the sum of the degrees in any graph is 2e because each edge contributes to the degrees of two vertices.
  - Assumes by contradiction that in a tree all but one vertex have degree at least 2, leading to a degree sum of at least 2n - 1.
  - Since a tree must have exactly n - 1 edges, the expected sum of degrees should be exactly 2n - 2, therefore having only one vertex with degree 1 results in an inconsistency.
- **Explanation**
  The instructor explains that if a tree had only one vertex with degree 1 and all other vertices had degrees 2 or more, then the sum of the degrees would be at least (2n - 1). However, based on the degree sum formula (sum of degrees = 2e) and the fact that e = n - 1 for trees, the sum should be 2n - 2. This mismatch shows the contradiction, proving that every tree must have at least two vertices with degree 1.
- **Degree Sum Contradiction Example**
  > Consider a tree with n vertices and n - 1 edges. If only one vertex has degree 1 and the rest have at least degree 2, then the minimal degree sum would be (1 + 2*(n - 1)) = 2n - 1, which exceeds the required 2n - 2 for a tree.

  i. Calculate degree sum with one vertex of degree 1 and (n - 1) vertices of degree at least 2.
  ii. Degree sum becomes 1 + 2*(n - 1) = 2n - 1, contradicting the formula 2e = 2*(n - 1) = 2n - 2.
  iii. Thus, at least two vertices must have degree 1.

4. Euler’s Formula for Planar Graphs

- **Key Points**
  - A planar graph can be drawn without crossing edges and it splits the plane into faces.
  - A spanning tree is a subset of the graph that connects all vertices without forming cycles, giving n = e + 1 for trees.
  - By constructing a dual graph (where each face of the original graph is represented as a vertex and each edge separating two faces corresponds to an edge in the dual), it is shown that the dual spanning tree has vertices equal to the number of faces.
  - Combining counts from the spanning tree and its dual leads to the formula: n - e + f = 2.
- **Explanation**
  The instructor begins with the observation that in a tree (a cycle-free connected graph) the number of vertices is one more than the number of edges. Next, by defining a dual graph where every face of the original planar graph becomes a

vertex and each edge that separates two faces is represented as an edge, a spanning tree in this dual graph is constructed. Counting nodes and edges in both the original graph and its dual, and relating them through the structure of the spanning tree, it is deduced that n - e + f = 2, which is Euler's famous formula.

- **Rectangle Example for Euler's Formula**

  > A rectangle drawn as a planar graph has n = 4 vertices, e = 4 edges, and f = 2 faces (one inside and one outside). Substituting these into Euler's formula: 4 - 4 + 2 equals 2.

  i. Identify the number of vertices (n = 4).
  ii. Count the number of edges (e = 4).
  iii. Determine the number of faces (f = 2, comprised of the interior and the exterior).
  iv. Verify that n - e + f = 4 - 4 + 2 = 2, which is in accordance with Euler's formula.

5. Planarity and Edge Bounds in Simple Planar Graphs

- **Key Points**
  - A planar graph can be drawn on a plane such that no edges cross.
  - A planar graph splits the plane into faces based on the drawing; for example, a triangle divides the plane into two faces while more complex graphs divide it into more.
  - Using the fact that the smallest face in a simple planar graph is a triangle (requiring at least 3 edges per face), it is shown that 3f <= 2e. This, combined with Euler's formula and an assumption that every vertex has degree >= 6, leads to a contradiction, proving that every simple planar graph with n > 2 vertices must have at least one vertex of degree at most 5. The speaker also states that a similar approach can show that e <= 3n - 6.
- **Explanation**
  The instructor explains that in a simple planar graph, if every face is bounded by at least 3 edges then the total number of edge-face incidences is at least 3f. Since each edge is counted twice (once for each adjacent face), it follows that 3f ≤ 2e. By combining this with the degree conditions (for instance, if every vertex has degree ≥ 6, then e ≥ 3n) a contradiction arises when compared with Euler's formula, thereby proving that every simple planar graph with n > 2 vertices must have at least one vertex of degree at most 5. The speaker notes that a similar method can be used to establish the maximum edge count in planar graphs as e ≤ 3n - 6.
- **Contradiction via Vertex Degree Assumption**

  > Assume in a simple planar graph with n > 2 that every vertex has degree ≥ 6, which implies e ≥ 3n. However, using the fact each face must be bounded by at least 3 edges and accounting for the double-counting of edges in faces (i.e., 3f ≤ 2e), one arrives at an inequality that contradicts the combinatorial structure

> imposed by Euler's formula, demonstrating the necessity of a vertex with degree at most 5.

   i. Assume all vertices in the graph have degree ≥ 6, leading to a lower bound on e (e ≥ 3n).
   ii. Count the edges from the perspective of faces, noting that each face has at least 3 edges, but every edge is counted twice, so 3f ≤ 2e.
   iii. Combine these relations with Euler's formula to show that the assumption causes an inconsistency, hence proving there must be at least one vertex with degree ≤ 5.

6. Degree Sum Formula in Graph Theory

- **Key Points**
  - It is based on the fact that every edge contributes to the degree count of two vertices.
  - The formula explains why counting the degrees over all vertices gives 2e, regardless of the structure of the graph.
  - This property is used to derive contradictions in proofs, such as in the case of demonstrating that a tree must have at least two vertices with degree 1.
- **Explanation**
  The instructor provides a clear explanation by taking an example graph and listing the degrees of various vertices. By summing these degrees and showing that each edge is counted exactly twice (once for each of its endpoints), it is established that the overall sum of vertex degrees equals 2e. This fundamental property is instrumental in several proofs, particularly in demonstrating contradictions when the expected degree sum does not align with the assumed structure of the graph.
- **Graph Degree Sum Example**
  > An example graph is presented where one vertex has degree 3, another vertex also has degree 3, and so on. When the degrees are added up, the sum equals 2 times the number of edges present in the graph, confirming the formula.

   i. List the degrees of all vertices in the example.
   ii. Add the degrees and verify that the total equals 2e.
   iii. Show that each edge's contribution is counted twice, once for each vertex it connects.

7. Graph Coloring in Graph Theory and Compiler Register Allocation

- **Key Points**
  - Definition of graph coloring with an example of a three-coloring.
  - Explains that in a compiler, variables active at the same time are represented as connected vertices and must be assigned different registers (colors), with

available registers (e.g., 16 or 64) acting as the palette.

- **Explanation**
  The lecture first introduces graph coloring by demonstrating a graph colored with three colors, emphasizing that no two adjacent vertices have the same color. It then draws an analogy with compilers: for two variables active simultaneously (represented as connected vertices), different registers must be allocated. This forces a graph coloring problem where available registers are like colors, ensuring no overlap in assignment.
- **Three-Coloring and Register Allocation Example**

  > A graph is colored using three colors to demonstrate a valid three-coloring. In a practical scenario, a compiler uses a similar technique: constructing a graph where active variables are vertices and shared usage creates an edge. With a limited number of registers (colors), the compiler assigns colors ensuring no two active variables share the same register.

  i. The example shows a graph colored in three different colors to respect adjacent coloring.
  ii. It parallels the process used in register allocation in compilers, where the limited number of registers forces a similar allocation problem.

8. Proof by Induction for Six-Coloring Planar Graphs

- **Key Points**
  - Base case: A graph with 6 vertices can be colored with 6 distinct colors regardless of edge configuration.
  - For graphs with more than 6 vertices, remove a vertex (with degree <= 5) and apply the induction hypothesis that the remaining graph can be six-colored.
  - Reintroduce the removed vertex, which by having at most 5 neighbors leaves one available color for proper coloring.
  - Mentions further improvements: five color theorem and the four color theorem, with historical and computational significance.
- **Explanation**
  The proof starts with a base case where 6 vertices can each be assigned different colors. For larger graphs, one identifies a vertex with degree 5 or less (guaranteed by planarity), removes it, and colors the smaller graph using the induction hypothesis. When the vertex is added back, since at most 5 colors are used by its neighbors, a sixth color is available. This method is foundational in understanding why planar graphs have limited coloring requirements.
- **Planar Graph Coloring via Induction**

  > A planar graph with n vertices (n > 6) is reduced by removing a vertex of degree at most 5. The smaller, planar graph is colored with six colors, and then the removed vertex is reinserted using the sixth, previously unused color.

   i. The process of removing a vertex simplifies the graph and allows the use of inductive reasoning.
   ii. Reintroducing the vertex with at most 5 colored neighbors guarantees the existence of an available color.

9. Euler’s Formula and Graph Properties

- **Key Points**
  - Sum of vertex degrees equals 2 times the number of edges.
  - In tree graphs, the number of nodes is equal to the number of edges plus one (V = E + 1).
  - These numerical results serve as a basis for establishing Euler’s formula for planar graphs.
- **Explanation**
  The lecture shows that the sum of all vertex degrees in any graph is 2E, where E is the number of edges. For trees, a special case of graphs, V = E + 1 holds true. Such numerical relationships are fundamental in deriving Euler’s formula, which connects vertices, edges, and faces in planar graphs and assists in proving various coloring theorems.
- **Tree Structure and Degree Sum Verification**
  > In a given tree, one can verify that the total sum of vertex degrees equals twice the number of edges and that the number of vertices is one more than the number of edges. These numerical validations help solidify the foundation for Euler’s formula in planar graphs.

   i. By confirming that V = E + 1 in a tree, the relationship between vertices and edges is established.
   ii. The sum of degrees being equal to 2E provides a numerical consistency check, leading to broader applications in planar graph theory.

10. Graph Representation Techniques: Adjacency Matrix vs. Adjacency List

- **Key Points**
  - Adjacency Matrix: A 2-dimensional array where the entry Aij is 1 if there is an edge between vertex i and j, and 0 otherwise; best for dense graphs.
  - Maximum number of edges in a graph with N vertices is calculated as N*(N-1)/2.
  - The matrix representation enables techniques from linear algebra, for example, using the square of the matrix to count paths of length 2 and applying eigenvalue analysis (e.g., Google’s PageRank).
  - Adjacency List: Stores a list of neighbors for each vertex; efficient for sparse graphs with many zeros in the matrix representation.

- **Explanation**
  The lecture outlines that an adjacency matrix is useful when the graph is dense, as every possible pair of vertices is checked for an edge. The diagonal and powers of this matrix reveal path counts between vertices, which can be harnessed for further analyses like eigenvalue computations. In contrast, adjacency lists are more memory-efficient for sparse graphs where storing a full N by N matrix would waste space.
- **Adjacency Matrix and List Implementation Example**
  > A graph is represented in two ways: as an adjacency matrix (an N by N array with entries marked 1 or 0) offering straightforward linear algebra operations, and as an adjacency list (a list for each vertex detailing its connections) which is preferable when the graph has comparatively few edges.

  i. The matrix is ideal for dense graphs because it directly shows connections and supports operations like matrix squaring to count paths.
  ii. For sparse graphs, the adjacency list is more efficient in terms of memory and processing.

11. Historical and Practical Aspects of Graph Coloring

- **Key Points**
  - Map-making in historical contexts required a minimal number of colors due to the high cost and scarcity of colored inks and dyes, such as indigo and purple.
  - The problem of coloring maps with the fewest colors led to the study of graph coloring and the eventual formulation of the four color theorem.
  - Modern applications include register allocation in compilers.
  - Historical significance: The four color theorem was the first major theorem to be proved using computer assistance, reducing the problem to between 800 and 1000 cases in the late 1960s.
- **Explanation**
  The lecture connects historical map-making, where the cost of colored inks made it imperative to minimize the number of colors used, to the development of graph coloring theories. This need drove the investigation into whether four colors would always suffice, culminating in the computer-assisted proof of the four color theorem. Such investigations have modern parallels in computer science, particularly in compiler design.
- **Historical Map Coloring Example**
  > The example details how, 200 years ago, map makers and royalty had to use a minimal set of colors due to the high cost of expensive dyes such as indigo and purple. This practical problem led to a theoretical challenge of proving that only four colors are sufficient to color any given map, eventually giving rise to the four color theorem.

   i. The scarcity and high cost of certain dyes meant that minimizing the number of colors was economically important.
   ii. The challenge of reducing color usage in maps was formalized into a graph coloring problem, providing historical context for modern computational problems.

Homework

[ ] Investigate register allocation using graph coloring through the programming assignment provided next session.
[ ] Review the inductive proof steps for planar graph coloring and prepare questions for class discussion on graph reduction and planarity preservation.
[ ] Solve practice problems involving Euler's formula, tree structures, and degree sums in various types of graphs.
[ ] Implement both adjacency matrix and adjacency list representations in a programming assignment, and compare their performances for different types of graphs.
[ ] Research the computer-assisted proof of the four color theorem and discuss its impact on modern computational methods in graph theory.

> **AI Suggestion**
>
> - The core of this lesson is understanding tree properties and Euler's formula for planar graphs. It's recommended to start with an induction exercise on trees—prove that for any tree with unique path properties, n = e + 1—to deeply internalize the connection between acyclic, connected graphs and their numerical characteristics.
>
> - Core content of Tree Properties and Euler's Formula: Explains the interrelationship between various definitions of a tree (unique path, acyclic, and connected) and demonstrates by induction that n = e + 1. It further details how spanning trees and dual graphs lead to Euler's formula for planar graphs (n − e + f = 2) and establishes edge bounds in simple planar graphs.
>
> - The core of this lesson is applying graph coloring techniques to real-world problems such as compiler register allocation. It's recommended to practice by working through exercises that map variables to registers using graph coloring principles to see how adjacent elements are assigned unique colors.
>
> - Core content of Graph Coloring and Compiler Register Allocation: Describes the concept of assigning colors to vertices so that no two adjacent vertices share the same color, including an inductive proof for six-coloring planar graphs, and explains its practical application in efficiently allocating registers in compilers.
>
> Additionally, here are some extracurricular resources:

> - Practical application of Tree Properties and Euler’s Formula: https://en.wikipedia.org/wiki/Planar_graph
> - Alternative perspective on Graph Coloring: https://en.wikipedia.org/wiki/Graph_coloring