Lecture 3: Trees

Shuai Li

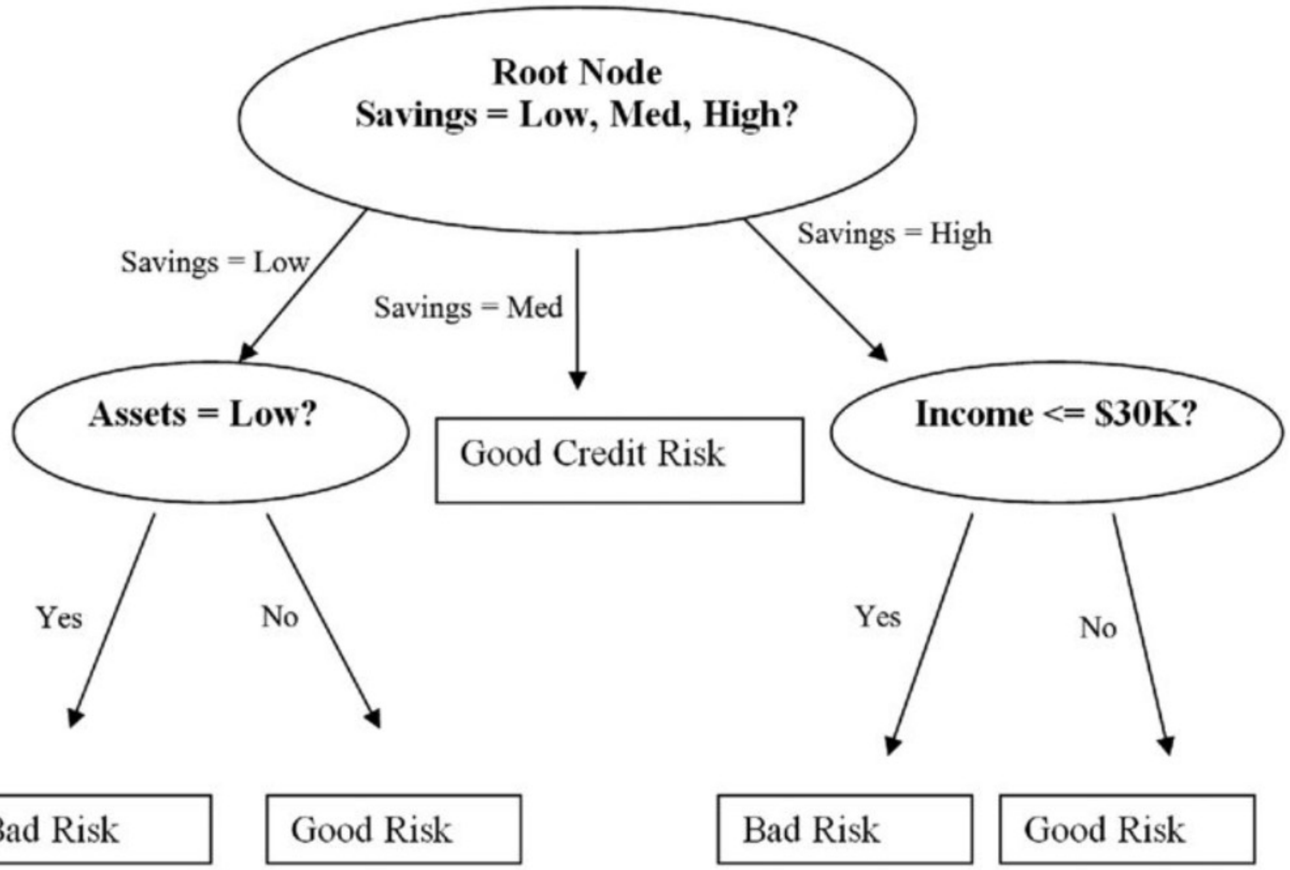
John Hopcroft Center, Shanghai Jiao Tong University

https://shuaili8.github.io

https://shuaili8.github.io/Teaching/CS3330/index.html

Trees

- A tree is a connected graph T with no cycles



Properties

- Recall that **Theorem** (1.2.18, W, Kőnig 1936)
  A graph is bipartite $\Leftrightarrow$ it contains no odd cycle
- $\Rightarrow$(Ex 3, S1.3.1, H) A tree of order $n \geq 2$ is a bipartite graph
- Recall that **Proposition** (1.2.14, W)
  An edge e is a bridge $\Leftrightarrow$ e lies on no cycle of G
  - Or equivalently, an edge e is not a bridge $\Leftrightarrow$ e lies on a cycle of G
- $\Rightarrow$ Every edge in a tree is a bridge
- T is a tree $\Leftrightarrow$ T is minimally connected, i.e. T is connected but $T - e$ is disconnected for every edge $e \in T$

Equivalent definitions (Theorem 1.5.1, D)

- T is a tree of order n
  $\Leftrightarrow$ Any two vertices of T are linked by a unique path in T
  $\Leftrightarrow$ T is minimally connected
  - i.e. T is connected but $T - e$ is disconnected for every edge $e \in T$

  $\Leftrightarrow$ T is maximally acyclic
  - i.e. T contains no cycle but $T + xy$ does for any non-adjacent vertices $x, y \in T$

  $\Leftrightarrow$ (Theorem 1.10, 1.12, H) T is connected with $n - 1$ edges

  $\Leftrightarrow$ (Theorem 1.13, H) T is acyclic with $n - 1$ edges

Leaves of tree

- A vertex of degree 1 in a tree is called a leaf
- Theorem (1.14, H; Ex9, S1.3.2, H) Let T be a tree of order $n \geq 2$. Then T has at least two leaves
- (Ex3, S1.3.2, H) Let T be a tree with max degree Δ. Then T has at least Δ leaves
- (Ex10, S1.3.2, H) Let T be a tree of order $n \geq 2$. Then the number of leaves is
$$2 + \sum_{v:d(v)\geq 3} (d(v) - 2)$$
- (Ex8, S1.3.2, H) Every nonleaf in a tree is a cut vertex
- Every leaf node is not a cut vertex

The center of a tree is a vertex or ‘an edge’

- Theorem (1.15, H) In any tree, the center is either a single vertex or a pair of adjacent vertices

Any tree can be embedded in a ‘dense’ graph

- Theorem (1.16, H) Let T be a tree of order $k+1$ with k edges. Let G be a graph with $\delta(G) \geq k$. Then G contains T as a subgraph

Spanning tree

- Given a graph G and a subgraph T, T is a spanning tree of G if T is a tree that contains every vertex of G
- Example: A telecommunications company tries to lay cable in a new neighbourhood
- Proposition (2.1.5c, W) Every connected graph contains a spanning tree

Minimal spanning tree - Kruskal’s Algorithm

- Given: A connected, weighted graph G

1. Find an edge of minimum weight and mark it.
2. Among all of the unmarked edges that do not form a cycle with any of the marked edges, choose an edge of minimum weight and mark it
3. If the set of marked edges forms a spanning tree of G, then stop. If not, repeat step 2

Example

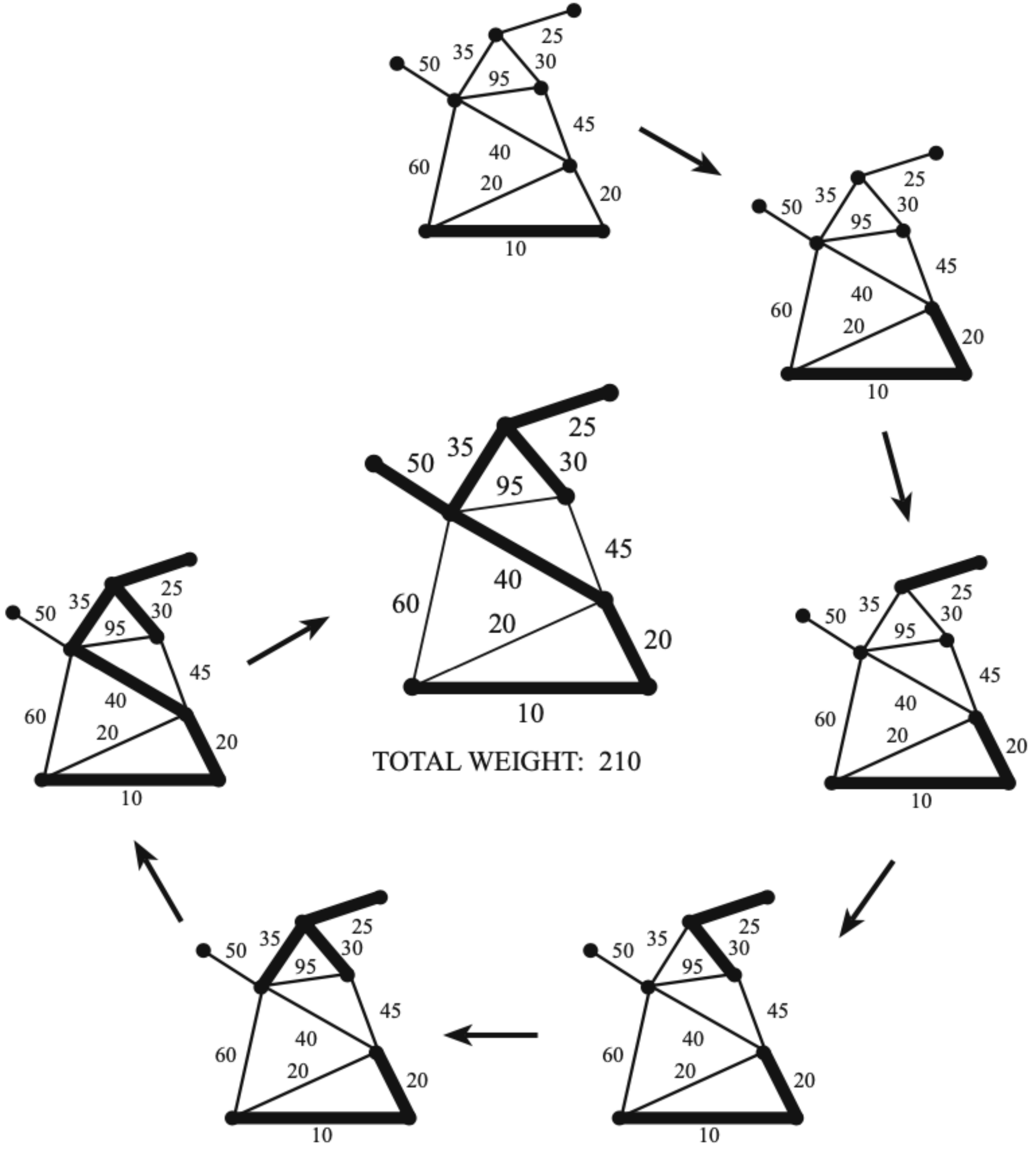


FIGURE 1.43. The stages of Kruskal's algorithm.

Theoretical guarantee of Kruskal's algorithm

- Theorem (1.17, H) Kruskal's algorithm produces a spanning tree of minimum total weight

Prim’s Algorithm

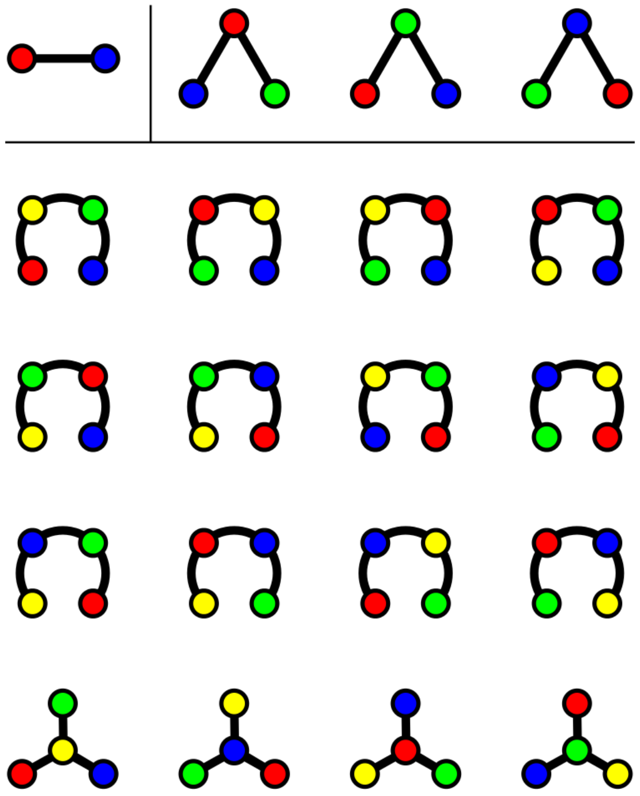
- Given: A connected, weighted graph G.

1. Choose a vertex v, and mark it.
2. From among all edges that have one marked end vertex and one unmarked end vertex, choose an edge e of minimum weight. Mark the edge e, and also mark its unmarked end vertex.
3. If every vertex of G is marked, then the set of marked edges forms a minimum weight spanning tree. If not, repeat step 2

- Exercise (Ex2.3.10, W) Prim’s algorithm produces a minimum-weight spanning tree of G

Cayley's tree formula

- Theorem (1.18, H; 2.2.3, W). There are n^{n-2} distinct labeled trees of order n



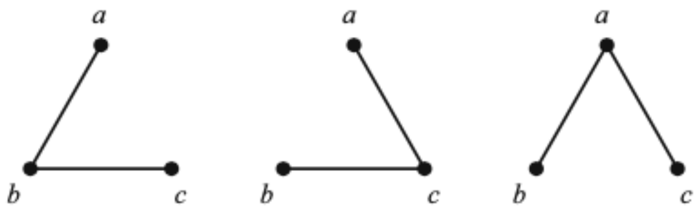


FIGURE 1.45. Labeled trees on three vertices.

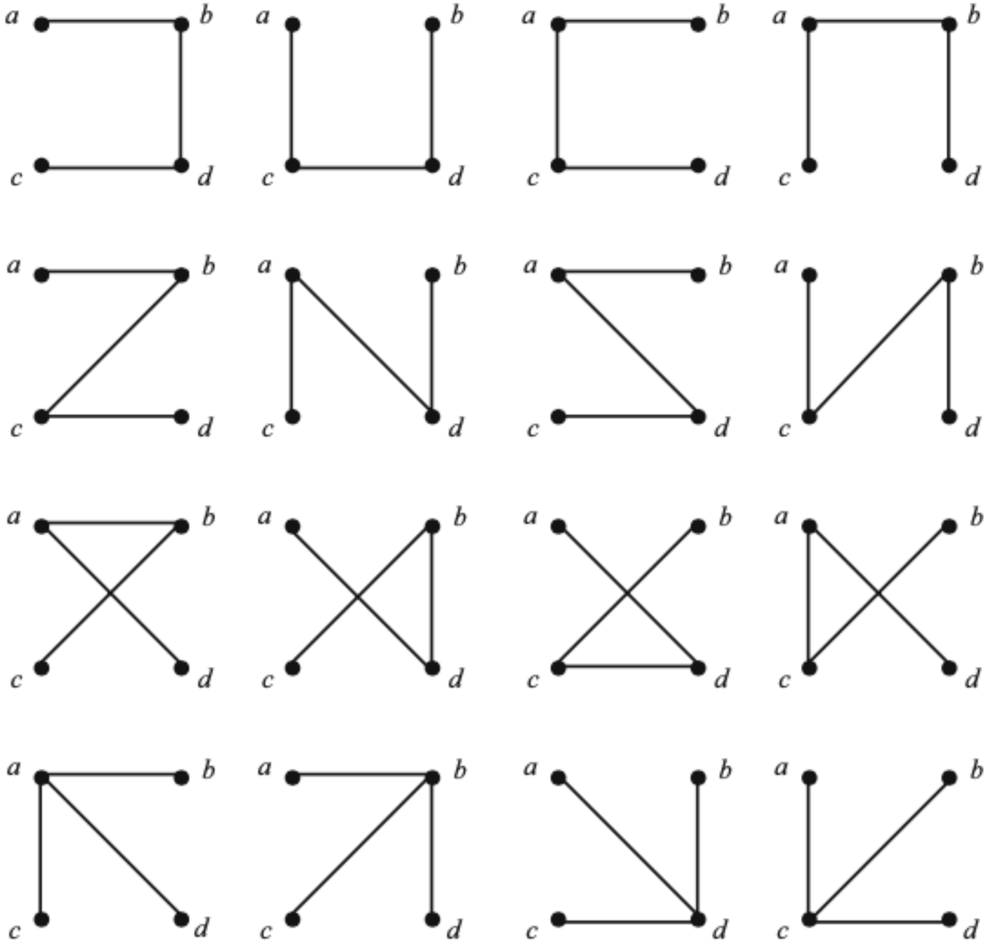


FIGURE 1.46. Labeled trees on four vertices.

Example

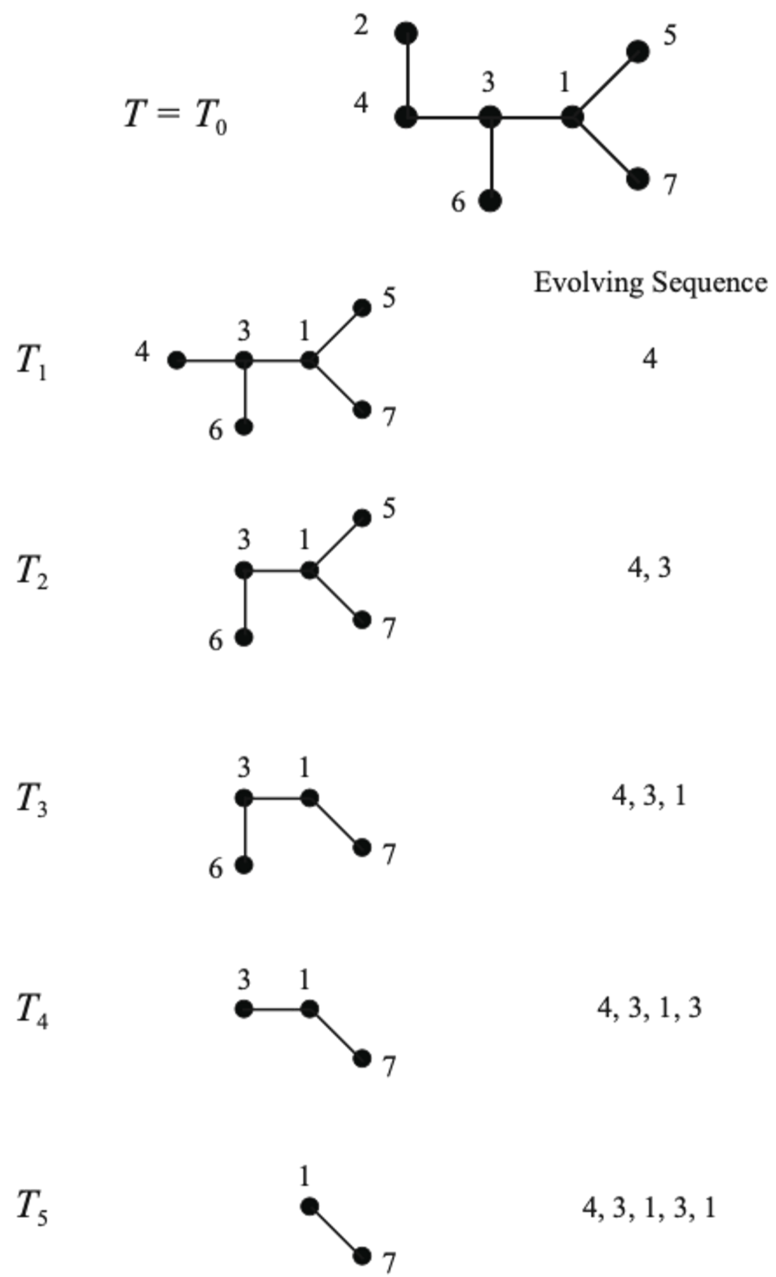


FIGURE 1.47. Creating a Prüfer sequence.

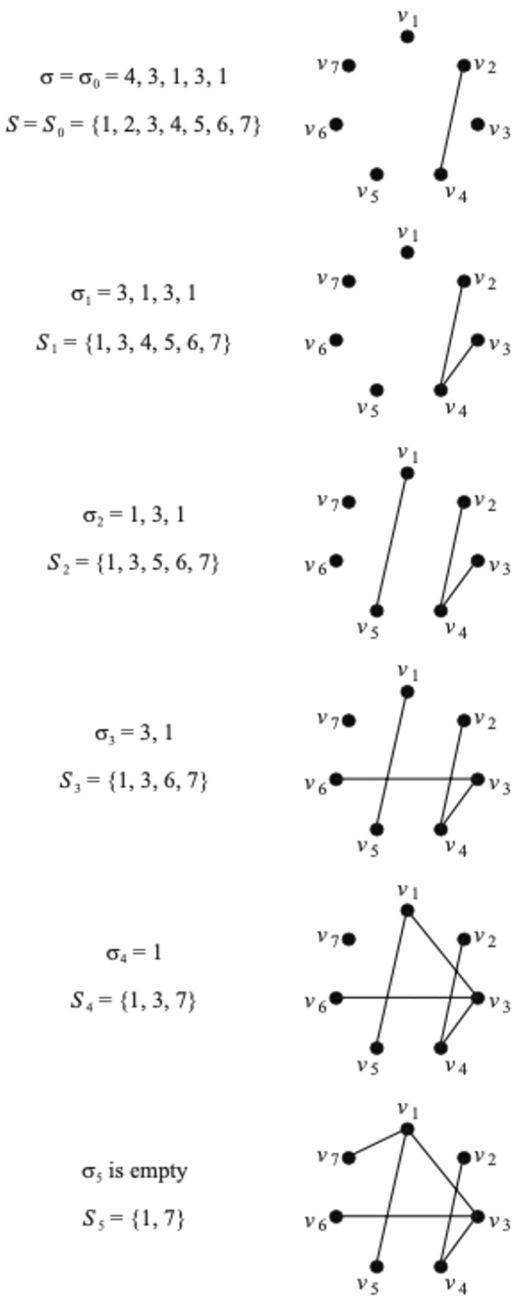
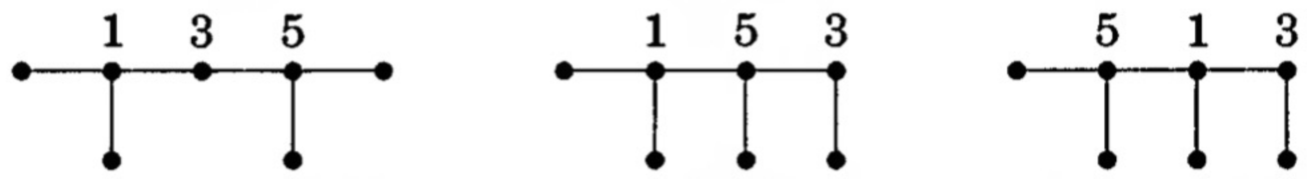


FIGURE 1.48. Building a labeled tree.

of trees with fixed degree sequence

- Corollary (2.2.4, W) Given positive integers $d_1, \dots, d_n$ summing to $2n-2$, there are exactly $\frac{(n-2)!}{\prod(d_i-1)!}$ trees with vertex set $[n]$ such that vertex i has degree d_i for each i

- Example (2.2.5, W) Consider trees with vertices $[7]$ that have degrees $(3,1,2,1,3,1,1)$



Matrix tree theorem - cofactor

- For an $n \times n$ matrix A, the i, j cofactor of A is defined to be

$$(-1)^{i+j} \det(M_{ij})$$

where M_{ij} represents the $(n-1) \times (n-1)$ matrix formed by deleting row i and column j from A

3 × 3 generic matrix [edit]

Consider a 3×3 matrix

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}.$$

Its cofactor matrix is

$$\mathbf{C} = \begin{pmatrix} +\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} \\ -\begin{vmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{12} \\ a_{31} & a_{32} \end{vmatrix} \\ +\begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \end{pmatrix},$$

Matrix tree theorem

- Theorem (1.19, H; 2.2.12, W; Kirchhoff) If G is a connected labeled graph with adjacency matrix A and degree matrix D, then the number of unique spanning trees of G is equal to the value of any cofactor of the matrix $D - A$
- If the row sums and column sums of a matrix are all 0, then the cofactors all have the same value
- Exercise Read the proof
- Exercise (Ex7, S1.3.4, H) Use the matrix tree theorem to prove Cayley's theorem

Example

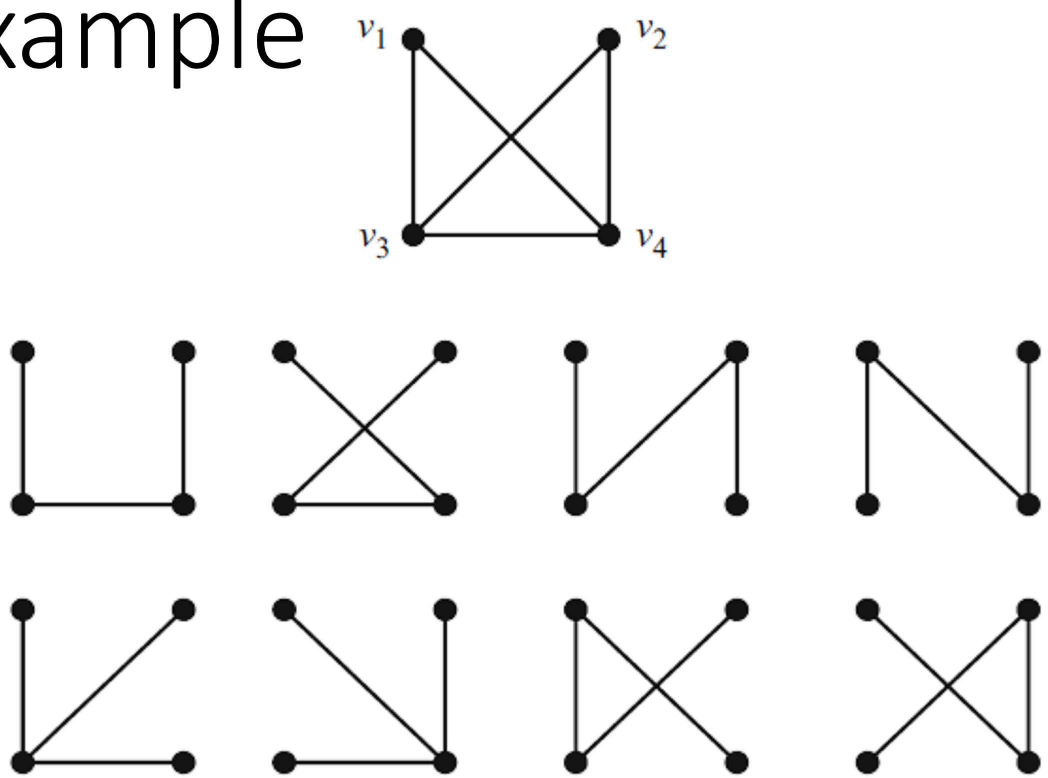


FIGURE 1.49. A labeled graph and its spanning trees.

The degree matrix D and adjacency matrix A are

$$D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix},$$

and so

$$D - A = \begin{bmatrix} 2 & 0 & -1 & -1 \\ 0 & 2 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix}.$$

The $(1, 1)$ cofactor of $D - A$ is

$$\det \begin{bmatrix} 2 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{bmatrix} = 8.$$

Score one for Kirchhoff!

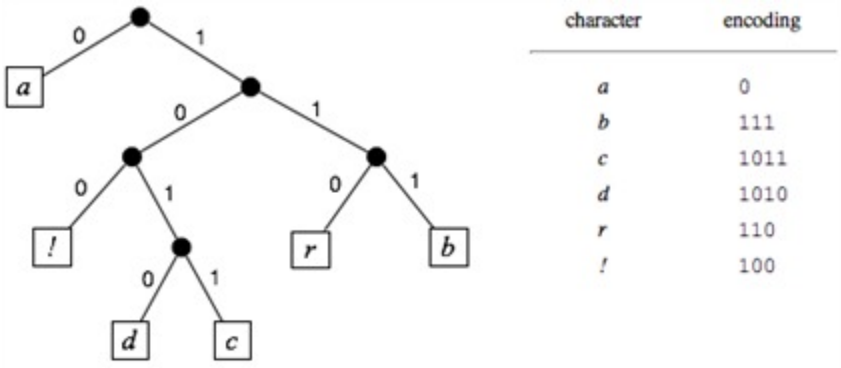
- Exercise (Ex6, S1.3.4, H) Let e be an edge of K_n. Use Cayley's Theorem to prove that $K_n - e$ has $(n-2)n^{n-3}$ spanning trees

Wiener index

- In a communication network, large diameter may be acceptable if most pairs can communicate via short paths. This leads us to study the average distance instead of the maximum
- Wiener index $D(G) = \sum_{u,v \in V(G)} d_G(u, v)$
- Theorem (2.1.14, W) Among trees with n vertices, the Wiener index $D(T)$ is minimized by stars and maximized by paths, both uniquely
- Over all connected n-vertex graphs, $D(G)$ is minimized by K_n and maximized (2.1.16, W) by paths
  - (Lemma 2.1.15, W) If H is a subgraph of G, then $d_G(u, v) \leq d_H(u, v)$

Prefix coding

- A binary tree is a rooted plane tree where each vertex has at most two children
- Given large computer files and limited storage, we want to encode characters as binary lists to minimize (expected) total length
- Prefix-free coding: no code word is an initial portion of another



A binary prefix code for the alphabet $\{a, b, c, d, r, !\}$

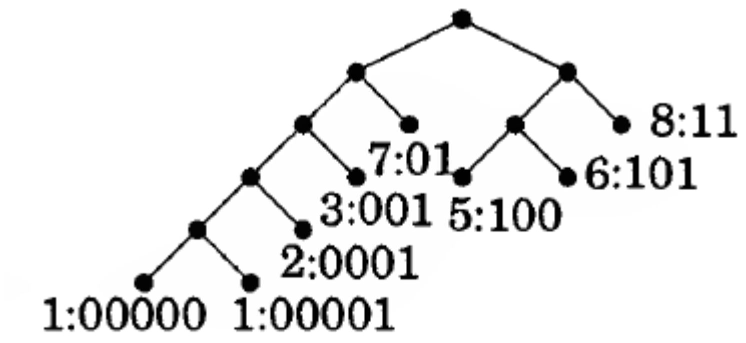
- Example: 11001111011

Huffman's Algorithm (2.3.13, W)

- Input: Weights (frequencies or probabilities) $p_1, \ldots, p_n$
- Output: Prefix-free code (equivalently, a binary tree)
- Idea: Infrequent items should have longer codes; put infrequent items deeper by combining them into parent nodes.
- Recursion: replace the two least likely items with probabilities p, p' with a single item of weight $p + p'$

Example (2.3.14, W)

a	5	100
b	1	00000
c	1	00001
d	7	01
e	8	11
f	2	0001
g	3	001
h	6	101



The average length is $\frac{5\times3+5+5+7\times2+\cdots}{33}=\frac{30}{11}<3$

Huffman coding is optimal

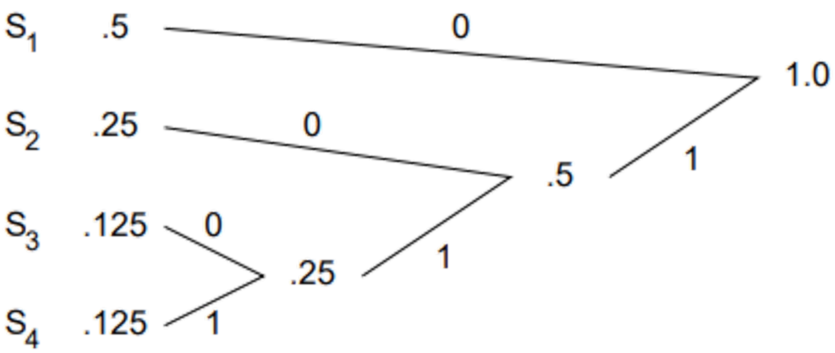
- Theorem (2.3.15, W) Given a probability distribution $\{p_i\}$ on n items, Huffman's Algorithm produces the prefix-free code with minimum expected length

Huffman coding and entropy

- The entropy of a discrete probability distribution $\{p_i\}$ is that

$$H(p) = -\sum_i p_i \log_2 p_i$$

- Exercise (Ex2.3.31, W) $H(p) \leq$ average length of Huffman coding $\leq H(p) + 1$
- Exercise (Ex2.3.30, W) When each p_i is a power of ½, average length of Huffman coding is $H(p)$



Codewords
0
10
110
111

$$\text{average length} = (1)\left(\frac{1}{2}\right) + (2)\left(\frac{1}{4}\right) + (3)\left(\frac{1}{8}\right) + (3)\left(\frac{1}{8}\right) = 1.75 \text{ bits/symbol}$$

$$H = \frac{1}{2}\log_2 2 + \frac{1}{4}\log_2 4 + \frac{1}{8}\log_2 8 + \frac{1}{8}\log_2 8 = \frac{1}{2} + \frac{1}{2} + \frac{3}{8} + \frac{3}{8} = 1.75$$

Summary

Shuai Li

https://shuaili8.github.io

- Trees
  - Is bipartite, edges are bridge; Equivalent definitions
  - Leaves, # of leaves, cut vertex; Center is a vertex or `an edge’
  - Any tree can be embedded in a ‘dense’ graph
- Spanning Tree
  - Every connected graph has a spanning tree
  - Minimal spanning tree, Kruskal’s Algorithm (with guarantee), Prim’s algorithm
  - Cayley’s tree formula, Prüfer code, # of trees with fixed degree sequence
  - Matrix tree theorem
- Wiener index
  - Among trees, minimized by starts, maximized by paths
  - Among connected graphs, minimized by complete graphs, maximized by paths
- Hoffman coding
  - Algorithm, optimality, entropy

Questions?