

Discrete Mathematics

Trees

H. Turgut Uyar Ayşegül Gençata Yayimli Emre Harmancı

2001-2014

1 / 57

License



© 2001-2014 T. Uyar, A. Yayimli, E. Harmancı

You are free to:

- ▶ Share – copy and redistribute the material in any medium or format
- ▶ Adapt – remix, transform, and build upon the material

Under the following terms:

- ▶ Attribution – You must give appropriate credit, provide a link to the license, and indicate if changes were made.
- ▶ NonCommercial – You may not use the material for commercial purposes.
- ▶ ShareAlike – If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

For more information:

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Read the full license:

<https://creativecommons.org/licenses/by-nc-sa/4.0/legalcode>

2 / 57

Topics

Trees

- Introduction
- Rooted Trees
- Binary Trees
- Decision Trees

Tree Problems

- Minimum Spanning Tree

3 / 57

Tree

Definition

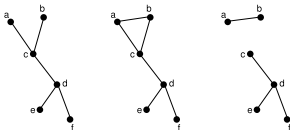
tree: a connected graph that contains no cycle

- ▶ **forest:** a graph where the connected components are trees

4 / 57

Tree Examples

Example



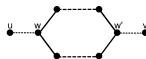
5 / 57

Tree Theorems

Theorem

In a tree, there is one and only one path between any two distinct nodes.

- ▶ there is at least one path because the tree is connected
- ▶ if there were more than one path, they would form a cycle



6 / 57

Tree Theorems

Theorem

let $T = (V, E)$ be a tree:

$$|E| = |V| - 1$$

- ▶ proof method: induction on the number of edges

7 / 57

Tree Theorems

Proof: Base step.

- ▶ $|E| = 0 \Rightarrow |V| = 1$
- ▶ $|E| = 1 \Rightarrow |V| = 2$
- ▶ $|E| = 2 \Rightarrow |V| = 3$
- ▶ assume that $|E| = |V| - 1$ for $|E| \leq k$

8 / 57

Tree Theorems

Proof: Induction step.

► $|E| = k + 1$



- let's remove the edge (y, z) :
 $T_1 = (V_1, E_1)$, $T_2 = (V_2, E_2)$

$$\begin{aligned} |V| &= |V_1| + |V_2| \\ &= |E_1| + 1 + |E_2| + 1 \\ &= (|E_1| + |E_2| + 1) + 1 \\ &= |E| + 1 \end{aligned}$$

□

9 / 57

Tree Theorems

Theorem

In a tree, there are at least two nodes with degree 1.

Proof.

- $2|E| = \sum_{v \in V} d_v$
 ► assume that there is only 1 node with degree 1:
 $\Rightarrow 2|E| \geq 2(|V| - 1) + 1$
 $\Rightarrow 2|E| \geq 2|V| - 1$
 $\Rightarrow |E| \geq |V| - \frac{1}{2} > |V| - 1$

□

10 / 57

Tree Theorems

Theorem

T is a tree (T is connected and contains no cycle).

\Leftrightarrow

*There is one and only one path
 between any two distinct nodes in T .*

\Leftrightarrow

*T is connected, but if any edge is removed
 it will no longer be connected.*

\Leftrightarrow

*T contains no cycle, but if an edge is added
 between any pair of nodes one and only one cycle will be formed.*

11 / 57

Tree Theorems

Theorem

T is a tree (T is connected and contains no cycle).

\Leftrightarrow

T is connected and $|E| = |V| - 1$.

\Leftrightarrow

T contains no cycle and $|E| = |V| - 1$.

12 / 57

Rooted Tree

- ▶ a hierarchy is defined between nodes
- ▶ hierarchy creates a natural direction on edges:
in and out degrees
- ▶ in-degree 0: **root** (only 1 such node)
- ▶ out-degree 0: **leaf**
- ▶ not a leaf: **internal** node

13 / 57

Node Level

Definition

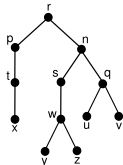
level of a node: distance from the root

- ▶ **parent**: adjacent node in the next upper level (only 1 such node)
- ▶ **child**: adjacent node in the next lower level
- ▶ **sibling**: node which has the same parent

14 / 57

Rooted Tree Example

Example

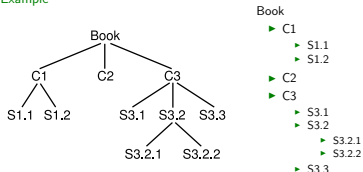


- ▶ root: *r*
- ▶ leaves: *x y z u v*
- ▶ internal nodes: *r p n t s q w*
- ▶ parent of *y*: *w*
- ▶ children of *w*: *y* and *z*
- ▶ *y* and *z* are siblings

15 / 57

Rooted Tree Example

Example



16 / 57

Ordered Rooted Tree

- ▶ sibling nodes are ordered from left to right
- ▶ **universal address system**
 - ▶ assign the address 0 to the root
 - ▶ assign the positive integers $1, 2, 3, \dots$ to the nodes at level 1, from left to right
 - ▶ let v be an internal node with address a , assign the addresses $a.1, a.2, a.3, \dots$ to the children of v from left to right

17 / 57

Lexicographic Order

Definition

Let b and c be two addresses.

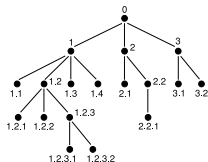
b comes before c if one of the following holds:

1. $b = a_1 a_2 \dots a_m x_1 \dots$
 $c = a_1 a_2 \dots a_m x_2 \dots$
 x_1 comes before x_2
2. $b = a_1 a_2 \dots a_m$
 $c = a_1 a_2 \dots a_m a_{m+1} \dots$

18 / 57

Lexicographic Order Example

Example



- ▶ 0 - 1 - 1.1 - 1.2
- 1.2.1 - 1.2.2 - 1.2.3
- 1.2.3.1 - 1.2.3.2
- 1.3 - 1.4 - 2
- 2.1 - 2.2 - 2.2.1
- 3 - 3.1 - 3.2

19 / 57

Binary Trees

Definition

$T = (V, E)$ is a **binary tree**:

$\forall v \in V \ d_v^o \in \{0, 1, 2\}$

- ▶ $T = (V, E)$ is a **complete** binary tree:
 $\forall v \in V \ d_v^o \in \{0, 2\}$

20 / 57

Expression Tree

- ▶ a binary operation can be represented as a binary tree
 - ▶ operator as the root, operands as the children
- ▶ every mathematical expression can be represented as a tree
 - ▶ operators at internal nodes, variables and values at the leaves

21 / 57

Expression Tree Examples

Example $(7 - a)$



Example $(a + b)$



22 / 57

Expression Tree Examples

Example $((7 - a)/5)$



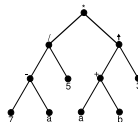
Example $((a + b) \uparrow 3)$



23 / 57

Expression Tree Examples

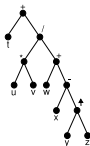
Example $((((7 - a)/5) * ((a + b) \uparrow 3))$



24 / 57

Expression Tree Examples

Example $(t + (u * v) / (w + x - y \uparrow z))$



25 / 57

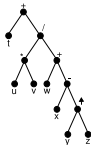
Expression Tree Traversals

1. **inorder** traversal:
traverse left subtree, visit root, traverse right subtree
2. **preorder** traversal:
visit root, traverse left subtree, traverse right subtree
3. **postorder** traversal:
traverse left subtree, traverse right subtree, visit root
 ▶ *reverse Polish notation*

26 / 57

Inorder Traversal Example

Example

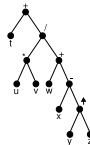


$t + u * v / w + x - y \uparrow z$

27 / 57

Preorder Traversal Example

Example

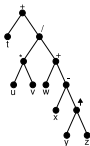


$+ t / * u v + w - x \uparrow y z$

28 / 57

Postorder Traversal Example

Example



*t u v * w x y z ↑ - + / +*

Expression Tree Evaluation

- ▶ inorder traversal requires parentheses for precedence
- ▶ preorder and postorder traversals do not require parentheses

Postorder Evaluation Example

Example (t u v * w x y z ↑ - + / +)

4 2 3 * 1 9 2 3 ↑ - + / +

4	2	3	*			
4	6	1	9	2	3	↑
4	6	1	9	8	-	
4	6	1	1	+		
4	6	2	/			
4	3	+				
7						

Regular Trees

Definition

$T = (V, E)$ is an **m-ary tree**: $\forall v \in V \ d_v^o \leq m$

- ▶ $T = (V, E)$ is a complete m -ary tree:
 $\forall v \in V \ d_v^o \in \{0, m\}$

Regular Tree Theorem

Theorem

Let $T = (V, E)$ be a complete m -ary tree.

- ▶ n : number of nodes
- ▶ l : number of leaves
- ▶ i : number of internal nodes

Then:

- ▶ $n = m \cdot i + 1$
- ▶ $l = n - i = m \cdot i + 1 - i = (m - 1) \cdot i + 1$

$$i = \frac{l - 1}{m - 1}$$

33 / 57

Regular Tree Examples

Example

- ▶ how many matches are played in a tennis tournament with 27 players?
- ▶ every player is a leaf: $l = 27$
- ▶ every match is an internal node: $m = 2$
- ▶ number of matches: $i = \frac{l-1}{m-1} = \frac{27-1}{2-1} = 26$

34 / 57

Regular Tree Examples

Example

- ▶ how many extension cords with 4 outlets are required to connect 25 computers to a wall socket?
- ▶ every computer is a leaf: $l = 25$
- ▶ every extension cord is an internal node: $m = 4$
- ▶ number of cords: $i = \frac{l-1}{m-1} = \frac{25-1}{4-1} = 8$

35 / 57

Decision Trees

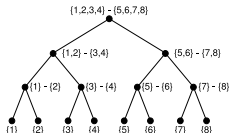
Example

- ▶ one of 8 coins is counterfeit (it's heavier)
- ▶ find the counterfeit coin using a beam balance

36 / 57

Decision Trees

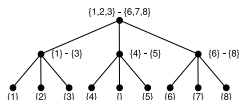
Example (in 3 weighings)



37 / 57

Decision Trees

Example (in 2 weighings)



38 / 57

Spanning Tree

- T is a **spanning tree** of G :
 T is a subgraph of G such that T is a tree and contains all the nodes of G

Definition

MST is a **minimum spanning tree** of G :

MST is a spanning tree of G such that the total weight of the edges in MST is minimal

39 / 57

Kruskal's Algorithm

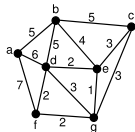
Kruskal's algorithm

1. $i \leftarrow 1$, $e_1 \in E$, $wt(e_1)$ is minimal
2. for $1 \leq i \leq n-2$:
 the selected edges are e_1, e_2, \dots, e_i
 select a new edge e_{i+1} from the remaining edges such that:
 - $wt(e_{i+1})$ is minimal
 - $e_1, e_2, \dots, e_i, e_{i+1}$ contains no cycle
3. $i \leftarrow i + 1$
 - $i = n-1 \Rightarrow$ the subgraph G containing the edges e_1, e_2, \dots, e_{n-1} is a minimum spanning tree
 - $i < n-1 \Rightarrow$ go to step 2

40 / 57

Kruskal's Algorithm Example

Example (initialization)

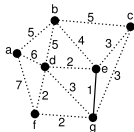


- ▶ $i \leftarrow 1$
- ▶ minimum weight: 1
(e, g)
- ▶ $T = \{(e, g)\}$

41 / 57

Kruskal's Algorithm Example

Example ($1 < 6$)

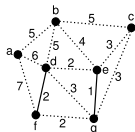


- ▶ minimum weight: 2
(d, e), (d, f), (f, g)
- ▶ $T = \{(e, g), (d, f)\}$
- ▶ $i \leftarrow 2$

42 / 57

Kruskal's Algorithm Example

Example ($2 < 6$)

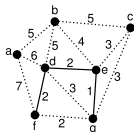


- ▶ minimum weight: 2
(d, e), (f, g)
- ▶ $T = \{(e, g), (d, f), (d, e)\}$
- ▶ $i \leftarrow 3$

43 / 57

Kruskal's Algorithm Example

Example ($3 < 6$)

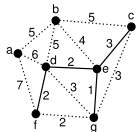


- ▶ minimum weight: 2
(f, g) forms a cycle
- ▶ minimum weight: 3
(c, e), (c, g), (d, g)
(d, g) forms a cycle
- ▶ $T = \{(e, g), (d, f), (d, e), (c, e)\}$
- ▶ $i \leftarrow 4$

44 / 57

Kruskal's Algorithm Example

Example ($4 < 6$)

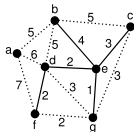


- ▶ $T = \{$
 $(e, g), (d, f), (d, e),$
 $(c, e), (b, e)$
 $\}$
- ▶ $i \leftarrow 5$

45 / 57

Kruskal's Algorithm Example

Example ($5 < 6$)

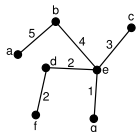


- ▶ $T = \{$
 $(e, g), (d, f), (d, e),$
 $(c, e), (b, e), (a, b)$
 $\}$
- ▶ $i \leftarrow 6$

46 / 57

Kruskal's Algorithm Example

Example ($6 \not< 6$)



- ▶ total weight: 17

47 / 57

Prim's Algorithm

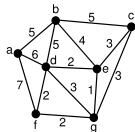
Prim's algorithm

1. $i \leftarrow 1, v_1 \in V, P = \{v_1\}, N = V - \{v_1\}, T = \emptyset$
2. for $1 \leq i \leq n-1$:
 $P = \{v_1, v_2, \dots, v_i\}, T = \{e_1, e_2, \dots, e_{i-1}\}, N = V - P$
 select a node $v_{i+1} \in N$ such that for a node $x \in P$
 $e = (x, v_{i+1}) \notin T, wt(e)$ is minimal
 $P \leftarrow P + \{v_{i+1}\}, N \leftarrow N - \{v_{i+1}\}, T \leftarrow T + \{e\}$
3. $i \leftarrow i + 1$
 - ▶ $i = n \Rightarrow$ the subgraph G containing the edges e_1, e_2, \dots, e_{n-1} is a minimum spanning tree
 - ▶ $i < n \Rightarrow$ go to step 2

48 / 57

Prim's Algorithm Example

Example (initialization)

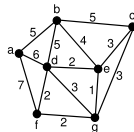


- ▶ $i \leftarrow 1$
- ▶ $P = \{a\}$
- ▶ $N = \{b, c, d, e, f, g\}$
- ▶ $T = \emptyset$

40 / 57

Prim's Algorithm Example

Example ($1 < 7$)

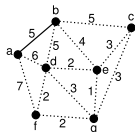


- ▶ $T = \{(a, b)\}$
- ▶ $P = \{a, b\}$
- ▶ $N = \{c, d, e, f, g\}$
- ▶ $i \leftarrow 2$

50 / 57

Prim's Algorithm Example

Example ($2 < 7$)

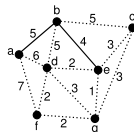


- ▶ $T = \{(a, b), (b, e)\}$
- ▶ $P = \{a, b, e\}$
- ▶ $N = \{c, d, f, g\}$
- ▶ $i \leftarrow 3$

51 / 57

Prim's Algorithm Example

Example ($3 < 7$)

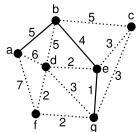


- ▶ $T = \{(a, b), (b, e), (e, g)\}$
- ▶ $P = \{a, b, e, g\}$
- ▶ $N = \{c, d, f\}$
- ▶ $i \leftarrow 4$

52 / 57

Prim's Algorithm Example

Example ($4 < 7$)

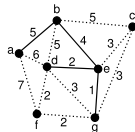


- ▶ $T = \{(a, b), (b, e), (e, g), (d, e)\}$
- ▶ $P = \{a, b, e, g, d\}$
- ▶ $N = \{c, f\}$
- ▶ $i \leftarrow 5$

53 / 57

Prim's Algorithm Example

Example ($5 < 7$)

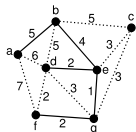


- ▶ $T = \{(a, b), (b, e), (e, g), (d, e), (f, g)\}$
- ▶ $P = \{a, b, e, g, d, f\}$
- ▶ $N = \{c\}$
- ▶ $i \leftarrow 6$

54 / 57

Prim's Algorithm Example

Example ($6 < 7$)

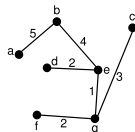


- ▶ $T = \{(a, b), (b, e), (e, g), (d, e), (f, g), (c, g)\}$
- ▶ $P = \{a, b, e, g, d, f, c\}$
- ▶ $N = \emptyset$
- ▶ $i \leftarrow 7$

55 / 57

Prim's Algorithm Example

Example ($7 \nless 7$)



- ▶ total weight: 17

56 / 57

References

Required Reading: Grimaldi

- ▶ Chapter 12: Trees
 - ▶ 12.1. Definitions and Examples
 - ▶ 12.2. Rooted Trees
- ▶ Chapter 13: Optimization and Matching
 - ▶ 13.2. Minimal Spanning Trees:
The Algorithms of Kruskal and Prim