Discrete Mathematics

Graphs

H. Turgut Uyar   Ayşegül Gençata Yayımlı   Emre Harmancı

2001-2011

License

Topics

Graphs

Definition

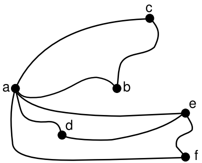
graph: $G = (V, E)$

- V: node (or *vertex*) set
- $E \subseteq V \times V$: edge set

- if $e = (v_1, v_2) \in E$:
  - v_1 and v_2 are *endnodes* of e
  - e is *incident* to v_1 and v_2
  - v_1 and v_2 are *adjacent*
- node with no incident edge: *isolated node*

Graph Example

Example



$$
\begin{aligned}
V &= \{a, b, c, d, e, f\} \\
E &= \{(a, b), (a, c), \\
&\quad (a, d), (a, e), \\
&\quad (a, f), (b, c), \\
&\quad (d, e), (e, f)\}
\end{aligned}
$$

Directed Graphs

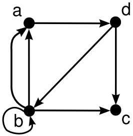
Definition

directed graph (or *digraph*): edges have directions

- directed edge: arc
- origin and terminating nodes

Directed Graph Example

Example



Multigraphs

Definition

parallel edges:
edges between the same node pair

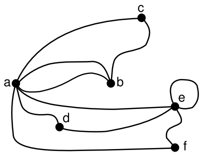
loop:
an edge whose ends are incident to the same node

plain graph:
a graph which does not contain any loops or parallel edges

multigraph:
a graph which is not plain

Multigraph Example

Example



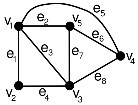
- parallel edges: (a, b)
- loop: (e, e)

Representation

- *incidence matrix*:
  - rows represent nodes, columns represent edges
  - cell: 1 if the edge is incident to the node, 0 otherwise
- *adjacency matrix*:
  - rows and columns represent nodes
  - cells represent the number of edges between the nodes

Incidence Matrix Example

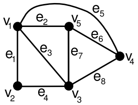
Example



	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8
v_1	1	1	1	0	1	0	0	0
v_2	1	0	0	1	0	0	0	0
v_3	0	0	1	1	0	0	1	1
v_4	0	0	0	0	1	1	0	1
v_5	0	1	0	0	0	1	1	0

Adjacency Matrix Example

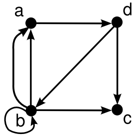
Example



	v_1	v_2	v_3	v_4	v_5
v_1	0	1	1	1	1
v_2	1	0	1	0	0
v_3	1	1	0	1	1
v_4	1	0	1	0	1
v_5	1	0	1	1	0

Adjacency Matrix Example

Example



	a	b	c	d
a	0	0	0	1
b	2	1	1	0
c	0	0	0	0
d	0	1	1	0

Degree

Definition

degree: number of edges incident to the node

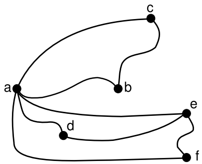
Theorem

if the degree of node v_i is d_i:

$$|E| = \frac{\sum_i d_i}{2}$$

Degree Example

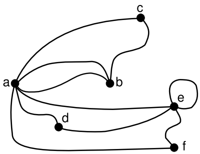
Example (plain)



$$\begin{aligned} d_a &= 5 \\ d_b &= 2 \\ d_c &= 2 \\ d_d &= 2 \\ d_e &= 3 \\ d_f &= 2 \\ Total &= 16 \\ |E| &= 8 \end{aligned}$$

Degree Example

Example (multigraph)



$$\begin{aligned} d_a &= 6 \\ d_b &= 3 \\ d_c &= 2 \\ d_d &= 2 \\ d_e &= 5 \\ d_f &= 2 \\ Total &= 20 \\ |E| &= 10 \end{aligned}$$

Degree in Directed Graphs

- two types of degree
  - *in-degree*: ${d_v}^i$
  - *out-degree*: ${d_v}^o$
- node with in-degree 0: *source*
- node with out-degree 0: *sink*
- $\sum_{v \in V} {d_v}^i = \sum_{v \in V} {d_v}^o = |A|$

Degree

Theorem
In an undirected graph, the number of nodes with an odd degree is even.

Proof.

- t_i: number of nodes of degree i
  $2|E| = \sum_i d_i = 1t_1 + 2t_2 + 3t_3 + 4t_4 + 5t_5 + \ldots$
  $2|E| - 2t_2 - 4t_4 - \cdots = t_1 + t_3 + \cdots + 2t_3 + 4t_5 + \ldots$
  $2|E| - 2t_2 - 4t_4 - \cdots - 2t_3 - 4t_5 - \cdots = t_1 + t_3 + t_5 + \ldots$
- since the left-hand side is even, the right-hand side is also even

$\square$

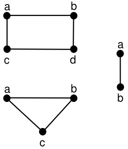
Regular Graphs

Definition
regular graph:
all nodes have the same degree

- n-regular: all nodes have degree n

Regular Graph Example

Example



Completely Connected Graphs

Definition

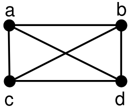
completely connected graph:

$\forall v_1, v_2 \in V \ (v_1, v_2) \in E$

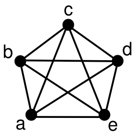
- K_n: a complete graph of n nodes

Completely Connected Graph Examples

Example (K_4)



Example (K_5)



Bipartite Graphs

Definition

bipartite graph:

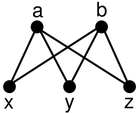
$V = V_1 \cup V_2 \wedge V_1 \cap V_2 = \emptyset$

$\forall (v_1, v_2) \in E \ v_1 \in V_1 \wedge v_2 \in V_2$

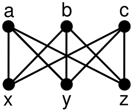
- *complete bipartite graph*:
  $\forall v_1 \in V_1 \forall v_2 \in V_2 \ (v_1, v_2) \in E$
  - $K_{m,n}$: $|V_1| = m$, $|V_2| = n$

Bipartite Graph Examples

Example ($K_{2,3}$)



Example ($K_{3,3}$)



Subgraph

Definition

subgraph:

if $G' = (V', E')$ is a subgraph of $G = (V, E)$

- $V' \subseteq V$
- $E' \subseteq E$
- $\forall (v_1, v_2) \in E' \; v_1 \in V' \wedge v_2 \in V'$

Isomorphism

Definition

isomorphic graphs:

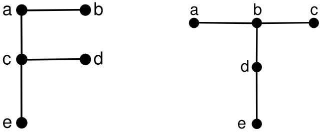
if $G = (V, E)$ and $G^\star = (V^\star, E^\star)$ are isomorphic

$\exists f : V \rightarrow V^\star \; (u, v) \in E \Rightarrow (f(u), f(v)) \in E^\star$

- f is bijective
- can be drawn the same way

Isomorphism Example

Example



- $f = \{(a, d), (b, e), (c, b), (d, c), (e, a)\}$

Isomorphism Example

Example (Petersen graph)

- $f = \{(a, q), (b, v), (c, u), (d, y), (e, r),$
  $(f, w), (g, x), (h, t), (i, z), (j, s)\}$

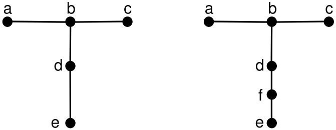
Homeomorphism

Definition

homeomorphic graph:
graph obtained by dividing an edge in an isomorphic graph with additional nodes

Homeomorphism Example

Example



Walk

Definition

walk:
a sequence of nodes and edges starting at node (v_0) and ending at node (v_n) in the form

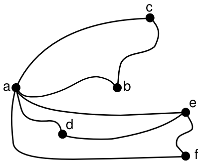
$$v_0, e_1, v_1, e_2, v_2, e_3, v_3, \ldots, e_{n-1}, v_{n-1}, e_n, v_n$$

where $e_i = (v_{i-1}, v_i)$

- no need to write the edges
- length: number of edges
- if $v_0 \neq v_n$ open, if $v_0 = v_n$ closed

Walk Example

Example



$(c, b), (b, a), (a, d), (d, e),$
$(e, f), (f, a), (a, b)$

c, b, a, d, e, f, a, b

length: 7

Trail

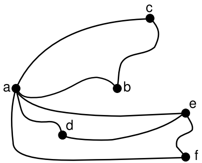
Definition

trail: a walk where edges are not repeated

- closed trail: circuit
- spanning trail: a trail that visits all the edges in the graph

Trail Example

Example



$(c, b), (b, a), (a, e), (e, d),$
$(d, a), (a, f)$

c, b, a, e, d, a, f

Path

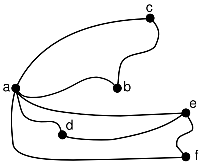
Definition

path: a walk where nodes are not repeated

- closed path: cycle
- spanning path: a path that visits all the nodes in the graph

Path Example

Example



$(c, b), (b, a), (a, d), (d, e),$
(e, f)

c, b, a, d, e, f

Connectivity

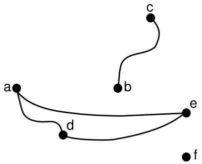
Definition

connected graph:
there is a path between all node pairs

- a disconnected graph can be separated into connected components

Connected Components Example

Example



- graph is disconnected: no path between a and c
- connected components:
  a, d, e
  b, c
  f

Distance

Definition

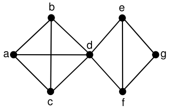
distance: length of the shortest path between two nodes

Definition

diameter: largest distance in the graph

Distance Example

Example



- distance between a and e: 2
- diameter: 3

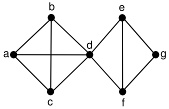
Cut-Point

Definition
$G - v$:
graph obtained by deleting node v and all its incident edges from graph G

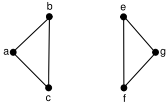
Definition
cut-point:
if G is connected but $G - v$ is disconnected then v is a cut-point

Cut-Point Example

G



$G - d$



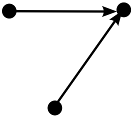
Directed Walks

- similar to undirected graphs
- assuming the arcs as undirected edges: *semi-walk*, *semi-trail*, *semi-path*

Weakly Connected Graph

Definition
weakly connected:
there is a semi-path between each node pair

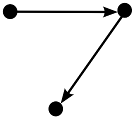
Example



Unilaterally Connected Graph

Definition
unilaterally connected:
for each node pair, there is
a path from one to the other

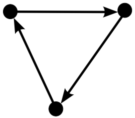
Example



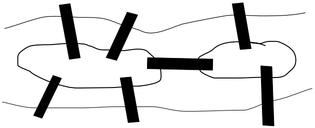
Strongly Connected Graph

Definition
strongly connected:
there is a path
between each node pair

Example



Bridges of Königsberg



- cross each bridge exactly once
  and return to the starting point

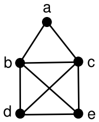
Traversable Graph

Definition
traversable graph:
a graph which contains a spanning trail

- an odd-degree node must be either the initial
  or the terminal node of the trail
- all nodes except the initial and the terminal nodes
  must have even degrees

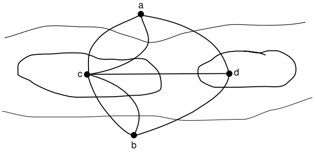
Traversable Graph Example

Example



- degrees of a, b and c are even
- degrees of d and e are odd
- a spanning trail can be formed starting from node d and ending at node e (or vice versa): $d, b, a, c, e, d, c, b, e$

Bridges of Königsberg



- all node degrees are odd: not traversable

Euler Graphs

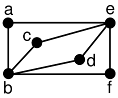
Definition

Euler graph:
a graph which contains a spanning circuit

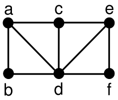
- Euler graph $\Leftrightarrow$ degrees of all nodes are even

Euler Graph Examples

Example (Euler graph)



Example (not an Euler graph)



Hamilton Graphs

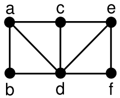
Definition

Hamilton graph:

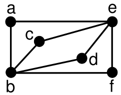
a graph which contains a spanning cycle

Hamilton Graph Examples

Example (Hamilton graph)



Example (not a Hamilton graph)



Connectivity Matrix

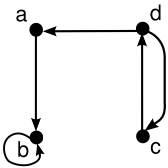
- if the adjacency matrix of the graph is A, the (i,j) element of A^k shows the number of walks of length k between the nodes i and j
- in an undirected graph with n nodes, the distance between two nodes is at most $n-1$
- connectivity matrix: $C = A^1 + A^2 + A^3 + \cdots + A^{n-1}$
  - if all elements are non-zero, then the graph is connected

Warshall's Algorithm

- it is easier to find whether there is a walk between two nodes instead of finding the number of walks
- for each node:
  - from all nodes which can reach the chosen node (the rows that contain 1 in the chosen column)
  - to the nodes which can be reached from the chosen node (the columns that contain 1 in the chosen row)

Warshall's Algorithm Example

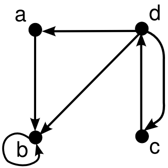
Example



	a	b	c	d
a	0	1	0	0
b	0	1	0	0
c	0	0	0	1
d	1	0	1	0

Warshall's Algorithm Example

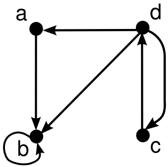
Example



	a	b	c	d
a	0	1	0	0
b	0	1	0	0
c	0	0	0	1
d	1	1	1	0

Warshall's Algorithm Example

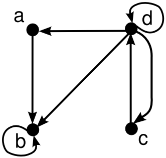
Example



	a	b	c	d
a	0	1	0	0
b	0	1	0	0
c	0	0	0	1
d	1	1	1	0

Warshall's Algorithm Example

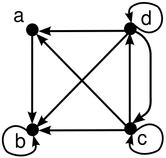
Example



	a	b	c	d
a	0	1	0	0
b	0	1	0	0
c	0	0	0	1
d	1	1	1	1

Warshall's Algorithm Example

Example



	a	b	c	d
a	0	1	0	0
b	0	1	0	0
c	1	1	1	1
d	1	1	1	1

Planar Graphs

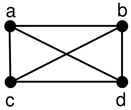
Definition

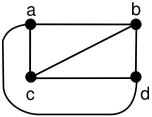
planar graph:
a graph that can be drawn on a plane
without any intersection of its edges

- map: a planar drawing of a graph

Planar Graph Example

Example (K_4)





Regions

- a map divides the plane into *regions*
- *degree of a region*:
  length of the circuit surrounding the region

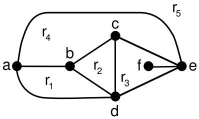
Theorem

if the degree of region r_i is d_{r_i}:

$$|E| = \frac{\sum_i d_{r_i}}{2}$$

Region Example

Example



$d_{r_1} = 3$ (abda)
$d_{r_2} = 3$ (bcdb)
$d_{r_3} = 5$ (cdefec)
$d_{r_4} = 4$ (abcea)
$d_{r_5} = 3$ (adea)

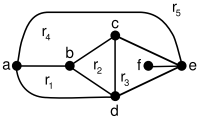
$\sum_r d_r = 18$
$|E| = 9$

Euler's Formula

Theorem (Euler's Formula)
In a planar and connected graph $|V| - |E| + |R| = 2$.

Euler's Formula Example

Example



- $|V| = 6$, $|E| = 9$, $|R| = 5$

Proof of Euler's Formula

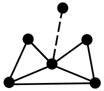
Proof
method: induction on $|E|$

- base step: one node, no edges
  $|V| = 1$, $|E| = 0$, $|R| = 1$
- assume it holds for a connected, planar graph with k nodes

Proof of Euler's Formula

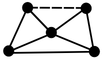
Induction Step.

- connect a new node to an existing node:



- $|V|$ is increased by 1, $|E|$ is increased by 1, $|R|$ remains the same

- add an edge between two existing nodes:



- $|V|$ remains the same, $|E|$ is increased by 1, $|R|$ is increased by 1

□

Planar Graph Theorems

Theorem

In a plain, planar graph:
$|V| \geq 3 \Rightarrow |E| \leq 3|V| - 6$

Proof.

- the sum of region degrees: $2|E|$
- degree of a region is at least 3
  $\Rightarrow 2|E| \geq 3|R| \Rightarrow |R| \leq \frac{2}{3}|E|$
- $|V| - |E| + |R| = 2$
  $\Rightarrow |V| - |E| + \frac{2}{3}|E| \geq 2 \Rightarrow |V| - \frac{1}{3}|E| \geq 2$
  $\Rightarrow 3|V| - |E| \geq 6 \Rightarrow |E| \leq 3|V| - 6$

□

Planar Graph Theorems

Theorem

In a connected, plain and planar graph
$|V| \geq 3 \Rightarrow \exists v \in V \; d_v \leq 5$

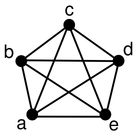
Proof.

- let $\forall v \in V \; d_v \geq 6$
  $\Rightarrow 2|E| \geq 6|V|$
  $\Rightarrow |E| \geq 3|V|$
  $\Rightarrow |E| > 3|V| - 6$: contradiction

□

Nonplanar Graphs

Theorem



K_5 is not planar.

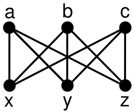
Proof.

- $|V| = 5$
- $3|V| - 6 = 3 \cdot 5 - 6 = 9$
- so $|E| \leq 9$
- but $|E| = 10$: contradiction

□

Nonplanar Graphs

Theorem



$K_{3,3}$ is not planar.

Proof.

- $|V| = 6, |E| = 9$
- if planar then $|R| = 5$
- degree of a region is at least 4
  $\Rightarrow \sum_{r \in R} d_r \geq 20$
- so $|E| \geq 10$
- but $|E| = 9$: contradiction

□

Kuratowski's Theorem

Theorem

The graph has a subgraph hemeomorphic to K_5 or $K_{3,3}$
$\Leftrightarrow$ the graph is not planar

Platonic Solids

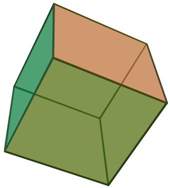
Definition

regular polyhedron:
a 3-dimensional solid where the faces are identical regular polygons

- the projection of a regular polyhedron onto the plane is a planar graph
  - every corner is a node
  - every side is an edge

Platonic Solids

Example (cube: regular hexahedron)



Platonic Solids

- v: number of nodes (corners)
- e: number of edges (side)
- r: number of regions (face)
- n: number of faces incident to a corner = node degree
- m: number of edges surrounding a face = region degree

- $m, n \geq 3$
- $2e = m \cdot r$
- $2e = n \cdot v$

Platonic Solids

- from Euler's formula:

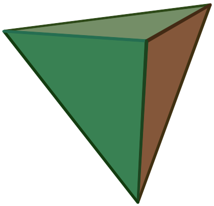
$$0 < 2 = v - e + r = \frac{2e}{n} - e + \frac{2e}{m} = e\left(\frac{2m - mn + 2n}{mn}\right)$$

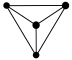
- Since $e, m, n > 0$:

$$2m - mn + 2n > 0 \Rightarrow mn - 2m - 2n < 0$$
$$\Rightarrow mn - 2m - 2n + 4 < 4 \Rightarrow (m-2)(n-2) < 4$$

- values satisfying the inequation:
  1. $m = 3, n = 3$
  2. $m = 4, n = 3$
  3. $m = 3, n = 4$
  4. $m = 5, n = 3$
  5. $m = 3, n = 5$

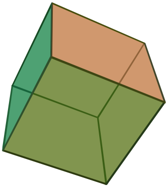
Tetrahedron

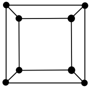




$m = 3, n = 3$

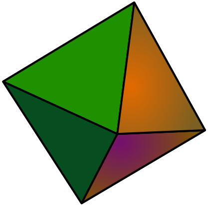
Hexahedron - Cube

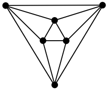




$m = 4, n = 3$

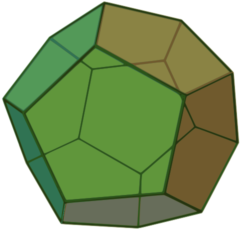
Octahedron

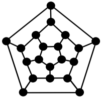




$m = 3, n = 4$

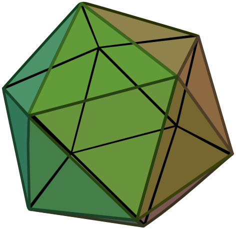
Dodecahedron





$m = 5, n = 3$

Icosahedron



$m = 3, n = 5$

Graph Coloring

Definition
proper coloring:
assign colors to all nodes in a graph $G = (V, E)$
so that for each $(v_1, v_2) \in E$ the colors of v_1 and v_2 are different

- using the minimum number of colors

Graph Coloring Example

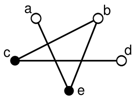
Example

- a company produce chemical compounds
- some compounds cannot be stored together
- such compounds must be placed in different storage areas
- store the compounds using the least number of storage areas

Graph Coloring

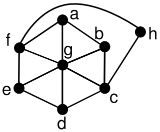
Example

- every compound is a node
- two compounds that cannot be stored together are adjacent



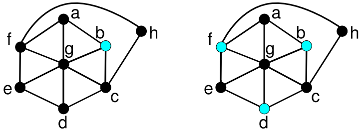
Graph Coloring Example

Example



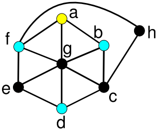
Graph Coloring Example

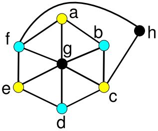
Example



Graph Coloring Example

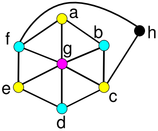
Example

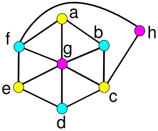




Graph Coloring Example

Example





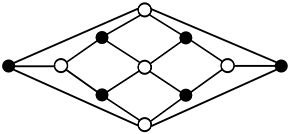
Chromatic Number

Definition
chromatic number:
Minimum number of colors needed to properly color the graph G: $\chi(G)$

- Calculating $\chi(G)$ is a very difficult problem
- for $n \geq 1$, $\chi(K_n) = n$

Example of Chromatic Number

Example (Herschel graph)



- chromatic number: 2

Graph Coloring Example

Example (Sudoku)

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

- every cell is a node
- cells of the same row are adjacent
- cells of the same column are adjacent
- cells of the same 3×3 block are adjacent
- every number is a color
- problem: properly color a graph that is partially-colored

Region Coloring

- coloring a map by assigning different colors to adjacent regions

Theorem (4 Color Theorem)

The regions in a planar map can be colored using 4 colors.

References

Required Text: Grimaldi

- Chapter 11: An Introduction to Graph Theory
- Chapter 7: Relations: The Second Time Around
  - 7.2. Computer Recognition: Zero-One Matrices and Directed Graphs

Tree

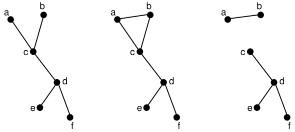
Definition

tree: $T = (V, E)$
a connected graph which contains no cycle

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

Tree Examples

Example

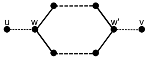


Tree Theorems

Theorem

In a tree, there is a unique path between any two distinct nodes.

- there is a path because the tree is connected
- if there were more than one path, it would cause a cycle:



Tree Theorems

Theorem

In a tree $T = (V, E)$*:* $|V| = |E| + 1$

- proof method: induction on the number of edges

Tree Theorems

Proof: Base step

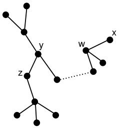
- $|E| = 0 \Rightarrow |V| = 1$
- $|E| = 1 \Rightarrow |V| = 2$
- $|E| = 2 \Rightarrow |V| = 3$

- assume that it is true for $|E| \leq k$

Tree Theorems

Proof: Induction step.

- $|E| = k + 1$



- delete edge (y, z):
  $T_1 = (V_1, E_1),\ T_2 = (V_2, E_2)$

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

□

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:
  $\Rightarrow 2|E| \geq 2(|V| - 1) + 1$
  $\Rightarrow 2|E| \geq 2|V| - 1$
  $\Rightarrow |E| \geq |V| - \frac{1}{2} > |V| - 1$ contradiction

□

Tree Theorems

Theorem

The following statements are equivalent:

1. *T is a tree (T is connected and contains no cycle).*
2. *There is a unique path between every pair of nodes in T.*
3. *T is connected, but if any edge is removed it will no longer be connected.*
4. *T contains no cycle, but if an edge is added between any pair of nodes a unique cycle will be formed.*

Tree Theorems

Theorem

The following statements are equivalent:

1. *T is a tree (T is connected and contains no cycle).*
2. *T is connected and $|E| = |V| - 1$.*
3. *T contains no cycle and $|E| = |V| - 1$.*

Rooted Tree

- there is a hierarchy between nodes
- natural direction on edges ⇒ in and out degrees
  - node with in-degree 0 (top of the hierarchy): root
  - nodes with out-degree 0: leaf
  - nodes that are not leaves: internal node

Node Level

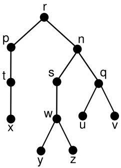
Definition

level: distance from the root

- parent: nearest node on the path from the root
- children: neighboring nodes in the next level
- sibling: nodes with the same parent

Rooted Tree Example

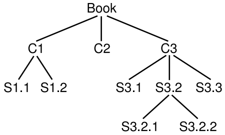
Example



- 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

Rooted Tree Example

Example (book order)



Book
- C1
  - S1.1
  - S1.2
- C2
- C3
  - S3.1
  - S3.2
    - S3.2.1
    - S3.2.2
  - S3.3

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 $1, 2, 3, \ldots$ to the nodes at level 1, from left to right
  - let v be an internal node with address a, assign the addresses $a.1, a.2, a.3, \ldots$ to the children of v from left to right

Lexicographic Order

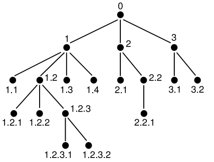
- let b and c be two addresses

Definition
for $b < c$:

1. $b = a_1.a_2.\ldots.a_m$
   $c = a_1.a_2.\ldots.a_m.a_{m+1}\ldots a_n$
2. $b = a_1.a_2.\ldots.a_m.x_1\ldots y$
   $c = a_1.a_2.\ldots.a_m.x_2\ldots z$
   $x_1 < x_2$

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

Binary Trees

Definition
binary tree:
$\forall v \in V \ d_v{}^o \in \{0, 1, 2\}$

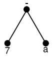
Definition
complete binary tree:
$\forall v \in V \ d_v{}^o \in \{0, 2\}$

Expression Tree

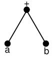
- binary operations can be represented by complete binary trees
  - operator as the root, operands as the children
- every mathematical expression can be represented as a binary tree
  - operators at internal nodes, variables and values at the leaves
  - *does not have to be a complete binary tree*

Expression Tree Examples

Example $(7 - a)$

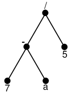


Example $(a + b)$

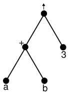


Expression Tree Examples

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

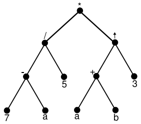


Example $((a + b) \uparrow 3)$



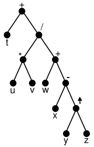
Expression Tree Examples

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



Expression Tree Examples

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

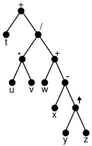


Expression Tree Traversals

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

Preorder Traversal Example

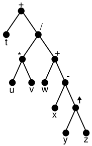
Example



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

Inorder Traversal Example

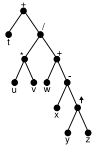
Example



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

Postorder Traversal Example

Example



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

Expression Tree Evaluation

- precedence in an expression tree:
  - inorder traversal requires parantheses
  - preorder and postorder traversals do not require parantheses

Postorder Evaluation Example

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

$4\ 2\ 3\ *\ 1\ 9\ 2\ 3\ \uparrow\ -\ +\ /\ +$

4	2	3	$*$			
4	6	1	9	2	3	$\uparrow$
4	6	1	9	8	$-$	
4	6	1	1	$+$		
4	6	2	$/$			
4	3	$+$				
7						

Searching Graphs

- searching nodes of a graph $G = (V, E)$ starting from node v_1
  - depth-first
  - breadth-first

Depth-First Search

1. $v \leftarrow v_1$, $T = \emptyset$, $D = \{v_1\}$
2. find smallest i in $2 \leq i \leq |V|$ such that $(v, v_i) \in E$ and $v_i \notin D$
   - if no such i exists: go to step 3
   - if found: $T = T \cup \{(v, v_i)\}$, $D = D \cup \{v_i\}$, $v \leftarrow v_i$, go to step 2
3. if $v = v_1$ then the result is T
4. if $v \neq v_1$ then $v \leftarrow parent(v)$, go to step 2

Breadth-First Search

1. $T = \emptyset$, $D = \{v_1\}$, $Q = (v_1)$
2. if Q is empty: the result is T
3. if Q not empty: $v \leftarrow front(Q)$, $Q \leftarrow Q - v$
   for $2 \leq i \leq |V|$ check the edges $(v, v_i) \in E$:
   - if $v_i \notin D$: $Q = Q + v_i$, $T = T \cup \{(v, v_i)\}$, $D = D \cup \{v_i\}$
   - go to step 3

Regular Tree

Definition
m-ary tree:
all internal nodes have out-degree m

Regular Tree Theorems

Theorem
in an m-ary tree

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

then

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

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

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: $i = \frac{l-1}{m-1} = \frac{27-1}{2-1} = 26$

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: $i = \frac{l-1}{m-1} = \frac{25-1}{4-1} = 8$

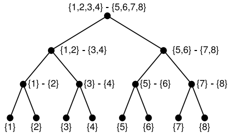
Decision Trees

Example (counterfeit coin problem)

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

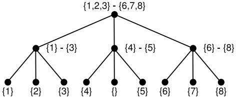
Decision Trees

Example (in 3 weighings)



Decision Trees

Example (in 2 weighings)



References

Required Text: Grimaldi

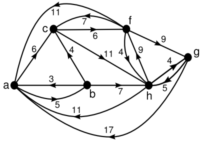
- Chapter 12: Trees
  - 12.1. Definitions and Examples
  - 12.2. Rooted Trees

Shortest Path

- Dijkstra's algorithm finds the shortest paths from a node to all other nodes

Dijkstra's Algorithm Example

Example (initialization)



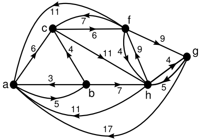
- starting node: *c*

a	$(\infty, -)$
b	$(\infty, -)$
c	$(0, -)$
f	$(\infty, -)$
g	$(\infty, -)$
h	$(\infty, -)$

Dijkstra's Algorithm Example

Example (From node c - base distance=0)

- $c \rightarrow f : 6, 6 < \infty$
- $c \rightarrow h : 11, 11 < \infty$



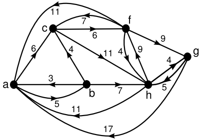
a	$(\infty, -)$	
b	$(\infty, -)$	
c	$(0, -)$	$\surd$
f	$(6, cf)$	
g	$(\infty, -)$	
h	$(11, ch)$	

- closest node: f

Dijkstra's Algorithm Example

Example (from node f - base distance=6)

- $f \rightarrow a : 6 + 11, 17 < \infty$
- $f \rightarrow g : 6 + 9, 15 < \infty$
- $f \rightarrow h : 6 + 4, 10 < 11$



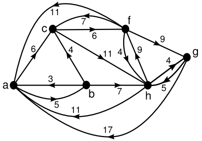
a	$(17, cfa)$	
b	$(\infty, -)$	
c	$(0, -)$	$\surd$
f	$(6, cf)$	$\surd$
g	$(15, cfg)$	
h	$(10, cfh)$	

- closest node: h

Dijkstra's Algorithm Example

Example (from node h - base distance=10)

- $h \rightarrow a : 10 + 11, 21 \not< 17$
- $h \rightarrow g : 10 + 4, 14 < 15$



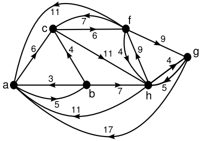
a	$(17, cfa)$	
b	$(\infty, -)$	
c	$(0, -)$	$\surd$
f	$(6, cf)$	$\surd$
g	$(14, cfhg)$	
h	$(10, cfh)$	$\surd$

- closest node: g

Dijkstra's Algorithm Example

Example (from node g - base distance=14)

- $g \rightarrow a : 14 + 17, 31 \not< 17$

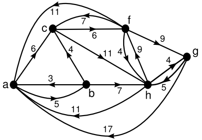


a	$(17, cfa)$	
b	$(\infty, -)$	
c	$(0, -)$	$\surd$
f	$(6, cf)$	$\surd$
g	$(14, cfhg)$	$\surd$
h	$(10, cfh)$	$\surd$

- closest node: a

Dijkstra's Algorithm Example

Example (from node a - base distance=17)



- $a \rightarrow b : 17 + 5, 22 < \infty$

a	$(17, cfa)$	$\surd$
b	$(22, cfab)$	
c	$(0, -)$	$\surd$
f	$(6, cf)$	$\surd$
g	$(14, cfhg)$	$\surd$
h	$(10, cfh)$	$\surd$

- last node: b

Spanning Tree

Definition

spanning tree:
a subgraph which is a tree and contains all the nodes of the graph

Definition

minimum spanning tree:
a spanning tree for which the total weight of edges is minimal

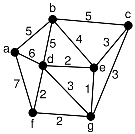
Kruskal's Algorithm

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

Kruskal's Algorithm Example

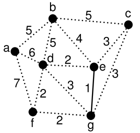
Example (initialization)



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

Kruskal's Algorithm Example

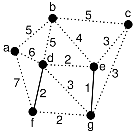
Example $(1 < 6)$



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

Kruskal's Algorithm Example

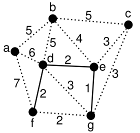
Example $(2 < 6)$



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

Kruskal's Algorithm Example

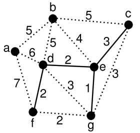
Example $(3 < 6)$



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

Kruskal's Algorithm Example

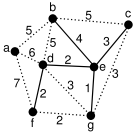
Example $(4 < 6)$



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

Kruskal's Algorithm Example

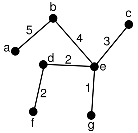
Example ($5 < 6$)



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

Kruskal's Algorithm Example

Example ($6 \not< 6$)



- total weight: 17

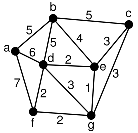
Prim's Algorithm

Prim's algorithm

1. $i \leftarrow 1$, $v_1 \in V$, $P = \{v_1\}$, $N = V - \{v_1\}$, $T = \emptyset$
2. for $1 \leq i \leq n - 1$:
   $P = \{v_1, v_2, \ldots, v_i\}$, $T = \{e_1, e_2, \ldots, e_{i-1}\}$, $N = V - P$
   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 $e_1, e_2, \ldots, e_{n-1}$ is a minimum spanning tree
   - $i < n \Rightarrow$ go to step 2

Prim's Algorithm Example

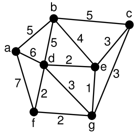
Example (initialization)



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

Prim's Algorithm Example

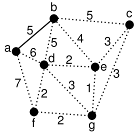
Example ($1 < 7$)



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

Prim's Algorithm Example

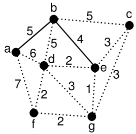
Example ($2 < 7$)



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

Prim's Algorithm Example

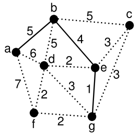
Example ($3 < 7$)



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

Prim's Algorithm Example

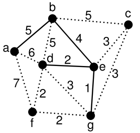
Example ($4 < 7$)



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

Prim's Algorithm Example

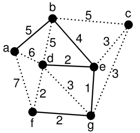
Example ($5 < 7$)



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

Prim's Algorithm Example

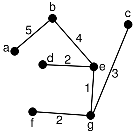
Example ($6 < 7$)



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

Prim's Algorithm Example

Example ($7 \not< 7$)



- total weight: 17

References

Required Text: Grimaldi

- Chapter 13: Optimization and Matching
  - 13.1. Dijkstra's Shortest Path Algorithm
  - 13.2. Minimal Spanning Trees:
    The Algorithms of Kruskal and Prim