Discrete Mathematics

Graphs

H. Turgut Uyar   Ayşegül Gençata Yayımlı   Emre Harmancı

2001-2014

Topics

Graphs

Definition

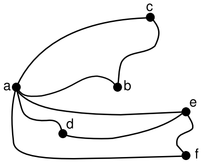
graph: $G = (V, E)$

- 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

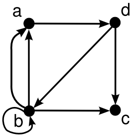
Definition

directed graph (or *digraph*): $D = (V, A)$

- V: node set
- $A \subseteq V \times V$: arc set
- if $a = (v_1, v_2) \in A$:
  - v_1: *origin* node of a
  - v_2: *terminating* node of a

Directed Graph Example

Example



Weighted Graphs

- in a weighted graph, labels are assigned to edges: weight, length, cost, delay, probability, ...

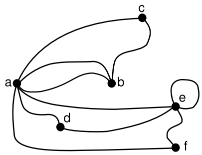
Multigraphs

- parallel edges: edges between the same pair of nodes
- loop: an edge starting and ending in the same node

- plain graph: a graph without any loops or parallel edges
- multigraph: a graph which is not plain

Multigraph Example

Example



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

Subgraph

Definition

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

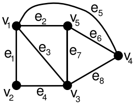
- $V' \subseteq V$, and
- $E' \subseteq E$, and
- $\forall (v_1, v_2) \in E' \; v_1, v_2 \in V'$

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
  - cell: 1 if the nodes are adjacent, 0 otherwise
  - in a multigraph, the cells can represent the number of edges between the nodes
  - in a weighted graph, the cells can represent the labels assigned to the edges

Incidence Matrix Example

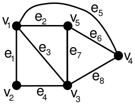
Example



	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

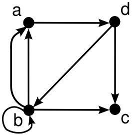
Example



	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

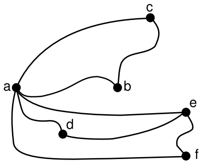
Definition
degree: number of edges incident to the node

Theorem
let d_i be the degree of node v_i:

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

Degree Example

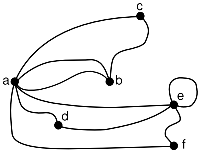
Example (plain graph)



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

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, there is an even number of nodes which have an odd degree.

Proof.

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

$\square$

Isomorphism

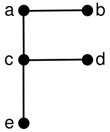
Definition

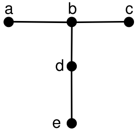
$G = (V, E)$ and $G^\star = (V^\star, E^\star)$ are isomorphic:

- $\exists f : V \rightarrow V^\star \; (u, v) \in E \Rightarrow (f(u), f(v)) \in E^\star$
- f is bijective
- G and $G^\star$ can be drawn the same way

Isomorphism Example

Example

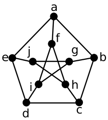


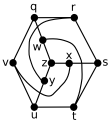


- $f = (a \mapsto d, b \mapsto e, c \mapsto b, d \mapsto c, e \mapsto a)$

Isomorphism Example

Example (Petersen graph)





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

Homeomorphism

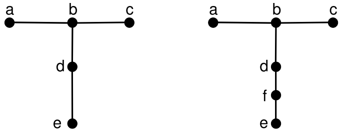
Definition

$G = (V, E)$ and $G^\star = (V^\star, E^\star)$ are homeomorphic:

- G and $G^\star$ are isomorphic except that some edges in $E^\star$ are divided with additional nodes

Homeomorphism Example

Example



Regular Graphs

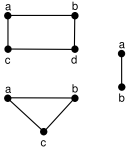
Definition

regular graph: all nodes have the same degree

- n-regular: all nodes have degree n

Regular Graph Examples

Example



Completely Connected Graphs

Definition

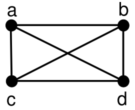
$G = (V, E)$ is completely connected:

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

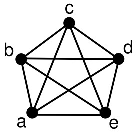
- there is an edge between every pair of nodes
- K_n: the completely connected graph with n nodes

Completely Connected Graph Examples

Example (K_4)



Example (K_5)



Bipartite Graphs

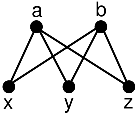
Definition

$G = (V, E)$ is bipartite:

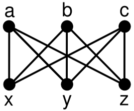
- $\forall (v_1, v_2) \in E \; v_1 \in V_1 \wedge v_2 \in V_2$
- $V_1 \cup V_2 = V, \; V_1 \cap V_2 = \emptyset$
- *complete bipartite*: $\forall v_1 \in V_1 \; \forall v_2 \in V_2 \; (v_1, v_2) \in E$
- $K_{m,n}$: $|V_1| = m$, $|V_2| = n$

Complete Bipartite Graph Examples

Example ($K_{2,3}$)



Example ($K_{3,3}$)



Walk

Definition

walk: a sequence of nodes and edges
from a starting node (v_0) to an ending node (v_n)

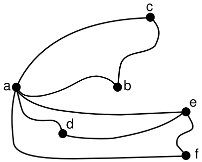
$$v_0 \xrightarrow{e_1} v_1 \xrightarrow{e_2} v_2 \xrightarrow{e_3} v_3 \longrightarrow \cdots \longrightarrow v_{n-1} \xrightarrow{e_n} v_n$$

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

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

Walk Example

Example



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

$$c \rightarrow b \rightarrow a \rightarrow d \rightarrow e \rightarrow f \rightarrow a \rightarrow b$$

length: 7

Trail

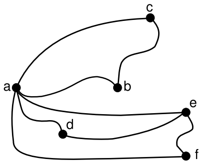
Definition

trail: a walk where edges are not repeated

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

Trail Example

Example



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

$$c \rightarrow a \rightarrow e \rightarrow d \rightarrow a \rightarrow f$$

Path

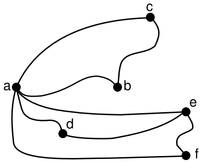
Definition

path: a walk where nodes are not repeated

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

Path Example

Example



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

$$c \to b \to a \to d \to e \to f$$

Connectivity

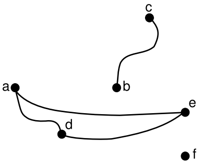
Definition

connected graph:
there is a path between every pair of nodes

- a disconnected graph can be divided 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

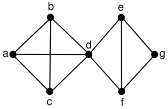
Definition

distance between nodes v_i and v_j:
the length of the shortest path between v_i and v_j

- diameter: the largest distance in the graph

Distance Example

Example



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

Cut-Points

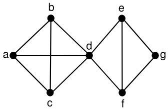
- the graph $G - v$ is obtained by deleting the node v and all its incident edges from the graph G

Definition

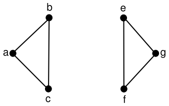
v is a cut-point for G:
G is connected but $G - v$ is not

Cut-Point Example

G



$G - d$



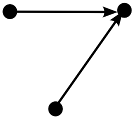
Directed Walks

- same as in undirected graphs
- ignoring the directions on the arcs: *semi-walk*, *semi-trail*, *semi-path*

Weakly Connected Graph

Definition
weakly connected:
there is a semi-path
between every pair of nodes

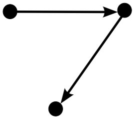
Example



Unilaterally Connected Graph

Definition
unilaterally connected:
for every pair of nodes, there is
a path from one to the other

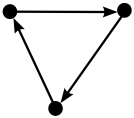
Example



Strongly Connected Graph

Definition
strongly connected:
there is a path in both directions
between every pair of nodes

Example



Connectivity Matrix

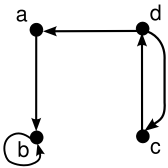
- let A be the adjacency matrix of an undirected graph $G = (V, E)$
- A^k_{ij}: number of walks of length k between nodes i and j
- the distance between two nodes is at most $|V| - 1$
- connectivity matrix:
  $C = A^1 + A^2 + A^3 + \cdots + A^{|V|-1}$
- if all elements of C are non-zero, then G is connected

Warshall's Algorithm

- it is easier to find whether there is a walk between two nodes rather than 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

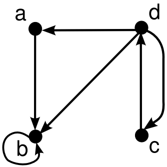
Example



	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

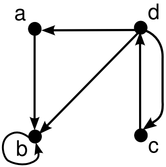
Example



	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

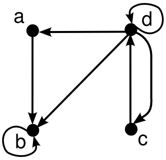
Example



	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

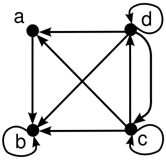
Example



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

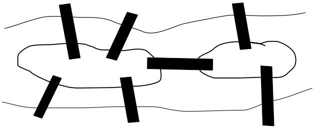
Warshall's Algorithm Example

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

Bridges of Königsberg

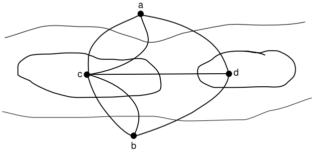


- cross each bridge exactly once
  and return to the starting point

Traversable Graphs

Definition

G is traversable: G contains a spanning trail

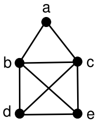


Traversable Graphs

- a node with an odd degree must be either the starting node or the ending node of the trail
- all nodes except the starting node and the ending node must have even degrees

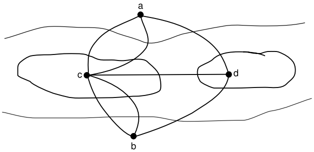
Traversable Graph Example

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):
  $d \rightarrow b \rightarrow a \rightarrow c \rightarrow e$
  $\rightarrow d \rightarrow c \rightarrow b \rightarrow e$

Bridges of Königsberg



- all node have odd degrees: not traversable

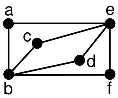
Euler Graphs

Definition
Euler graph: a graph that contains a closed spanning trail

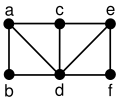
- G is an Euler graph $\Leftrightarrow$ all nodes in G have even degrees

Euler Graph Examples

Example (Euler graph)



Example (not an Euler graph)

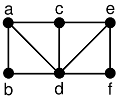


Hamilton Graphs

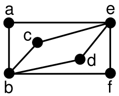
Definition
Hamilton graph: a graph that contains a closed spanning path

Hamilton Graph Examples

Example (Hamilton graph)



Example (not a Hamilton graph)



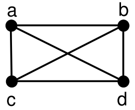
Planar Graphs

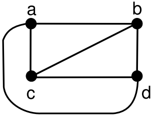
Definition
G is planar:
G can be drawn on a plane without intersecting its edges

- a map of G: a planar drawing of G

Planar Graph Example

Example (K_4)





Regions

- a map divides the plane into regions
- degree of a region:
  length of the closed trail that surrounds the region

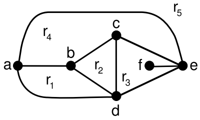
Theorem

let d_{r_i} be the degree of region 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

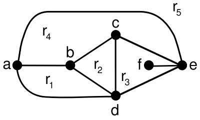
Theorem (Euler's Formula)

let $G = (V, E)$ be a planar, connected graph, and let R be the set of regions in a map of G:

$$|V| - |E| + |R| = 2$$

Euler's Formula Example

Example



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

Planar Graph Theorems

Theorem

let $G = (V, E)$ be a connected, planar graph where $|V| \geq 3$:
$|E| \leq 3|V| - 6$

Proof.

- 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

let $G = (V, E)$ be a connected, planar graph where $|V| \geq 3$:
$\exists v \in V \; d_v \leq 5$

Proof.

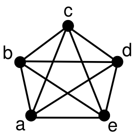
- let $\forall v \in V \; d_v \geq 6$
  $\Rightarrow 2|E| \geq 6|V|$
  $\Rightarrow |E| \geq 3|V|$
  $\Rightarrow |E| > 3|V| - 6$

□

Nonplanar Graphs

Theorem

K_5 is not planar.



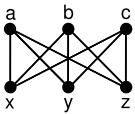
Proof.

- $|V| = 5$
- $3|V| - 6 = 3 \cdot 5 - 6 = 9$
- $|E| \leq 9$ should hold
- but $|E| = 10$

□

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
  $\Rightarrow \sum_{r \in R} d_r \geq 20$
- $|E| \geq 10$ should hold
- but $|E| = 9$

$\square$

Kuratowski's Theorem

Theorem

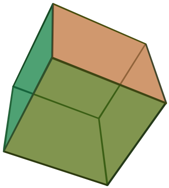
G contains a subgraph homeomorphic to K_5 or $K_{3,3}$.
$\Leftrightarrow$
G is not planar.

Platonic Solids

- *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
  - every face is a region

Platonic Solids

Example (cube)



Platonic Solids

- v: number of corners (nodes)
- e: number of sides (edges)
- r: number of faces (regions)
- n: number of faces meeting at a corner (node degree)
- m: number of sides of a face (region degree)

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

Platonic Solids

- from Euler's formula:

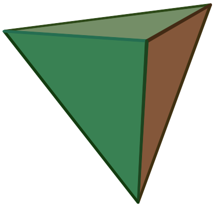
$$2 = v - e + r = \frac{2e}{n} - e + \frac{2e}{m} = e\left(\frac{2m - mn + 2n}{mn}\right) > 0$$

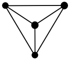
- $e, m, n > 0$:

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

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

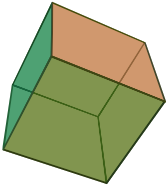
Tetrahedron

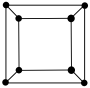




$m = 3, n = 3$

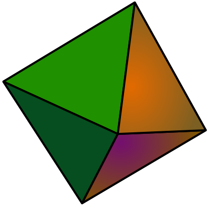
Hexahedron

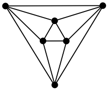




$m = 4, n = 3$

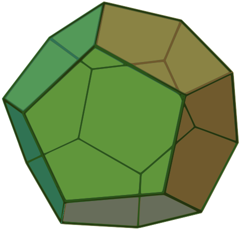
Octahedron

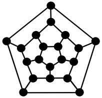




$m = 3, n = 4$

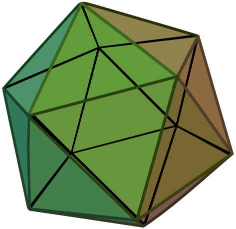
Dodecahedron





$m = 5, n = 3$

Icosahedron



$m = 3, n = 5$

Graph Coloring

Definition

proper coloring of $G = (V, E)$: $f : V \to C$
where C is a set of colors

- $\forall (v_i, v_j) \in E \; f(v_i) \neq f(v_j)$
- minimizing $|C|$

Graph Coloring Example

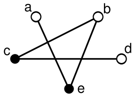
Example

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

Graph Coloring

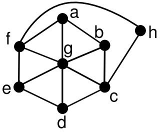
Example

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



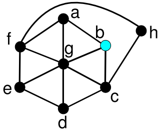
Graph Coloring Example

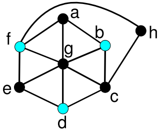
Example



Graph Coloring Example

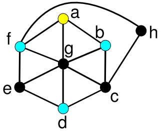
Example

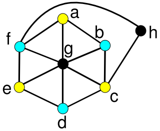




Graph Coloring Example

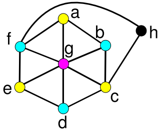
Example

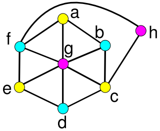




Graph Coloring Example

Example





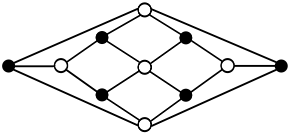
Chromatic Number

Definition
chromatic number of G: $\chi(G)$
minimum number of colors needed to color G

- finding $\chi(G)$ is a very difficult problem
- $\chi(K_n) = n$

Chromatic Number Example

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 (Four Color Theorem)

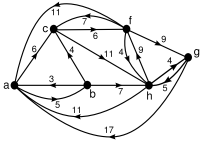
The regions in a map can be colored using four colors.

Shortest Path

- finding the shortest paths from a starting node to all other nodes: Dijkstra's algorithm

Dijkstra's Algorithm Example

Example (initialization)

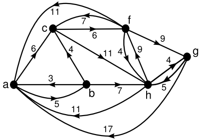


- starting node: c

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

Dijkstra's Algorithm Example

Example (from node c - base distance=0)



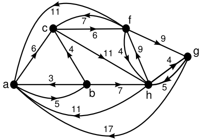
- $c \rightarrow f : 6, 6 < \infty$
- $c \rightarrow h : 11, 11 < \infty$

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)



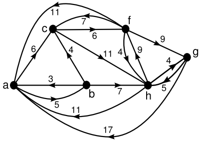
- $f \rightarrow a : 6 + 11, 17 < \infty$
- $f \rightarrow g : 6 + 9, 15 < \infty$
- $f \rightarrow h : 6 + 4, 10 < 11$

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)



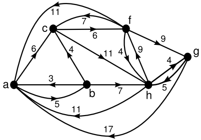
- $h \rightarrow a : 10 + 11, 21 \not< 17$
- $h \rightarrow g : 10 + 4, 14 < 15$

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)



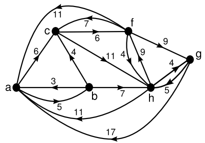
- $g \rightarrow a : 14 + 17, 31 \not< 17$

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

Searching Graphs

- searching nodes of 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 $v \leftarrow backtrack(v)$, 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

References

Required Reading: 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
- Chapter 13: Optimization and Matching
  - 13.1. Dijkstra's Shortest Path Algorithm