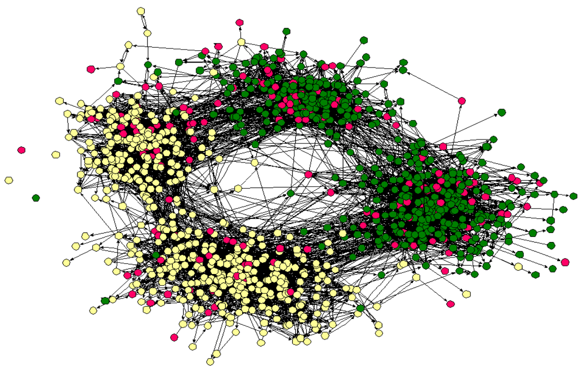
Graph (Contd)

Sesi 10

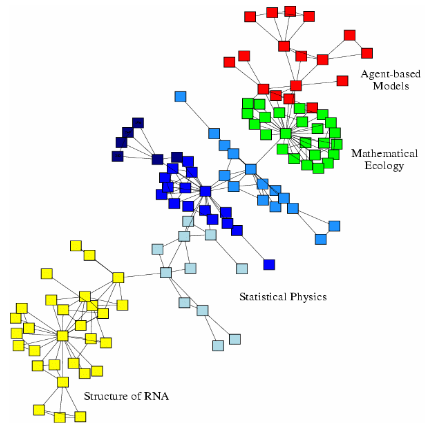
What makes a problem graph-like?

- There are two components to a graph
  - Nodes and edges
- In graph-like problems, these components have natural correspondences to problem elements
  - Entities are nodes and interactions between entities are edges
- Most complex systems are graph-like

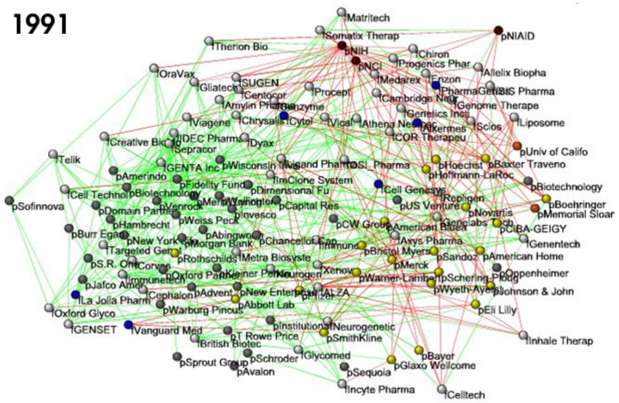
Friendship Network



Scientific collaboration network

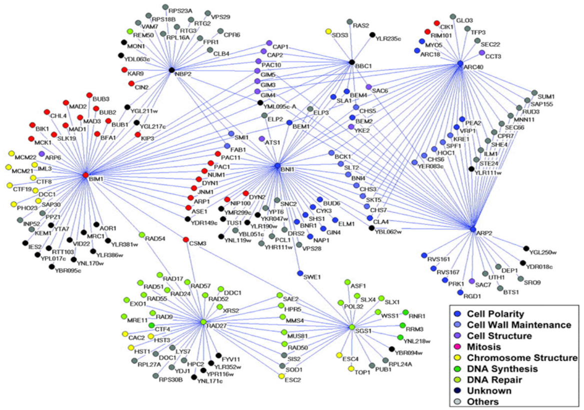


Business ties in US biotech-industry



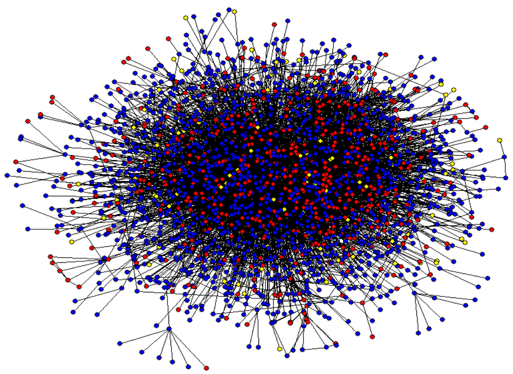
5

Genetic interaction network



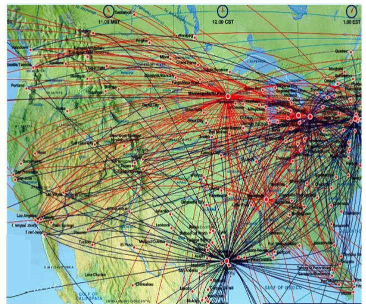
6

Protein-Protein Interaction Networks



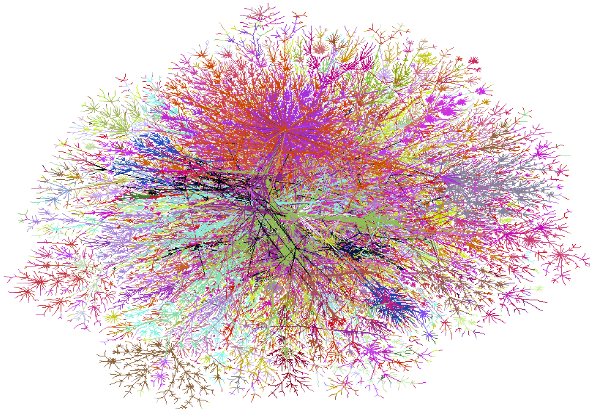
7

Transportation Networks

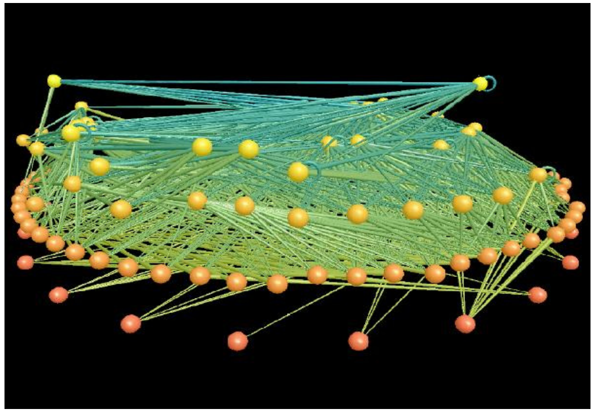


8

Internet



Ecological Networks



Structures and structural metrics

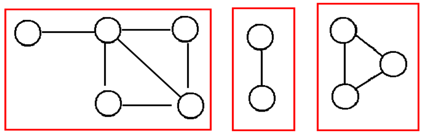
- Graph structures are used to isolate interesting or important sections of a graph
- Structural metrics provide a measurement of a structural property of a graph
  - Global metrics refer to a whole graph
  - Local metrics refer to a single node in a graph

Graph structures

- Identify interesting sections of a graph
  - Interesting because they form a significant domain-specific structure, or because they significantly contribute to graph properties
- A subset of the nodes and edges in a graph that possess certain characteristics, or relate to each other in particular ways

Connectivity & Component

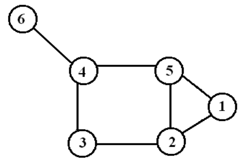
- A graph is ***connected*** if
  - you can get from any node to any other by following a sequence of edges OR
  - any two nodes are connected by a path.
- A directed graph is ***strongly connected*** if there is a directed path from any node to any other node.
- Every disconnected graph can be split up into a number of connected ***components***.



13

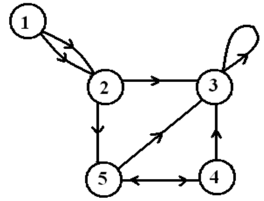
Degree

- Undirected Graph
  - Number of edges incident on a node



The degree of 5 is 3

- Directed Graph
  - In-degree: Number of edges entering
  - Out-degree: Number of edges leaving
- Degree = indeg + outdeg



outdeg(1)=2
indeg(1)=0

outdeg(2)=2
indeg(2)=2

outdeg(3)=1
indeg(3)=4

14

Degree: Simple Facts

- If G is a graph with m edges, then

$$\sum \deg(v) = 2m = 2\ |E\ |$$

- If G is a digraph then

$$\sum \text{indeg}(v)=\sum \text{outdeg}(v) = |E\ |$$

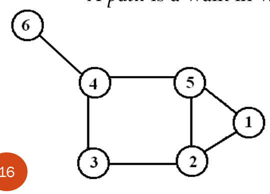
- Number of Odd degree Nodes is even

15

Walks, Cycle & Path

- A ***walk of length k*** in a graph is a succession of k (not necessarily different) edges of the form

  uv,vw,wx,…,yz.
- This walk is denote by uvwx…xz, and is referred to as a ***walk between u and z***.
- A walk is ***closed*** is u=z ➔Closed Walks
- A ***cycle*** is a closed walk in which all the edges are different
- A *path* is a walk in which all the edges and all the nodes are different.



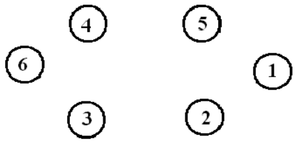
Walks, Cycle and Paths

1,2,5,2,3,4 walk of length 5	1,2,5,2,3,2,1 CW of length 6	1,2,3,4,6 path of length 4
1,2,5,1 3-cycle	2,3,4,5,2 4-cycle	

16

Special Types of Graphs

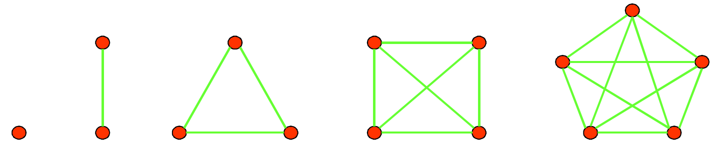
- Empty Graph / Edgeless graph
  - No edge



- Null graph
  - No nodes
  - Obviously no edge

Special Graphs

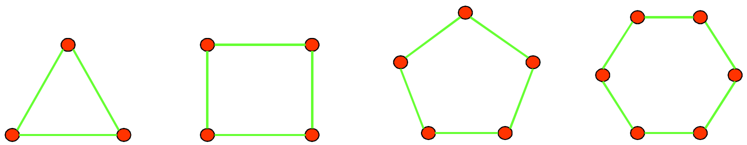
- **Definition:** The **complete graph** on n vertices, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices.



K_1 K_2 K_3 K_4 K_5

Special Graphs

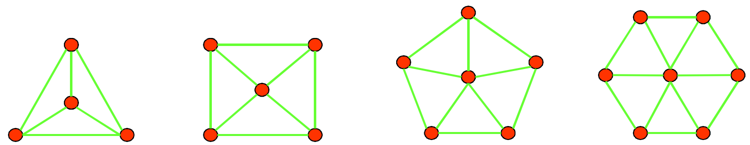
- **Definition:** The **cycle** C_n, $n \geq 3$, consists of n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$.



C_3 C_4 C_5 C_6

Special Graphs

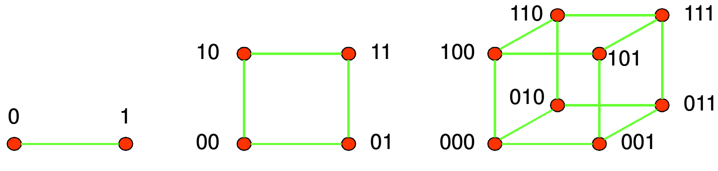
- **Definition:** We obtain the **wheel** W_n when we add an additional vertex to the cycle C_n, for $n \geq 3$, and connect this new vertex to each of the n vertices in C_n by adding new edges.



W_3 W_4 W_5 W_6

Special Graphs

•**Definition:** The **n-cube,** denoted by Q_n, is the graph that has vertices representing the 2^n bit strings of length n. Two vertices are adjacent if and only if the bit strings that they represent differ in exactly one bit position.



Q_1 Q_2 Q_3

21

Special Graphs

•**Definition:** A simple graph is called **bipartite** if its vertex set V can be partitioned into two disjoint nonempty sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 with a vertex in V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2).

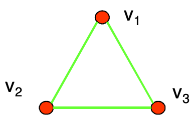
•For example, consider a graph that represents each person in a village by a vertex and each marriage by an edge.

•This graph is **bipartite**, because each edge connects a vertex in the **subset of males** with a vertex in the **subset of females** (if we think of traditional marriages).

22

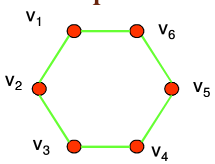
Special Graphs

•**Example I:** Is C_3 bipartite?

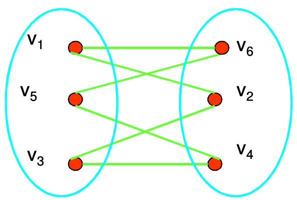


No, because there is no way to partition the vertices into two sets so that there are no edges with both endpoints in the same set.

•**Example II:** Is C_6 bipartite?



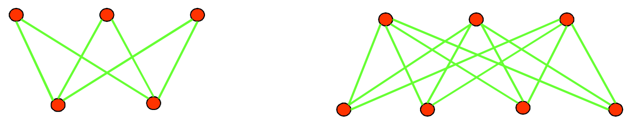
Yes, because we can display C_6 like this:



23

Special Graphs

•**Definition:** The **complete bipartite graph** $K_{m,n}$ is the graph that has its vertex set partitioned into two subsets of m and n vertices, respectively. Two vertices are connected if and only if they are in different subsets.

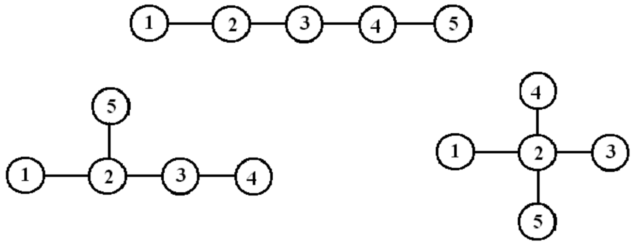


$K_{3,2}$ $K_{3,4}$

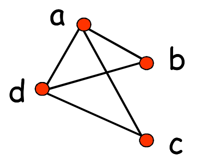
24

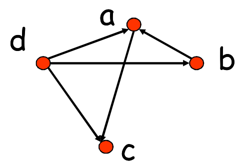
Trees

- Connected Acyclic Graph
- Two nodes have *exactly* one path between them



Representing Graphs

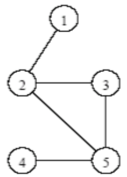




Vertex	Adjacent Vertices
a	b, c, d
b	a, d
c	a, d
d	a, b, c

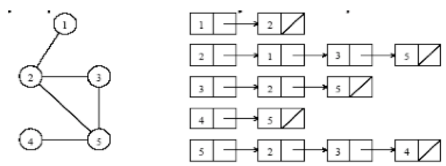
Initial Vertex	Terminal Vertices
a	c
b	a
c	
d	a, b, c

Representing Graphs



$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

An undirected graph and its adjacency matrix representation.



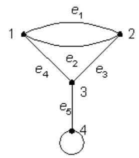
An undirected graph and its adjacency list representation.

Representation

- Matric
  - Incidence Matrix
    - V x E
    - [vertex, edges] contains the edge's data
  - Adjacency Matrix
    - V x V
    - Boolean values (adjacent or not)
    - Or Edge Weights
- List
  - Edge List
    - pairs (ordered if directed) of vertices
    - Optionally weight and other data
  - Adjacency List (node list)

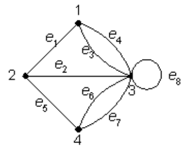
Representation (Matrix)

- Incidence Matrix $a_{ij} = \begin{cases} 1, & \text{jika simpul } i \text{ bersisian dengan sisi } j \\ 0, & \text{jika simpul } i \text{ tidak bersisian dengan sisi } j \end{cases}$



	e_1	e_2	e_3	e_4	e_5
1	1	1	0	1	0
2	1	1	1	0	0
3	0	0	1	1	1
4	0	0	0	0	1

- Adjacency Matrix $a_{ij} = \begin{cases} 1, & \text{jika simpul } i \text{ dan } j \text{ bertetangga} \\ 0, & \text{jika simpul } i \text{ dan } j \text{ tidak bertetangga} \end{cases}$



	1	2	3	4
1	0	1	2	0
2	1	0	1	1
3	2	1	1	2
4	0	1	2	0

29

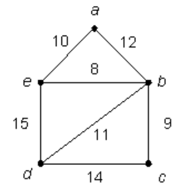
Degree

- Untuk graf tak-berarah,

$$d(v_i) = \sum_{j=1}^{n} a_{ij}$$

- Untuk graf berarah,
  - $d_{in}(v_j) = \text{jumlah nilai pada kolom } j = \sum_{i=1}^{n} a_{ij}$
  - $d_{out}(v_i) = \text{jumlah nilai pada baris } i = \sum_{j=1}^{n} a_{ij}$

Graph A



Matrix Graph A

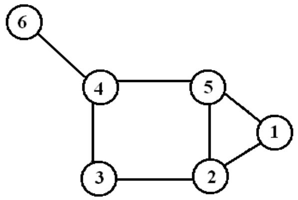
	a	b	c	d	e
a	∞	12	∞	∞	10
b	12	∞	9	11	8
c	∞	9	∞	14	∞
d	∞	11	14	∞	15
e	10	8	∞	15	∞

Degree Graph A ???

30

Topological Distance

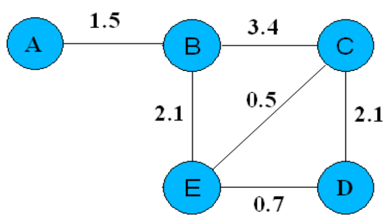
- A shortest path is the minimum path connecting two nodes.
- The number of edges in the shortest path connecting *p* and *q* is the ***topological distance*** between these two nodes, $d_{p,q}$
- Distance Matrix
  - |V| x |V| matrix D = (d_{ij}) such that d_{ij} is the topological distance between i and j.



	1	2	3	4	5	6
1	0	1	2	2	1	3
2	1	0	1	2	1	3
3	2	1	0	1	2	2
4	2	2	1	0	1	1
5	1	1	2	1	0	2
6	3	3	2	1	2	0

31

Adjacency Matrix of Weighted graphs

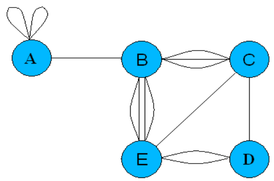


	A	B	C	D	E
A	0	1.5	0	0	0
B	1.5	0	3.4	0	0
C	0	3.4	0	2.1	0.5
D	0	0	2.1	0	0.7
E	0	2.1	0.5	0.7	0

32

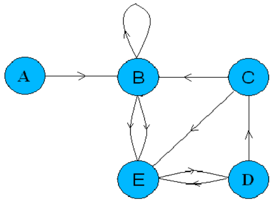
Adjacency Matrix of Multigraphs

Undirected Graph



	A	B	C	D	E
A	2	1	0	0	0
B	1	0	3	0	4
C	0	3	0	1	1
D	0	0	1	0	2
E	0	4	1	2	0

Directed Graph

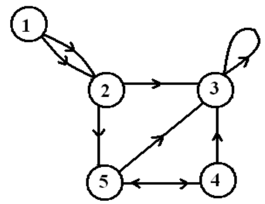


	A	B	C	D	E
A	0	1	0	0	0
B	0	1	0	0	2
C	0	1	0	0	1
D	0	0	1	0	1
E	0	0	0	1	0

33

Representation (List)

- ***Adjacency-list representation***
  - an array of $|V|$ lists, one for each vertex in V.
  - For each $u \in V$, $ADJ[u]$ points to all its adjacent vertices.
- Edge and Node Lists

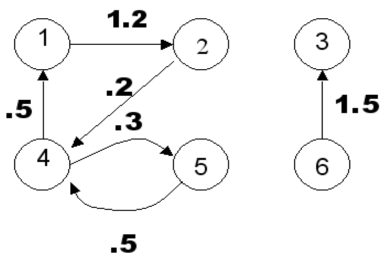


Edge List
1 2
1 2
2 3
2 5
3 3
4 3
4 5
5 3
5 4

Node List
1 2 2
2 3 5
3 3
4 3 5
5 3 4

34

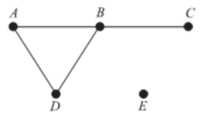
Edge Lists for Weighted Graphs



Edge List
1 2 1.2
2 4 0.2
4 5 0.3
4 1 0.5
5 4 0.5
6 3 1.5

35

List Representation



(a)

Vertex	Adjacency list
A	B, D
B	A, C, D
C	B
D	A, B
E	∅

(b)

•Vertex File

VERTEX	NEXT-V	PTR

•Edge File

EDGE	ADJ	NEXT

Vertex file

START 3

	1	2	3	4	5	6	7	8
VERTEX	C		A		E	B		D
NEXT-V	8		6		0	1		5
PTR	4		5		0	7		8

Edge file

	1	2	3	4	5	6	7	8	9	10
ADJ	2 (B)	1 (A)	3 (C)	2 (B)	4 (D)	1 (A)	4 (D)	2 (B)		
NEXT	0	0	2	0	1	0	3	6		

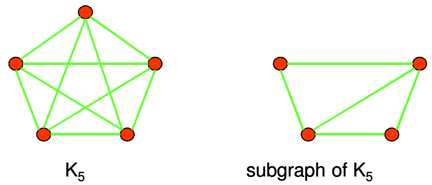
36

Operations on Graphs

•**Definition:** A **subgraph** of a graph G = (V, E) is a graph H = (W, F) where W⊆V and F⊆E.

•**Note:** Of course, H is a valid graph, so we cannot remove any endpoints of remaining edges when creating H.

•**Example:**

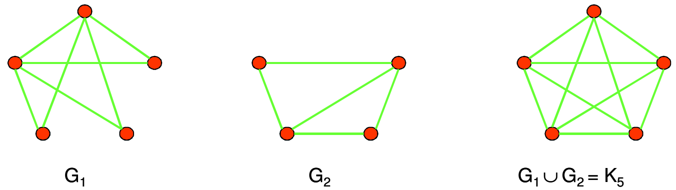


37

Operations on Graphs

•**Definition:** The **union** of two simple graphs G_1 = (V_1, E_1) and G_2 = (V_2, E_2) is the simple graph with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$.

•The union of G_1 and G_2 is denoted by $\mathbf{G_1 \cup G_2}$.



38

Isomorphism of Graphs

•**Definition:** The simple graphs G_1 = (V_1, E_1) and G_2 = (V_2, E_2) are **isomorphic** if there is a bijection (an one-to-one and onto function) f from V_1 to V_2 with the property that a and b are adjacent in G_1 if and only if f(a) and f(b) are adjacent in G_2, for all a and b in V_1.

•Such a function f is called an **isomorphism**.

•In other words, G_1 and G_2 are isomorphic if their vertices can be ordered in such a way that the adjacency matrices M_{G_1} and M_{G_2} are identical.

•From a visual standpoint, G_1 and G_2 are isomorphic if they can be arranged in such a way that their **displays are identical** (of course without changing adjacency).

•Unfortunately, for two simple graphs, each with n vertices, there are **n! possible isomorphisms** that we have to check in order to show that these graphs are isomorphic.

•However, showing that two graphs are **not** isomorphic can be easy.

39

Isomorphism of Graphs

•For this purpose we can check **invariants**, that is, properties that two isomorphic simple graphs must both have.

•For example, they must have

- the same number of vertices,
- the same number of edges, and
- the same degrees of vertices.

•Note that two graphs that **differ** in any of these invariants are not isomorphic, but two graphs that **match** in all of them are not necessarily isomorphic.

40

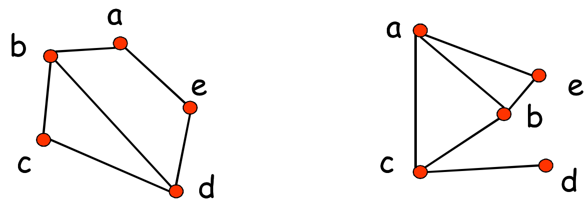
Isomorphism of Graphs

•**Example I:** Are the following two graphs isomorphic?

•**Solution:** Yes, they are isomorphic, because they can be arranged to look identical. You can see this if in the right graph you move vertex b to the left of the edge {a, c}. Then the isomorphism f from the left to the right graph is: f(a) = e, f(b) = a,
f(c) = b, f(d) = c, f(e) = d.

Isomorphism of Graphs

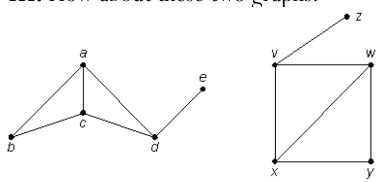
•**Example II:** How about these two graphs?



- **Solution:** No, they are not isomorphic, because they differ in the degrees of their vertices. Vertex d in right graph is of degree one, but there is no such vertex in the left graph.

Isomorphism of Graphs

•**Example III:** How about these two graphs?



G_1 G_2

- **Solution:** Yes.

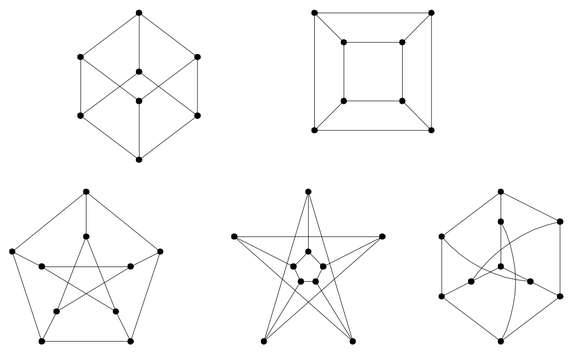
Coz $A_{G1}=A_{G2}$

$$A_{G1} = \begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & 0 & 1 & 1 & 1 & 0 \\ b & 1 & 0 & 1 & 0 & 0 \\ c & 1 & 1 & 0 & 1 & 0 \\ d & 1 & 0 & 1 & 0 & 1 \\ e & 0 & 0 & 0 & 1 & 0 \end{array}$$

$$A_{G2} = \begin{array}{c|ccccc} & x & y & w & v & z \\ \hline x & 0 & 1 & 1 & 1 & 0 \\ y & 1 & 0 & 1 & 0 & 0 \\ w & 1 & 1 & 0 & 1 & 0 \\ v & 1 & 0 & 1 & 0 & 1 \\ z & 0 & 0 & 0 & 1 & 0 \end{array}$$

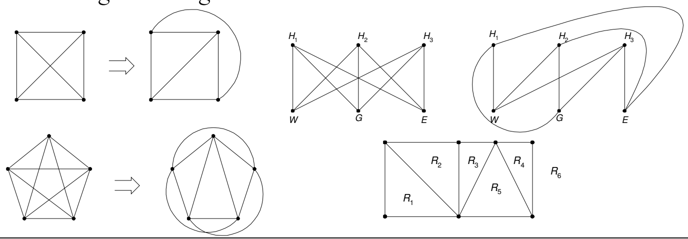
Isomorphism of Graphs

•**Example IV:** How about these graphs?



Graf Planar (Planar Graph) & Graf Bidang (Plane Graph)

- *Graf Planar (Planar Graph)* : Graf yang dapat digambarkan pada bidang datar dengan sisi-sisi tidak saling memotong
- Graf bidang (*plane graph*) : Graf planar yang digambarkan dengan sisi-sisi yang tidak saling berpotongan
- Manakah Grapf-graf berikut ini yang termasuk graf planar dan/atau graf bidang?



Referensi

1. Ernesto Estrada, "Introduction to Network Theory: Basic Concepts", Institute of Complex Systems at Strathclyde Department of Mathematics, Department of Physics, 2010
2. Dr. Djamel Bouchaffra, "CSE 504 Discrete Structures & Foundations of Computer Science, Ch. 8 (part 1): Graphs"
3. Y. Peng, "Graph", University of Maryland
4. Rinaldi Munir, "Materi Kuliah Matematika Diskrit",Informatika-ITB, Bandung,2003
5. Rinaldi Munir, "Matematika Diskrit",Informatika, Bandung,2001