Introduction to Graphs

Introduction

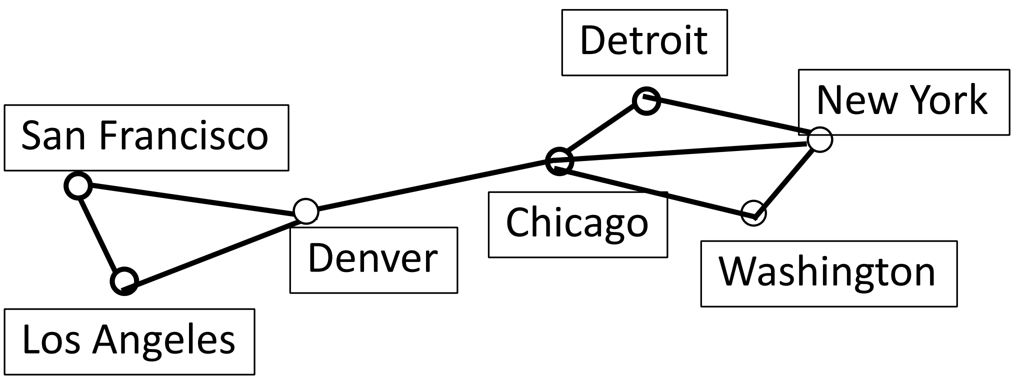
- In computer science, **graph** is an efficient **data structure** for simplication, visualization and analysis of the problems.

- Graphs can represent individual or entire interactions, states, flow, etc. of the elements in the problems.

Applications of Graphs

1. Computer networks

- The network is made up of data centers (represents the location by point) in the cities and communication links between the data centers (represents the links by line segments).



Applications of Graphs

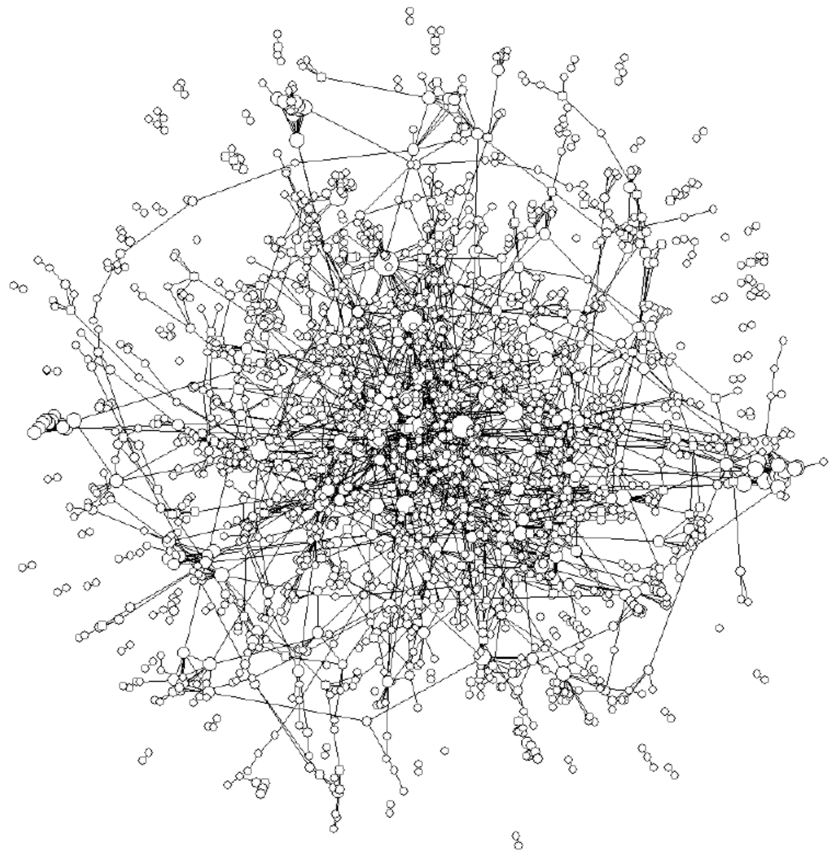
2. Visualization and analysis of protein interactions

- Drawing of the protein-protein physical interaction data with 2167 nodes and 2948 edges.

(protein)

(interaction)

CS+생명과학



Applications of Graphs

4. Visualization and analysis of virus infection route

메르스

http://dj.kbs.co.kr/resources/2015-06-04

simple
multi
psuedo
undirected: 선의 방향성 X

Simple Graph

- A *simple graph* consists of
  - a nonempty set of *vertices*
  - a set of *edges* (unordered pairs of distinct elements of the *vertices*)

∅은 아닌, 점의 집합

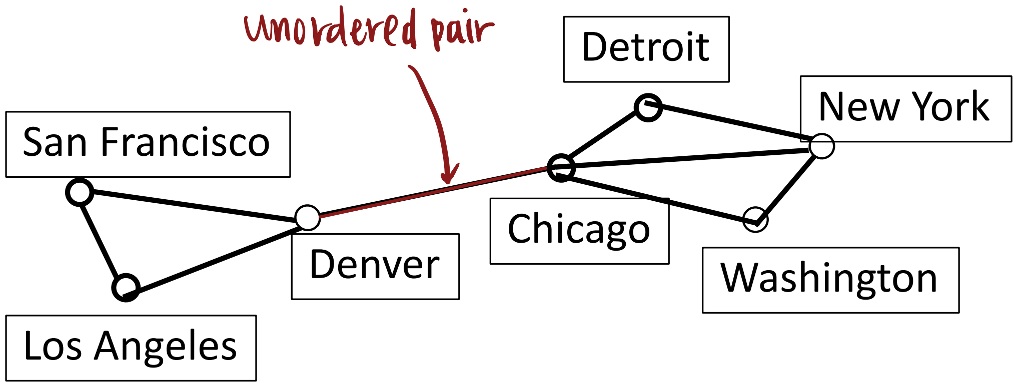
edge 없이 vertex 하나 있어도
graph라고 하기 때문

(① — ② == ② — ①)

단, 같은 vertex (X).

① — ① 안됨 unordered라고 했으므로

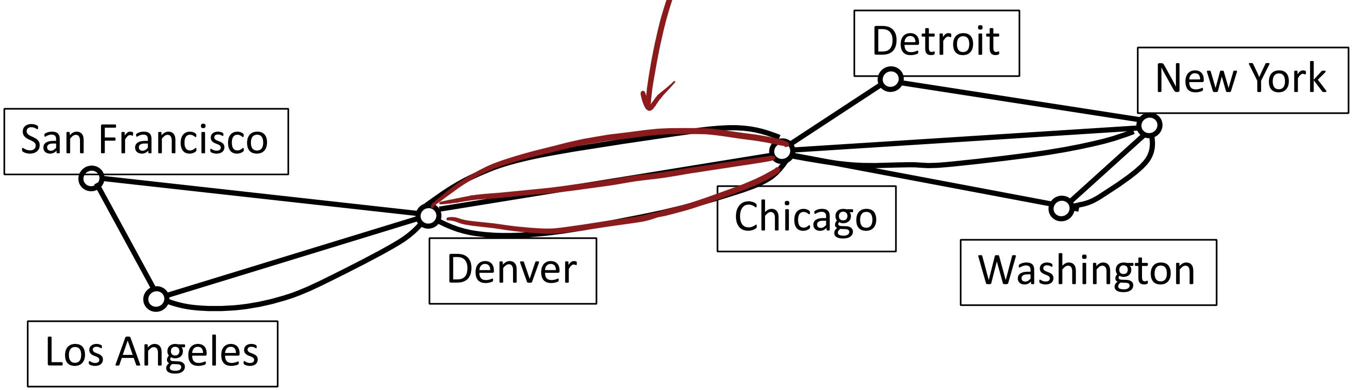
Simple Graph Example



- This simple graph represents a network between cities.
- The network is made up of data centers in the cities and links between the data centers.

Multigraph

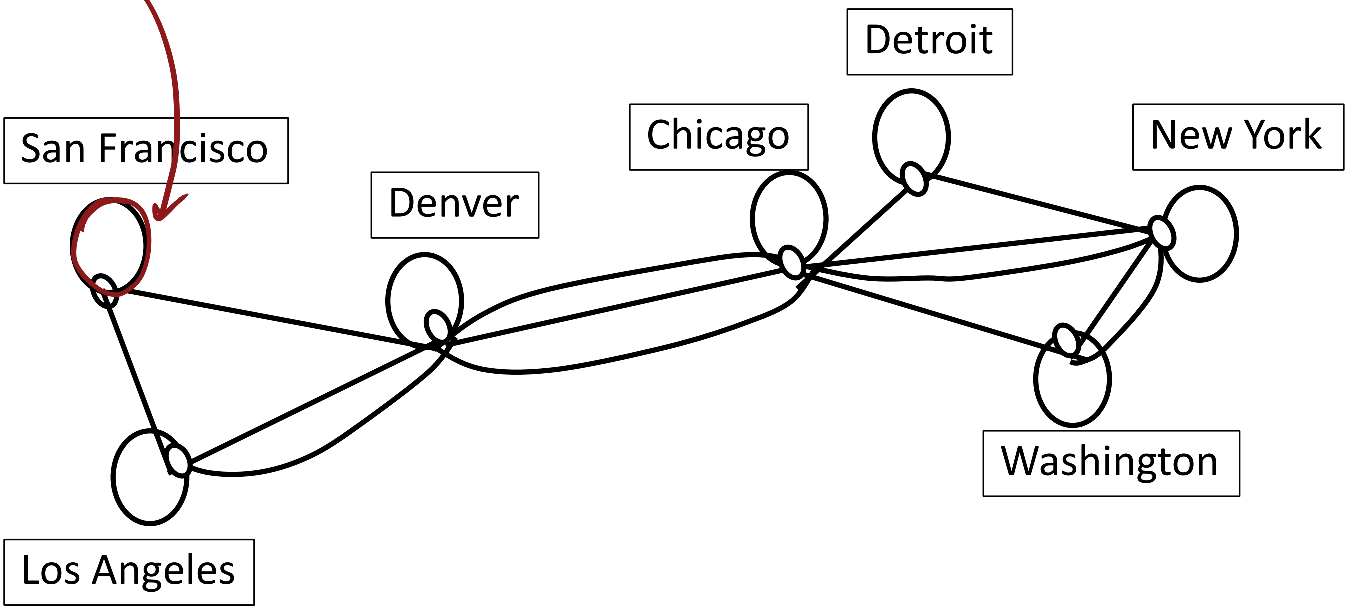
- A *multigraph* can have *multiple edges* (two or more edges connecting the same pair of vertices).



- There can be multiple lines between two data centers in the network.

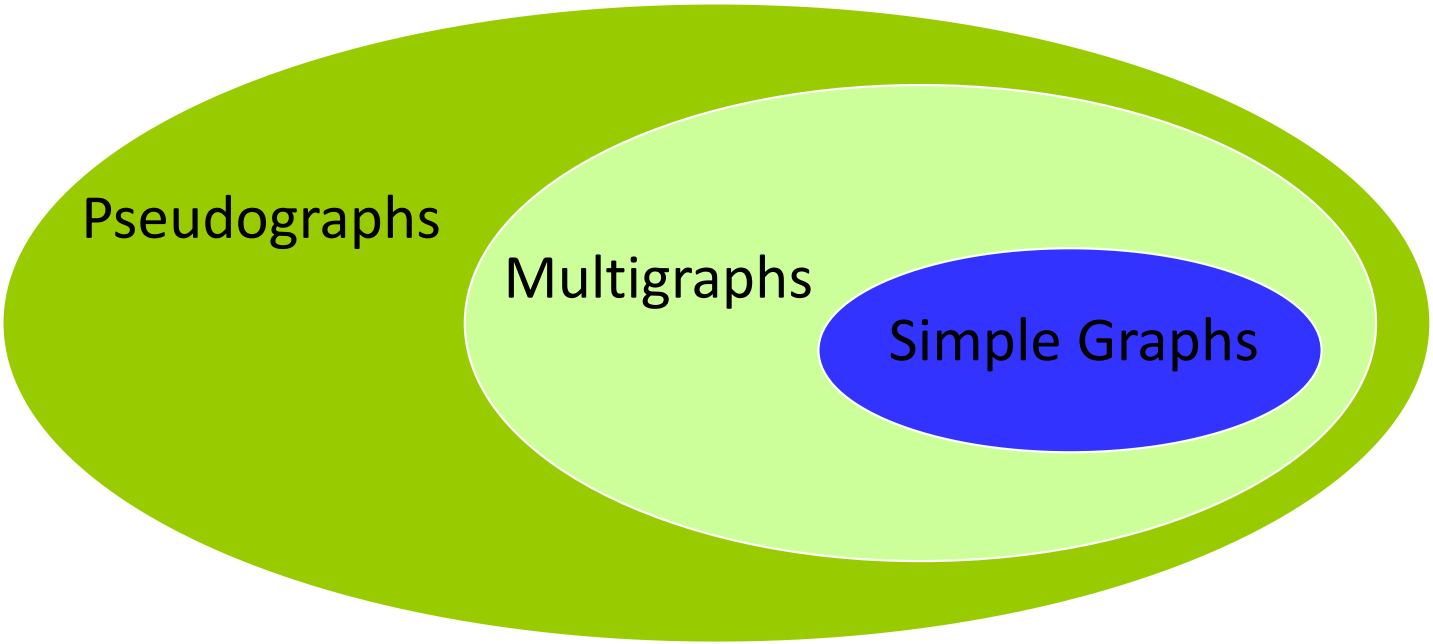
Pseudograph

- A *Pseudograph* can have multiple edges and *loops* (an edge connecting a vertex to itself).



- There can be lines of inner-networks in cities.

Types of Undirected Graphs



Summary

Type	Edges	Loops	Multiple Edges
Simple Graph	Undirected	NO	NO
Multigraph	Undirected	NO	YES
Pseudograph	Undirected	YES	YES

How to Graph a Model

개발시만라지별

- When we build a graph model, we need to make sure that we have correctly answered three key questions about the structure of a graph as following:
    1. Are the edges of graph undirected or directed or both?
    2. If the graph is undirected, are multiple edges present that connect the same pair of vertices? – Is it a multigraph?
    3. Are loops present? – Is it a pseudograph?

Graph Terminology

Vertex and Edge

- A *simple graph* consists of
  - a nonempty set of *vertices* called V
  - a set of *edges* (unordered pairs of distinct elements of V) called E
- Notation: $G = (V,E)$

Adjacent Vertices in Undirected Graphs

- Two vertices, u and v in an undirected graph G are called *adjacent* (or neighbors) in G, if $\{u,v\}$ is an edge of G.

  edge가 있는경우에만

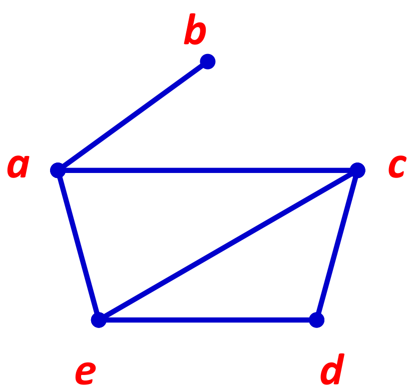
- An edge e connecting u and v is called *incident* with vertices u and v, or is said to

  어떠한일이있다는 면에서

  connect u and v.
- The vertices u and v are called *endpoints* of edge $\{u,v\}$.

Adjacency List

- Specify the vertices that are adjacent to each vertex of the graph



상호치환

Adjacency List	
Vertex	Adjacent Vertices
a	*b*, *c*, *e*
b	*a*
c	*a*, *d*, *e*
d	*c*, *e*
e	*a*, *c*, *d*

Degree of a Vertex in Undirected Graphs

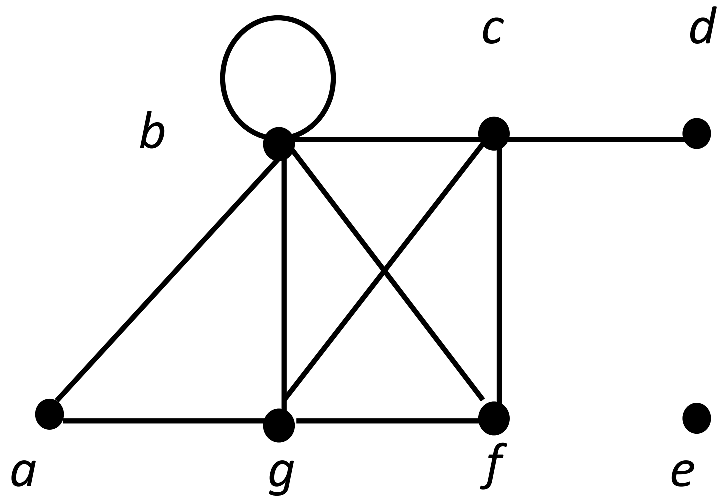
- The *degree of a vertex* in an undirected graph is the number of edges incident with it.

  주의

  들어갔다 나오는것 이기 때문에 +2

  - except that a loop at a vertex contributes twice to the degree of that vertex
- The degree of a vertex *v* is denoted by deg(*v*).

Example



- Find the degrees of all the vertices:

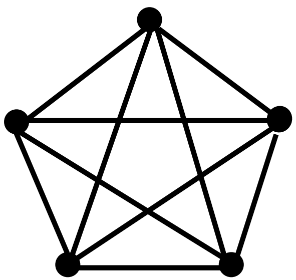
deg(a) = 2, deg(b) = 6, deg(c) = 4, deg(d) = 1,

deg(e) = 0, deg(f) = 3, deg(g) = 4

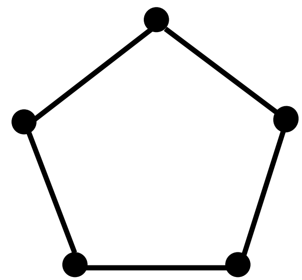
집합.

Subgraph

- A *subgraph* of a graph $G = (V,E)$ is a graph $H = (W,F)$ where $W \subseteq V$ and $F \subseteq E$.



K_5

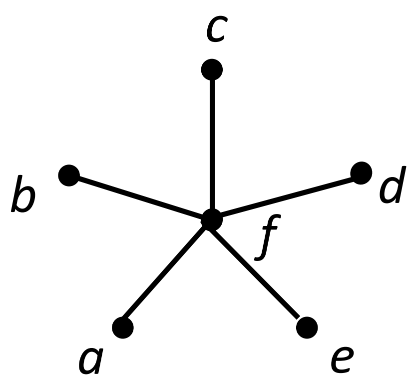


C_5

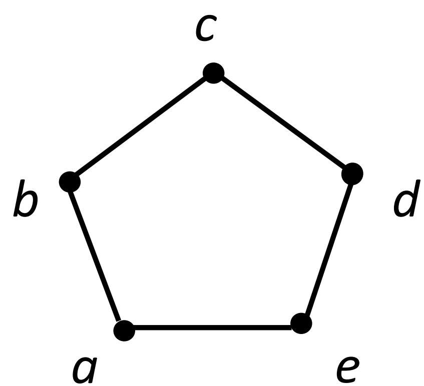
Is C_5 a subgraph of K_5?

Union 합집합

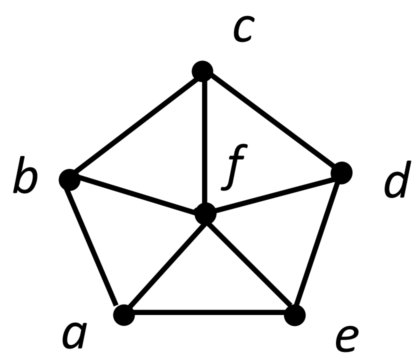
- The *union* of 2 simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the simple graph with vertex set $V = V_1 \cup V_2$ and edge set $E = E_1 \cup E_2$. The union is denoted by $G_1 \cup G_2$.



S_5



C_5



W_5

$$S_5 \cup C_5 = W_5$$

Connectivity

Paths in Undirected Graphs

($V_0 - V_1 - V_2 - \cdots - V_n$)

- There is a ***path*** from vertex v_0 to vertex v_n if there is a sequence of edges from v_0 to v_n
  - This path is labeled as $v_0, v_1, v_2, \ldots, v_n$ and has a **length of *n***. (edge의 길이가 1이라고 가정했을때)
- The path is a ***circuit*** if the path begins and ends with the same vertex.
- ??? A path is ***simple*** if it does not contain the same sequence of two vertices (v_i, v_j) on an edge more than once.

한번만

edge를 한번만 지난다는 뜻이 아님.

즉 $V_i \to V_j \to V_i$ (o) $V_i \to V_j \to V_i \to V_j$ (x)

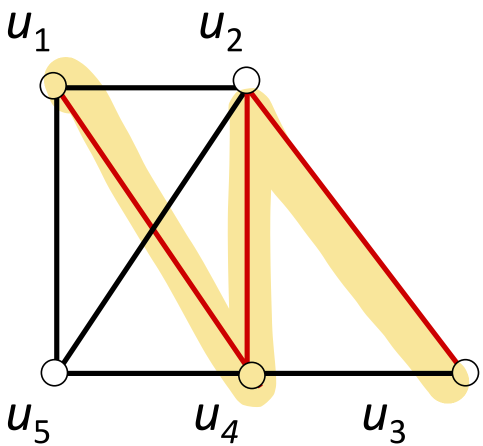
Paths in Undirected Graphs

(vertex)

- A path or circuit is said to ***pass through*** the vertices $v_0, v_1, v_2, \ldots, v_n$ or ***traverse*** the edges $e_1, e_2, \ldots, e_n$.

(edge)

Example



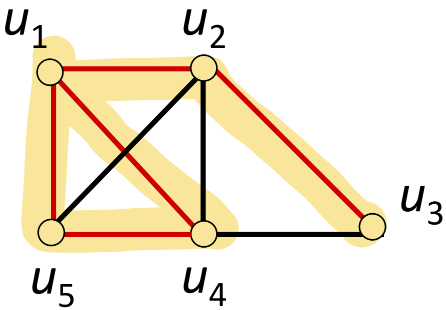
- u_1, u_4, u_2, u_3

  – Is it simple?
  – *yes*

  – What is the length?
  – *3*

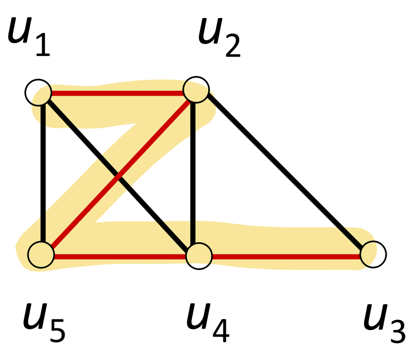
  – Does it have any circuits?
  – *no*

Example



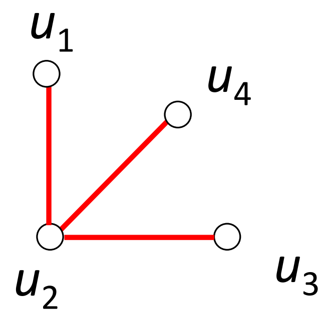
- $u_1, u_5, u_4, u_1, u_2, u_3$
  - Is it simple?
  - *yes*
  - What is the length?
  - *5*
  - Does it have any circuits?
  - Yes; u_1, u_5, u_4, u_1

Example



- u_1, u_2, u_5, u_4, u_3
  - Is it simple?
  - *yes*
  - What is the length?
  - *4*
  - Does it have any circuits?
  - *no*

Example

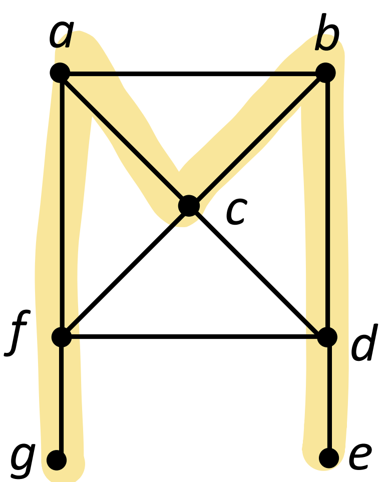


- u_1, u_2, u_4, u_2, u_3
  - Is it simple?
  - *yes* u_2-u_4 와 u_4-u_2는 다르다
  - What is the length?
  - *4*
  - Does it have any circuits?
  - *yes:* u_2, u_4, u_2

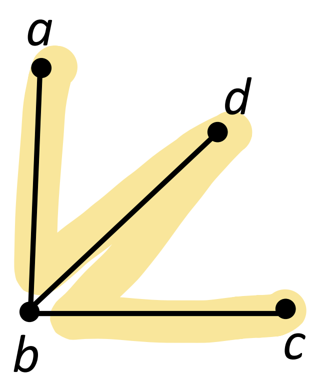
- u_1, u_2, u_1, u_2
  - Is it simple?
  - *no :* it contains the same sequence of two vertices (u_1, u_2) twice.
  - What is the length?
  - *3*
  - Does it have any circuits?
  - *yes:* u_1, u_2, u_1 or u_2, u_1, u_2

Connectedness

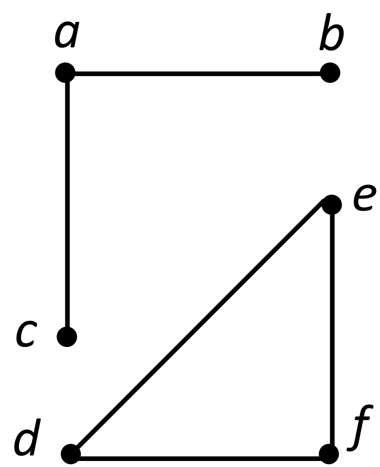
- An undirected graph is called ***connected*** if there is a simple path to pass through every vertices.
- Example
  - Are the following graphs connected?



Yes



Yes

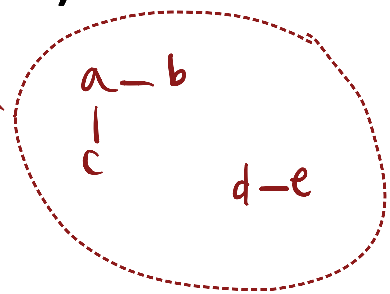


No

Connectedness (Cont.)

- A graph that is not connected is the union of two or more disjoint connected subgraphs (called the ***connected components*** of the graph).

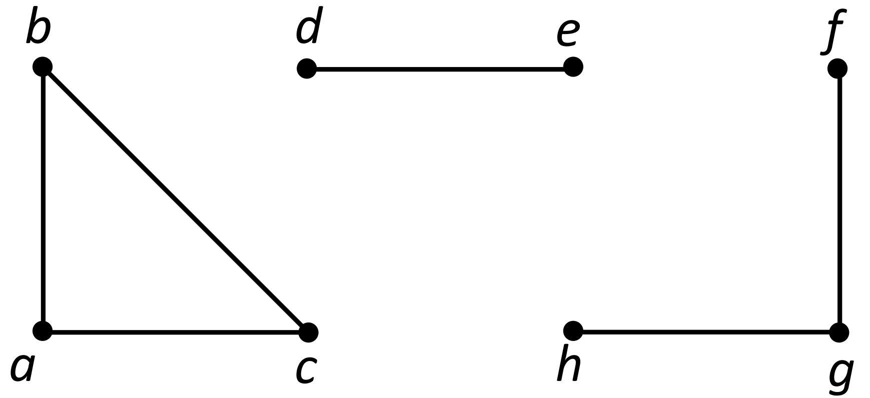
연결되지 않은 그래프는
연결된 그래프의 합
(connected components)

ex


connected component를 찾아서 특징화하는 것이 중요.
connected로 바꾸려면 components만 연결하면 됨

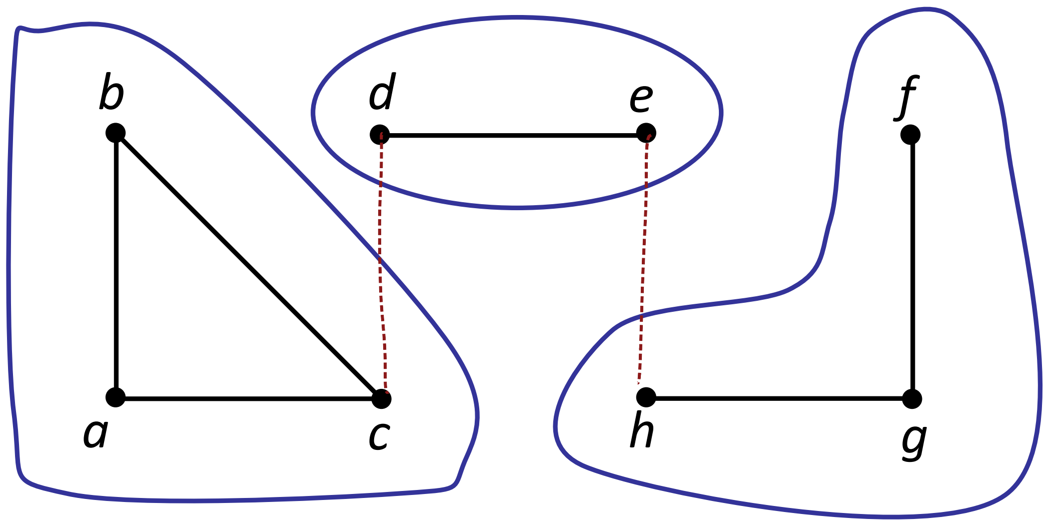
Example

- What are the connected components of the following graph?



Example

- What are the connected components of the following graph?



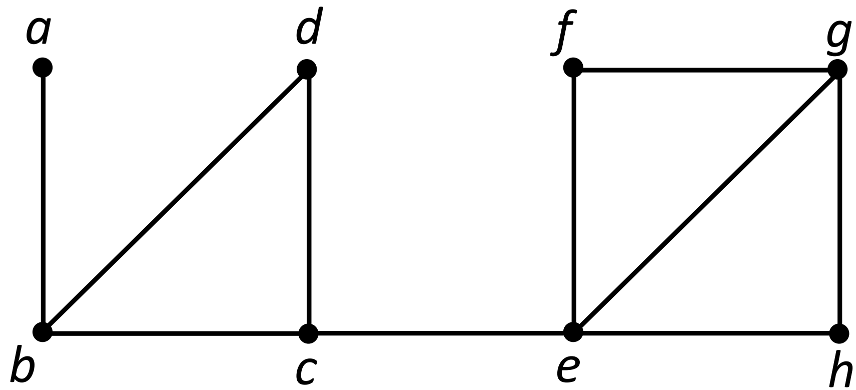
{a, b, c}, {d, e}, {f, g, h}

Cut edges and vertices

- If one can remove a vertex (and all incident edges) and produce a graph with more connected components, the vertex is called a ***cut vertex***.
- If removal of an edge creates more connected components, the edge is called a ***cut edge***.

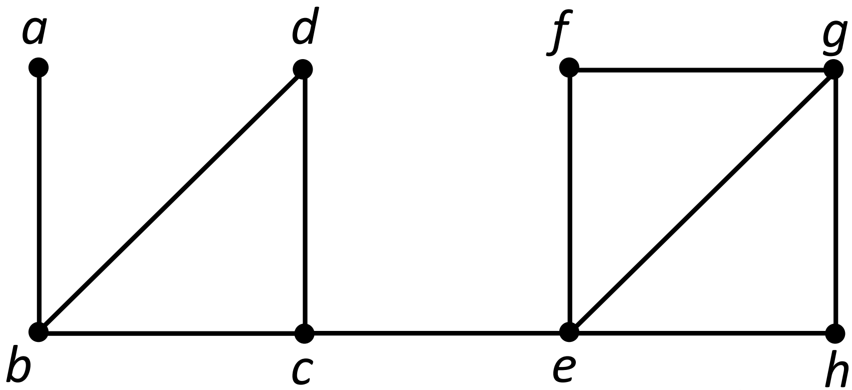
Example

- Find the cut vertices and cut edges in the following graph.



Example

- Find the cut vertices and cut edges in the following graph.



Connected component는 connected sub-graph
→ simple path가 존재한다
→ 최소 2개의 vertice로 그들의 관계를 따짐
∴ 한개의 vertex는 connected component로 볼수 없다 (graph는 맞음)

Cut vertices: c, e, ~~b~~

Cut edge: (c, e), ~~(a, b)~~