Graphs

Instructor: Dr. Sunho Lim (Ph.D., Assistant Professor)

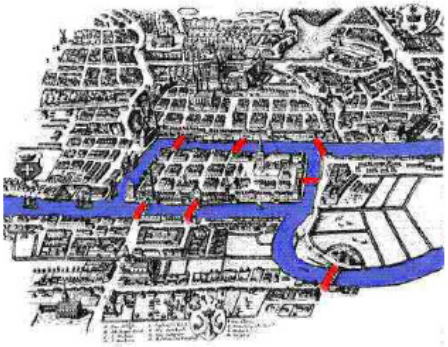
Lecture 13

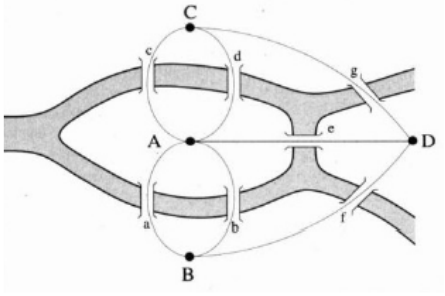
sunho.lim@ttu.edu

Adapted partially from Data Structures and Algorithms in C++, Adam Drozdek, 4th Edition, Cengage Learning; and Algorithms and Data Structures, Douglas Wilhelm Harder, Mmath

Introduction

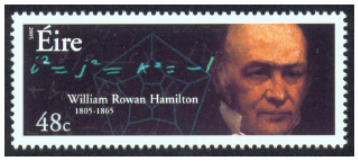
- Graph theory
  - start with Euler who was asked to find a nice path across the seven Königsberg bridges
  - the (Eulerian) path should cross over each of the **seven bridges exactly once**

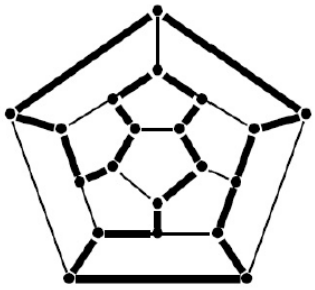




Introduction (cont.)

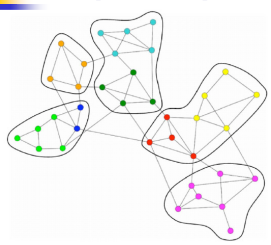
- Another early bird,
  - Sir William Rowan Hamilton (1805-1865)
  - In 1859, developed a toy based on finding a path **visiting all cities** in a graph **exactly once** and sold it to a toy maker in Dublin
  - never was a big success…

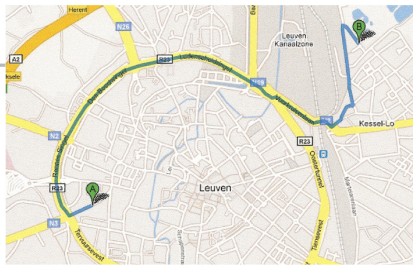


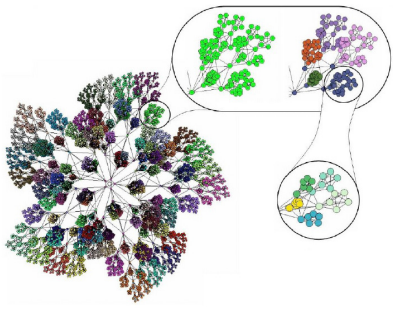


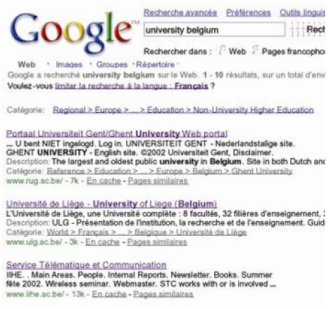
Introduction (cont.)

- Now used in everywhere…
  - large scale problem, ranking, shortest path, etc.







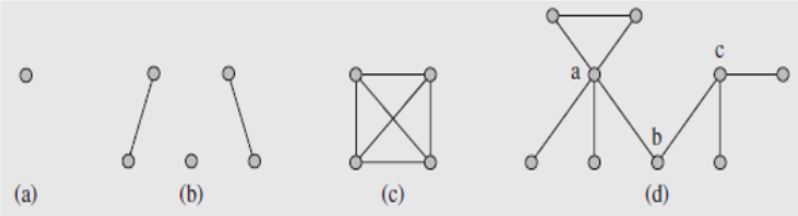


Introduction (cont.)

- Trees
  - quite flexible, but inherent limitation -- only express hierarchical structures
- Graphs
  - a collection of nodes and the connections between them
  - **generalize** a **tree**

Terminologies

- A ***simple graph***
  - $G = (V, E)$ consists of a (finite) set denoted by V, and a collection E, of **unordered pairs** $\{u, v\}$ of distinct elements from V
  - V, called a ***vertex*** or a ***point*** or a ***node***
  - E, *called* an ***edge*** or a ***line*** or a ***link***
  - The number of vertices, $|V|$, and edges, $|E|$



Terminologies (cont.)

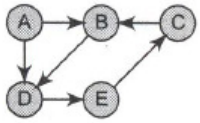


Figure 13.2 Directed graph

- A ***directed graph, digraph***
  - G = (V, E), **(v_i, v_j) != (v_j, v_i)**
  - In a simple graph (undirected graph), (v_i, v_j) = (v_j, v_i)
- A **multigraph**
  - two vertices can be joined by **multiple edges**
- A **pseudograph**
  - a multigraph allowing for **loops**
  - a vertex can be joined with itself by an edge

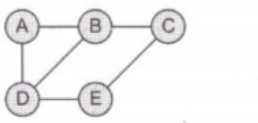
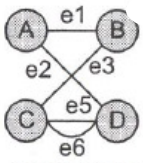
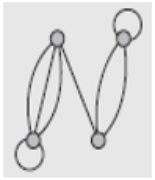


Figure 13.1 Undirected graph



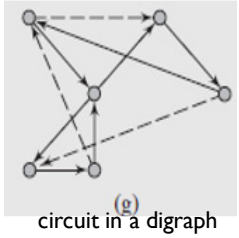
(a) Multi-graph



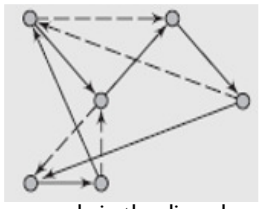
pseudograph

Terminologies (cont.)

- A ***path***
  - a sequence of edges, edge(v1, v2), edge(v2, v3), …, edge(v_{n-1}, v_n)
  - denoted as path v1, v2, v3, …, v_{n-1}, v_n
  - if v1 = v_n and **no edge is repeated**,
    - **circuit**
  - If the vertices in a circuit are different,
    - **cycle**



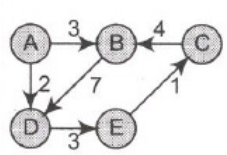
circuit in a digraph



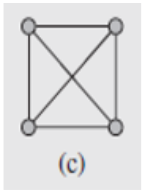
cycle in the digraph

Terminologies (cont.)

- A **weighted graph**
  - an assigned number (e.g., weight, cost, distance, length, etc.) on each edge
- A **complete graph**
  - **exactly one edge** between each pair of distinct vertices



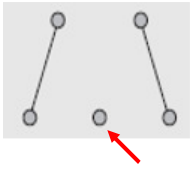
(c) Weighted graph



complete graph

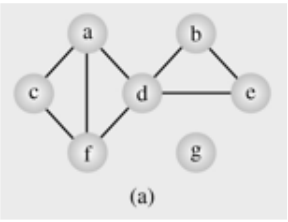
Terminologies (cont.)

- A ***subgraph G'*** of graph G = (V, E),
  - *G'* = (*V'*, *E'*), where V' € V and E' € E
- V_i and V_j are **adjacent**,
  - if the edge(V_i, V_j) is in E
  - such an edge is called **incident** with the vertices V_i and V_j
- The **degree** of a vertex v,
  - deg(v), the **number of edges** incident with v
  - if deg(v) = 0, v is an **isolated vertex**

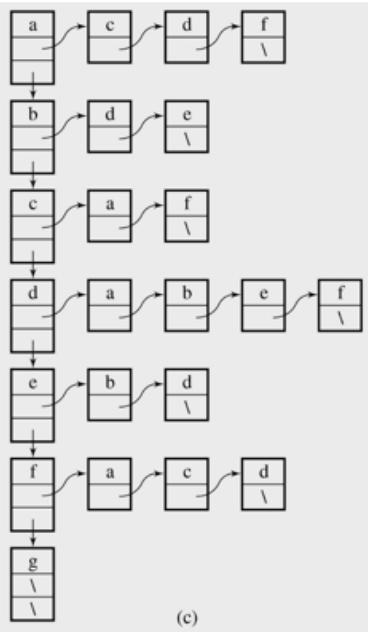


Graph Representation

- An ***adjacency list***
  - specify all vertices adjacent to each vertex of the graph



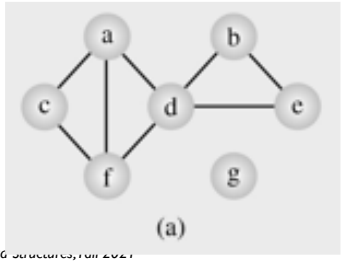
a	c	d	f		
b	d	e			
c	a	f			
d	a	b	e	f	
e	b	d			
f	a	c	d		
g					



Graph Representation (cont.)

- An ***adjacency matrix***
  - a |*V*| x |*V*| binary matrix where,

$$a_{ij} = \begin{cases} 1 & \text{if there exists an edge } (v_i v_j) \\ 0 & \text{otherwise} \end{cases}$$

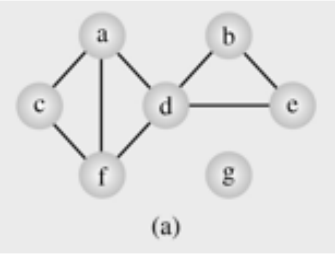


	a	**b**	**c**	**d**	**e**	**f**	**g**
a	0	0	1	1	0	1	0
b	0	0	0	1	1	0	0
c	1	0	0	0	0	1	0
d	1	1	0	0	1	1	0
e	0	1	0	1	0	0	0
f	1	0	1	1	0	0	0
g	0	0	0	0	0	0	0

(d)

Graph Representation (cont.)

- An ***incidence matrix***
  - a |V| x |E| binary matrix where,
  - a_{ij} = 1 if **edge e_j** is **incident** with **vertex v_i**, otherwise 0



(a)

	ac	ad	af	bd	be	cf	de	df
a	1	1	1	0	0	0	0	0
b	0	0	0	1	1	0	0	0
c	1	0	0	0	0	1	0	0
d	0	1	0	1	0	0	1	1
e	0	0	0	0	1	0	1	0
f	0	0	1	0	0	1	0	1
g	0	0	0	0	0	0	0	0

(e)