CMSC/MATH 420: Graph Theory

Spring 2018

Week 1

Jan. 23rd: Syllabus and definitions

Definition 1. A *graph* (or simple graph) is a pair of sets $(\mathbf{V}, \mathbf{E})$ such that $\mathbf{V} \neq \varnothing$ and $\mathbf{E}$ is a set of two-element subsets of $\mathbf{V}$. The elements of $\mathbf{V}$ are *vertices* and $|\mathbf{V}| = n$. The elements of E are *edges* and $|\mathbf{E}| = m$.

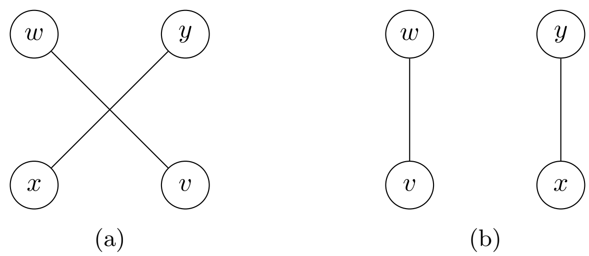


Figure 1: Simple graphs

Definition 2. A *multi-graph* is a pair of sets $(\mathbf{V}, \mathbf{E})$ such that $\mathbf{V} \neq \varnothing$ and E is a finite family of unordered pairs of $\mathbf{V}$.

Definition 3. A *digraph* (or directed graph) is a pair $(\mathbf{V}, \mathbf{A})$ with $\mathbf{V} \neq \varnothing$ and $\mathbf{A}$ is a set of ordered pairs of elements of $\mathbf{V}$ the elements of $\mathbf{V}$ are nodes and the elements of $\mathbf{A}$ are arcs.

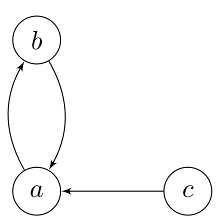


Figure 2: Directed graph

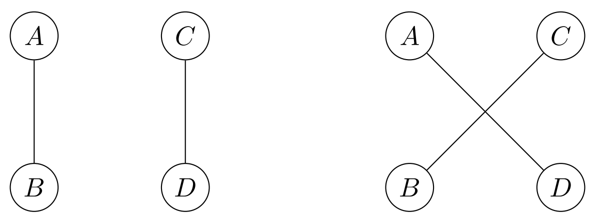
Definition 4. If $\mathbf{G}$ is a graph with vertices u & v and $\{u, v\}$ is an edge, then vertex u is said to join v. The vertices u & v are *adjacent*. The vertices u & v are *incident* to the edge $\{u, v\}$. Two edges are *adjacent* if they have a vertex in common. The degree of a vertex v is the number of edges incident to v and is denoted $\mathbf{deg}(v)$.

Definition 5. If $\mathbf{deg}(v) = 0$, then v is *isolated.* If $\mathbf{deg}(v) = 1$, then v is an *end or pendent* vertex.

Theorem 1 (Handshaking Lemma)**.** *In any graph, the number of vertices of odd degree is even.*

Jan. 25th: Definitions and Theorems

Definition 6. Two graphs, $\mathbf{G}_1$ and $\mathbf{G}_2$, are *isomorphic* if there is a bijective correspondence between the vertices of $\mathbf{G}_1$ and $\mathbf{G}_2$ that preserves adjacencies.



(a) $\mathbf{G}_1$ (b) $\mathbf{G}_2$

A ⇔ A
B ⇒ D
C ⇔ C
D ⇒ B

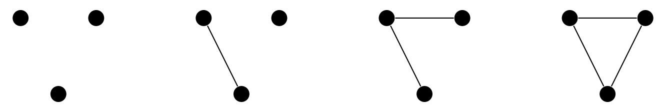
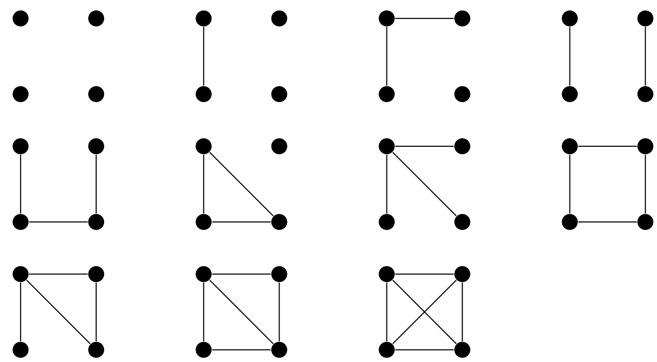
(c) Mapping

Figure 3: Isomorphic Graphs

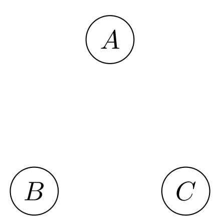
Theorem 2. *Isomorphisms are equivalence relationships.*

- *Reflexive* $\boldsymbol{G}_1 \sim \boldsymbol{G}_1$
- *Symmetric* *If* $\boldsymbol{G}_1 \sim \boldsymbol{G}_2$ *then* $\boldsymbol{G}_2 \sim \boldsymbol{G}_1$
- *Transitive* *If* $\boldsymbol{G}_1 \sim \boldsymbol{G}_2$ *and* $\boldsymbol{G}_2 \sim \boldsymbol{G}_3$ *then* $\boldsymbol{G}_1 \sim \boldsymbol{G}_3$

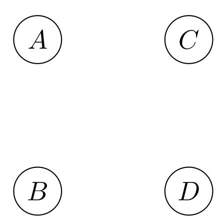
Table 1: Table of isomorphisms for vertices 1 through 4

One Vertex	
Two Vertices	
Three Vertices	
Four Vertices	

Definition 7. $\mathbf{N}_n$ is the *NULL graph* which contains n vertices and no edges.

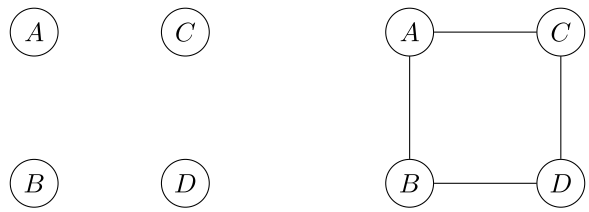


(a) $\mathbf{N}_3$



(b) $\mathbf{N}_4$

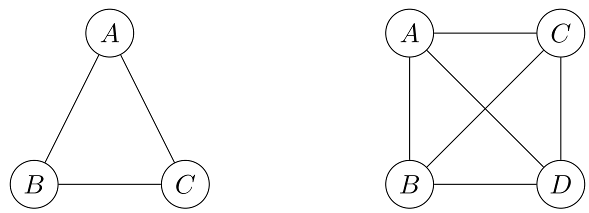
Definition 8. A graph is *regular* of degree r if each vertex has degree r.



(a) Regular of degree 0 (b) Regular of degree 2

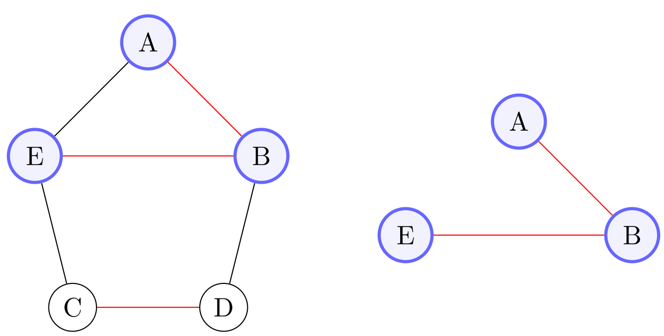
Definition 9. A *cubic graph* is regular of degree 3.

Definition 10. $\mathbf{K}_n$ is the *complete graph* meaning it is regular of degree $n-1$



(a) $\mathbf{K}_3$ (b) $\mathbf{K}_4$

Definition 11. A *subgraph* of **G** have vertices **H** and edges **F** where $\mathbf{H} \subseteq \mathbf{V}$ and $\mathbf{F} \subseteq \mathbf{E}$ but only the edges of **E** that correspond to the vertices in **H** are used.



(a) **G** with **H** and **F** marked (b) The subgraph of **G**

Week 2

Jan. 30th: Definitions and Notes

Definition 12. A *bipartite* graph $\mathbf{G}$ is a graph whose vertex set can be written as $\mathbf{V} = \mathbf{V}_1 \cup \mathbf{V}_2$ where $\mathbf{V}_1 \cap \mathbf{V}_2 = \emptyset$ and $\mathbf{V}_1 \neq \emptyset$ and $\mathbf{V}_2 \neq \emptyset$. Each edge of $\mathbf{E}$ joins a vertex from $\mathbf{V}_1$ to $\mathbf{V}_2$. $m = rs$ where m is the number of edges

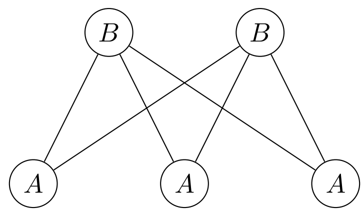
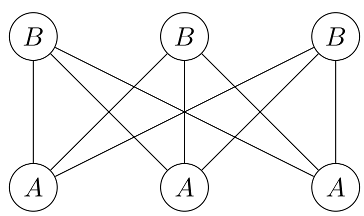
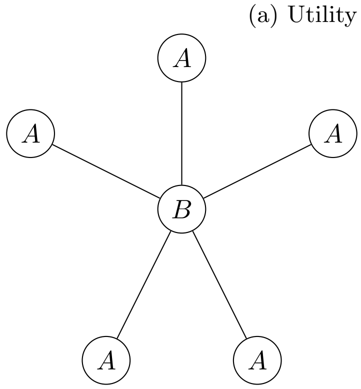


Figure 8: Bipartite Graph $\mathbf{K}_{2,3}$

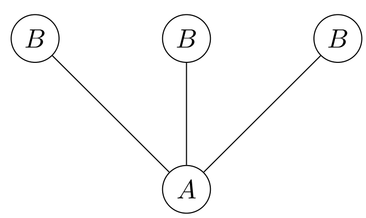
Definition 13. A *complete bipartite* graph $\mathbf{K}_{r,s}$ $|\mathbf{V}_1| = r$ and $\mathbf{V}_2 = s$



(a) Utility Graph $\mathbf{K}_{3,3}$



(b) Star Graph $\mathbf{K}_{1,5}$



(c) Claw Graph $\mathbf{K}_{1,3}$

How to store a graph: sets adjacency matrix pointer structure/nodes

Table 2: Structure

label/name? value? What is it connected to? Vector/Linked list of pointers?

Definition 14. *Adjacency Matrix*: Let $\mathbf{G}$ be a graph with vertices $\mathbf{V}_1, ..., \mathbf{V}_n$ $\mathbf{A(G)} = \mathbf{A} = [\mathbf{a}_{ij}]$ is nxn matrix where $\mathbf{a}_{ij}$ is the number of edges which join $\mathbf{V}_i$ and $\mathbf{V}_j$.

$$\begin{pmatrix} * & A & M & T & J & Q \\ A & 0 & 1 & 1 & 0 & 0 \\ M & 1 & 0 & 1 & 0 & 0 \\ T & 1 & 1 & 0 & 0 & 0 \\ J & 0 & 0 & 0 & 0 & 1 \\ Q & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

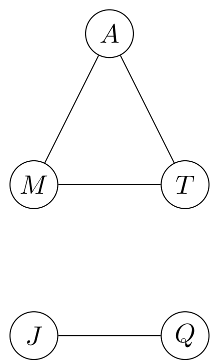


Figure 10: $\mathbf{G}$

If a directed graph, adjacency matrix may not be symmetrical. Size is potentially better for pointers because it doesn't count non-edges. If you want to know where there are not edges, adjacency matrix is better.

Definition 15. If edges are labeled, you can use an *incidence matrix*. $\mathbf{M} = [\mathbf{m}_{ij}]$ is an nxm matrix matrix where $\mathbf{m}_{ij} = \begin{cases} 1, & \text{if } \mathbf{v}_i \text{ is incident to edge} \\ 0, & \text{if otherwise} \end{cases}$

$$\begin{pmatrix} * & A & M & T & J & Q \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 & 0 \\ 4 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$\frac{n(n-1)}{2}$ = max number of edges in a complete graph

Definition 16. Let $\mathbf{G}_1$ and $\mathbf{G}_2$ be graphs with vertices $\mathbf{V}(\mathbf{G}_1)$ and $\mathbf{V}(\mathbf{G}_2)$ and edges $\mathbf{E}(\mathbf{G}_1)$ and $\mathbf{E}(\mathbf{G}_2)$. The *union* of two graphs $\mathbf{G}_1 \cup \mathbf{G}_2$, is $\mathbf{V}(\mathbf{G}_1) \cup \mathbf{V}(\mathbf{G}_2)$ and $\mathbf{E}(\mathbf{G}_1) \cup \mathbf{E}(\mathbf{G}_2)$.

Definition 17. A graph is *disconnected* if it is the union of two graphs.

Definition 18. A graph is *connected* if it is not disconnected.

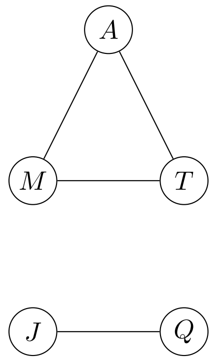


Figure 11: **G**

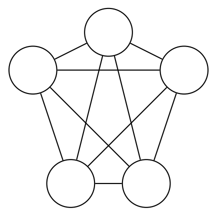
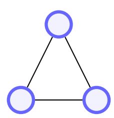
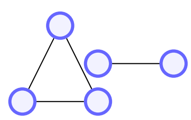


Figure 12: Max Edges $\mathbf{K}_5$



(a) $\mathbf{G}_1$

(b) $\mathbf{G}_2$



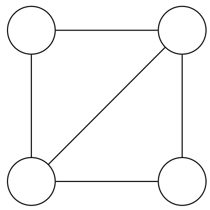
(c) Separate Components

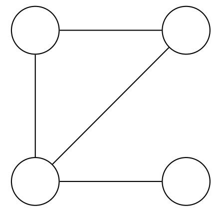
Definition 19. If **F** is a set of edges of **G** then **G-F** is the graph resulting in the removal of **f** edges.

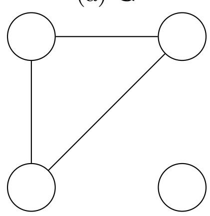
If **S** is a set of vertices then $\mathbf{G} - \mathbf{S}$ is the graph resulting from removing vertices in **S** and incident edges.

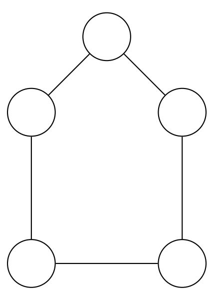
Definition 20. A connected graph that is regular of degree 2 is called a *Cycle*, $\mathbf{C}_n$

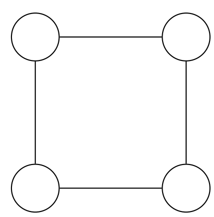
Compliment matrix, in terms of adjacency matrix, all 1's become 0's.

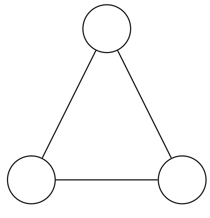

(a) $\mathbf{G}$


(b) $\mathbf{G} - 1$


(c) $\mathbf{G} - 1, 2$

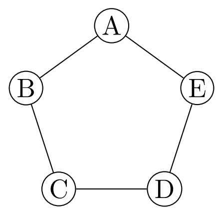

(a) $\mathbf{C}_5$


(b) $\mathbf{C}_4$

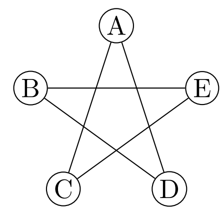

(c) $\mathbf{C}_3$

Feb. 1st: Definitions and Theorems

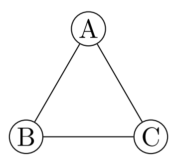
Definition 21. A *compliment* of a graph **G** is the graph **G'** with the same vertices are adjacent in **G'** if and only if they are not adjacent in **G**.



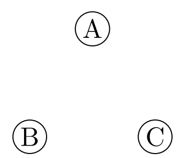
(a) **G**



(b) $\mathbf{G}'$



(c) **H**

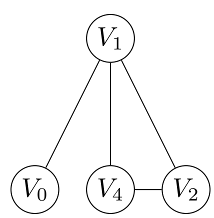


(d) $\mathbf{H}'$

Figure 16: Graphs and their complements

Definition 22. A *walk* in a graph is any finite sequence of edges, $v_0\ v_1$, $v_1\ v_2$, $v_2\ v_3$,. . . , $v_{k-1}\ v_k$.

- The initial vertex is v_0
- The final vertex is v_k



(a) **J**

(b) $V_1 = V_3$

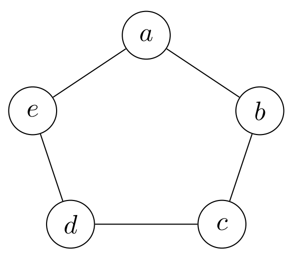
Definition 23. If all edges are distinct, we have a trail.

Definition 24. If vertices are distinct, we have a path.

Definition 25. If V_0=V_k, we have a cycle. A cycle is a path with at least one edge where the first vertex (V_0) and the last vertext (V_k) are the same.

Theorem 3. *Adjacency Matrix Theorem: Let A be an adjacency matrix for a graph G with n vertices. If A^K = $[n_{ij}]$, then n_{ij} is the number of walks of length K from V_i to V_j.*

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



(a) **G**

Induction :

- Base case.
- Assume **k** case is true, or **k-1** is true.
- Prove that **k+1** case is trueby using the **k** case.

Proof. Base Case:
If $\mathbf{n}_{ij}$ = **1**, then there is an aedge from $\mathbf{v}_i$ to $\mathbf{v}_j$. If $\mathbf{n}_{ij}$ = **0**, then there is no edge from $\mathbf{v}_i$ to $\mathbf{v}_j$. We've already shown that A^1 works, so our base case is done. If n_{ij}=1, then there is an edge from i to j. If n_{ij}=0, then there is no edge from i to j.
Assume that the k case: $\mathbf{A}^k$ = $[\mathbf{n}_{ij}]$. $\mathbf{n}_{ij}$ is the walks of length from **k** from $\mathbf{v}_i$ to $\mathbf{v}_j$.

$$\begin{pmatrix}0&1&0&0&1\\1&0&1&0&0\\0&1&0&1&0\\0&0&1&0&1\\1&0&0&1&0\end{pmatrix}\cdot\begin{pmatrix}0&1&0&0&1\\1&0&1&0&0\\0&1&0&1&0\\0&0&1&0&1\\1&0&0&1&0\end{pmatrix}=\begin{pmatrix}2&0&1&1&0\\0&2&0&1&1\\1&0&2&0&1\\1&1&0&2&0\\0&1&1&0&2\end{pmatrix}$$

3.) Show for **k+1**:
So $\mathbf{A}^{k+1} = \mathbf{A}^{k} = \mathbf{A}^{1}$.
$\mathbf{A}^{k+1} = [\mathbf{t}_{ij}]$
$\mathbf{t}_{ij} = \sum_{l=1}^{n} \mathbf{a}_{il}\ \mathbf{n}_{lj}$. □

Week 3

Feb. 6th: Definitions and Notes

Definition 26. A *Jordan Curve* is a continuous curve with no intersections. A closed Jordan Curve is one whose end points coincide.

Definition 27. A graph can be *embedded* in a space **T** if it is isomorphic to a graph drawn in **T** with points representing vertices and Jordan curves respesenting edges such that there are no crossings.

Definition 28. A *crossing* occurs when:

- Two Jordan Curves intersect at a point that is not a vertex.
- An edge passes through a vertex that is not incident to the edge.

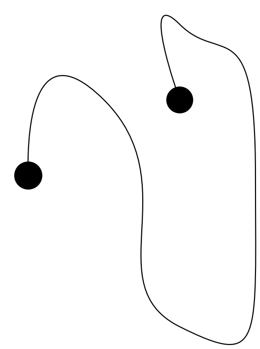


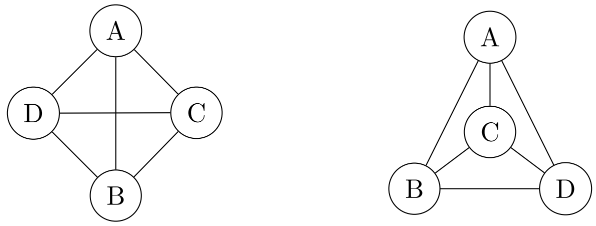
Figure 19: **Jordan Curve**

Definition 29. A graph is *planar* if and only if it can be embedded on the surface of a sphere.

Definition 30. A graph is *plane* if its picture is a planar graph.

Theorem 4. *All graphs can be embedded in 3D. For each edge, put the edge on a page of a book. List the vertices along the spine of the book.*

Theorem 5. *A graph is planar if and only if it can be embedded on the surface of a sphere.*



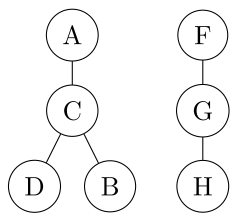
(a) With crossings (b) Without crossings

Figure 20: A *plane* graph

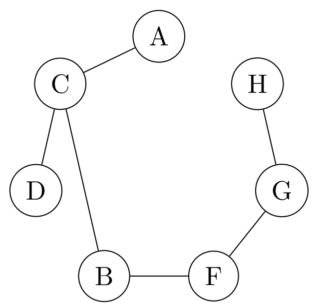
Feb. 8th: Definitions and Notes

Definition 31. A *Forest* is a graph without cycles.

Definition 32. A *Tree* is a connected forest.



(a) A *forest*



(b) A *tree*

Algorithm 1 Depth First Traversal

procedure DepthFirst(G,v)
  if v is not marked **then**
    v ←marked
    for all w adjacent to v **do**
      DepthFirst(G,w)
    end for
  end if
end procedure

Definition 33. A *weighted graph* is a graph with weights associated with edges.

Definition 34. The *Minimum Spanning Tree* (MST) is the tree with the minimum weight, that reaches every vertex.

Algorithm 2 Breadth First Traversal

```
procedure BREADTHFIRST(G,v)
    ENQUEUE(Q,v)
    while Q is not empty do
        v ←DEQUEUE(Q)
        v ←marked
        for all w adjacent to v do
            if w is not marked then
                ENQUEUE(Q,w)
            end if
        end for
    end while
end procedure
```

Algorithm 3 Prim's Algorithm

```
procedure PRIM
    v ∈ V
    while v is not NULL do
        w gets NULL
        set w to the closest unmarked vertex
        mark w
        set v to w
    end while
end procedure
```

Algorithm 4 Kruskal's Algorithm

```
procedure KRUSKAL
    for all v ∈ V do
        build a tree with v
    end for
    sort the edges of E by weight
    for all e ∈ E do
        if the vertices of e are in different trees then
            merge trees by adding the edge
        end if
    end for
end procedure
```

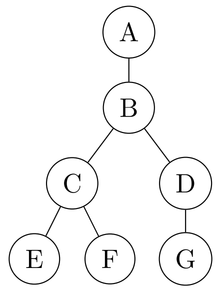


Figure 22: Depth First Traversal: $A \to B \to C \to E \to F \to D \to G$

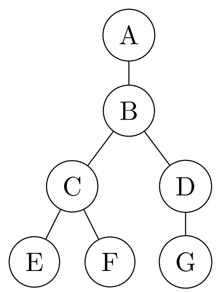


Figure 23: Breadth First Traversal: $A \to B \to C \to D \to E \to F \to G$

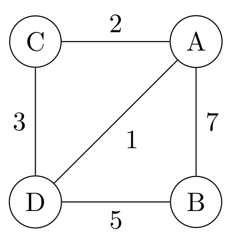


Figure 24: A *weighted* graph

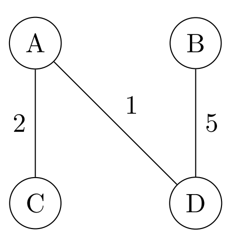


Figure 25: A *weighted* graph