# Notes on Graphs

### Graphs

- A **graph** $G$ consists of a finite nonempty set $V$ of **vertices** and a set $E$ of **edges**, where an edge is a unordered pair of distinct vertices of $V$.
- We denote a graph $G$ with the vertex set $V$ and the edge set $E$ by $G = (V,E)$. And the vertex set of a graph $G$ is denoted by $V(G)$ and the edge set of $G$ is denoted by $E(G)$.
- Given that $e = (u,v) \in E$, we say that $u$ is **adjacent** to $v$ and $u$ (or $v$) is **incident** to $e$.
- A **directed graph** (or **digraph**) consists of a finite nonempty set $V$ of **vertices** and a set $E$ of **directed edges** (or **arcs**), where a directed edge is an ordered pair of distinct vertices of $V$.

### Subgraphs

- A **subgraph** of $G = (V,E)$ is a graph having a vertex set $V' \subseteq V$ and an edge set $E' \subseteq E$ such that, for each $(u,v) \in E'$, $u,v \in V'$.
- A **spanning subgraph** is a subgraph containing all the vertices of $V$.
- For any set $S \subseteq V$, the **induced subgraph** $G\langle S\rangle$ is the maximal subgraph of $G$ with vertex set $V$.
- Thus, $(u,v) \in E(G\langle S\rangle)$ if and only if $(u,v) \in E(G)$

### Graph isomorphism

- Two graphs $G$ and $H$ are **isomorphic** if there exists a one-to-one correspondence $\theta$ between $V(G)$ and $V(H)$ which preserves adjacency, i.e.

$$(u,v) \in E(G) \quad \Leftrightarrow \quad (\theta(u),\theta(v)) \in E(H).$$

- An **invariant** of a graph $G$ is a number associated with $G$ which has the same value for any graph isomorphic to $G$.
- A **complete set of invariants** determines a graph up to isomorphism.

### Walks and connectedness

- A **walk** of a graph is an alternating sequence of vertices of edges $\langle v_0, e_0, v_1, \cdots, v_{n-1}, e_n, v_n\rangle$, beginning and ending with vertices, in which each edge is incident to the two vertices immediately preceding and following it.
- A walk is **closed** if $v_0 = v_n$ and is **open** otherwise.
- A walk is a **trail** if all the edges are distinct.
- A walk is a **path** if all the vertices (and thus necessarily all the edges) are distinct.
- If the walk is closed, then it is a **cycle** if its $n$ vertices are distinct and $n \geq 3$.
- We denote by $C_n$ the graph consisting of a cycle with $n$ vertices.
- We denote by $P_n$ a path with $n$ vertices.
- **Theorem**: The edge set of a graph can be partitioned into cycles if and only if every vertex has even degree.
- A graph is **connected** if every pair of vertices are joined by a path.
- A maximally connected subgraph of $G$ is called a **connected component** (or **component**) of $G$.
- The **length** of a walk $\langle v_0, e_0, v_1, \cdots, v_n\rangle$ is $n$, the number of edges in it.
- The **girth** of a graph $G$, denoted $g(G)$, is the length of a shortest cycle in $G$.
- The **circumference** of a graph $G$, denoted $c(G)$ is the length of any longest cycle.
- The **distance** $d(u,v)$ between two vertices $u$ and $v$ in $G$ is the length of a shortest path joining them if any. Otherwise, $d(u,v) = \infty$.
- **Theorem**: In a connected graph, distance is a *metric*; that is, for all vertices $u,v$ and $w$,
  (a) $d(u,v) \geq 0$, with $d(u,v) = 0$ iff $u = v$.
  (b) $d(u,v) = d(v,u)$.
  (c) $d(u,v) + d(v,w) \geq d(u,w)$.
- The **diameter** $d(G)$ of a connected graph $G$ is the length of any longest $u$-$v$ path.
- The **square** $G^2$ of a graph $G$ has $V(G^2) = V(G)$ with $u,v$ adjacent in $G^2$ whenever $d(u,v) \leq 2$ in $G$.
- Adjacency matrices for powers $G^2, G^3, \cdots$ can be obtained by multiplying the adjacency matrix $A$ for $G$. Actually, $A^k_{uv}$ contains the number of distinct paths between $u$ and $v$ whose length is $\leq k$. Adjacency matrix $G^n$ is essentially a transitive closure.

### Degrees

- The **degree** of a vertex $v_i$ in graph $G$, denoted by $d_i$ or degree$(v_i)$, is the number of edges incident to $v_i$.
- **Theorem**: The sum of the degrees of the vertices of a graph $G = (V,E)$ is twice the number of edges,

$$\sum_{v \in V} \text{degree}(v) = 2|E|.$$

- **Corollary**: In any graph, the number of vertices of odd degree is even.
- The minimum degree among the vertices of $G$ is denoted $\delta(G)$.
- The maximum degree among the vertices of $G$ is denoted $\Delta(G)$.
- If $\delta(G) = \Delta(G) = r$, then all vertices have the same degree and $G$ is called **regular** of degree $r$.
- 3-regular graphs are called **cubic**.
- **Corollary**: Every cubic graph has an even number of vertices.

## Trees: Characterization

- A graph is **acyclic** if it has no cycles.
- A **tree** is a connected acyclic graph.
- Any graph without cycles is a **forest**, where the components of a forest are trees.
- **Theorem**: The following statements are equivalent for a graph $G$:

  (a) $G$ is a tree.

  (b) Every two vertices of $G$ are joined by a unique path.

  (c) $G$ is connected and $m = n - 1$.

  (d) $G$ is acyclic and $m = n - 1$.

  (e) $G$ is acyclic and if two nonadjacent vertices of $G$ are joined by an edge $e$, then $G + e$ has exactly one cycle.

  (f) $G$ is connected, is not $K_n$ for $n \geq 4$, and if any two nonadjacent vertices of $G$ are joined by an edge $e$, then $G + e$ has exactly one cycle.

  (g) $G$ is not $K_3 \cup K_1$ or $K_3 \cup K_2$, $m = n - 1$, and if any two nonadjacent vertices of $G$ are joined by an edge $e$, then $G + e$ has exactly one cycle.

- **Corollary** [1]: Every nontrivial tree has at least two endvertices.

## Trees: Centers and centroids

- The **eccentricity** $e(v)$ of a vertex $v$ in a *connected* graph $G$ is $\max\{d(u,v)\}$ for all $u \in V$ (i.e. distance to the farthest vertex).
- The **radius** $r(G)$ is the minimum eccentricity of the vertices.
- The **diameter** $d(G)$ is the maximum eccentricity of the vertices.
- A vertex $v$ is a **central vertex** of $G$ if $e(v) = R(G)$, and the **center** of $G$ is the set of all central vertices of $G$.
- **Theorem**: Every tree has a center consisting of either one vertex or two *adjacent* vertices.
- A **branch** at a vertex $u$ of a tree $T$ is a maximal subtree containing $u$ as an *endvertex*.
- The number of branches at $u$ is degree$(u)$.
- The **weight** at a vertex $u$ of a tree $T$ is the maximum number of edges in any branch at $u$.
- A vertex $v$ is a **centroid vertex** of a tree $T$ if $v$ has minimum weight, and the **centroid** of $T$ consists of all such vertices.
- **Theorem**: Every tree has a centroid consisting of either one vertex of two *adjacent* vertices.

## Trees: Block-cutvertex trees

- For a connected graph $G$ with blocks $\{B_i\}$ and cutvertices $\{c_j\}$, the **block-cutvertex graph**[1] of $G$, denoted by $bc(G)$, is defined as the graph having vertex set $\{B_i\} \cup \{c_j\}$, with two vertices adjacent if one corresponds to a block $B_i$ and the other to a cutvertex $c_j$ and $c_j$ is in $B_i$.
- A block-cutvertex graph is a bipartite graph.
- **Theorem**: A graph $G$ is the block-cutvertex graph of some graph $H$ if and only if it is a tree in which the distance between any two endvertices is even.

## Independent cycles and cocycles

- A **0-chain** of $G$ is a linear combination $\sum \varepsilon_i v_i$ of vertices and a **1-chain** is a sum $\sum \varepsilon_i e_i$ of edges.
- The **boundary operator** $\partial$ sends 1-chains to 0-chains according to the rules:

  (a) $\partial$ is linear.

  (b) if $e = (u,v)$, then $\partial x = u + v$.

- The **coboundary operator** $\delta$ sends 0-chains to 1-chains by the rules.

  (a) $\delta$ is linear.

  (b) if $\delta v = \sum \varepsilon_i e_i$, where $\varepsilon_i = 1$ whenever $x_i$ is incident with $v$.

## Independence sets

- A subset $S$ of $V$ is an **independence set** of $G$ if no two vertices of $S$ are adjacent in $G$.
- An independent set $S$ is a **maximum independent set** if $G$ has no independent set $S'$ with $|S'| \supseteq |S|$.
- A subset $V'$ of $V$ is a **covering** of $G$ if every edge of $G$ has at least one end in $V'$.
- **maximum independent sets VS minimum vertex covering**
- **Theorem**: A set $S \subseteq V$ is an independent set of $G$ if and only if $V - S$ is a covering of $G$.
- The size of a maximum independence set is called the **independence number** of $G$ and is denoted by $\alpha(G)$.
- The size of a minimum covering of $G$ is the **covering number** of $G$ and is denoted by $\beta(G)$.
- **Theorem**: For any graph $G = (V,E)$, $\alpha(G) + \beta(G) = |V|$.
- An **edge covering** of $G$ is a subset $E'$ of $E$ such that each vertex of $G$ is incident to some edge in $E'$.
- Edge analogue of an independent set is a set of edges which are pairwise non-adjacent, that is, a *matching*.
- **maximum matching VS minimum edge covering**
- We denote the number of edges in a **maximum matching** of $G$ by $\alpha'(G)$ and call it the **edge independence number**.
- We denote the size of minimum edge cover of $G$ by $\beta'(G)$ and call it the **edge covering number**.

[1] For an application, see [2]

- **Theorem**: For any graph $G = (V,E)$, if $\delta > 0$, then $\alpha'(G) + \beta'(G) = |V|$.
- **Theorem**: In a bipartite graph $G$ with $\delta > 0$, the number of vertices in a maximum independent set is equal to the number of edges in a minimum edge cover.

**Depth-first search**

- DFS has a nice feature that partitions the edge set into forward edges, backward edges, tree edges, and cross edges, which can be used for binconnectivity, planarity algorithms.
- **topological sorting**: a vertex is **finished** only after all vertices reachable from it are finished; so ordering vertices in decreasing order of finish time is a topological order
- An **articulation point** (a.k.a. **cut vertex**) is a vertex whose deletion disconnects the remaining graph into multiple components.
- A graph is **biconnected** if there is no articulation point.
- A **biconnected component** of a graph is a maximal subset of edges s.t. the corresponding induced subgraph is biconnected. Typically, an articulation point join different biconnected components.
- **Hopcroft-Tarjan algorithm for biconnected components**:

$$low(v) = \min_{w \in V} \{D[v], D[w]\}$$

  where $D[v]$ is the discover time of $v$, $(u,w)$ is a back edge for some descendent $u$ of $v$. That is $low(v)$ of a vertex $v$ is the discovery number of the vertex closest to the root that can be reached from $v$ by following zero or more tree edges downward and at most one back edge upward.
- **Theorem**: Let $T$ be a DFS tree of a connected graph $G$, and let $v$ be a nonroot vertex of $T$. Vertex $v$ is a **cut vertex** if and only if there is a child $w$ of $v$ in $T$ with $low(w) \geq D[v]$.
- **strong connectivity**: can be found by two DFS over $G$: once for $G$ and second time with $G^T$ (edges of $G$ are reversed).
  (a) call **DFS**($G$) to compute **finish times** $F(v)$
  (b) call **DFS**($G^T$) but visit vertices in the order of decreasing $F(v)$.

**Minimum spanning trees**

- **Kruskal's algorithm**:
- **Prim's algorithm**:

**Single-source shortest paths**

**All-pairs shortest paths**

- $D^k(i,j)$: distance between vertices $i$ and $j$ which goes through vertices $\leq k$.
- Iterate the following for $1 \leq k \leq n$

$$D_{ij}^k = \begin{cases} w_{ij} & k = 0, \\ \min\{D_{ij}^{k-1}, D_{ik}^{k-1} + D_{kj}^{k-1}\} & k \leq 1 \end{cases}$$

**Network flows**

- A **flow network** $G = (V,E)$ is a directed graph where each edge $\langle u,v \rangle \in E$ has a nonnegative **capacity** $c(u,v) \geq 0$. There are two special vertices: a **source** $s \in V$ and a **sink** $t \in V$.
- A **flow** in $G$ is a function $f : V \times V \to \mathsf{R}$, which satisfies the following constraints (i.e. is a valid assignment of flow):
  1. **capacity constraints**: for all $u,v \in V$, $f(u,v) \leq c(u,v)$.
  2. **skew symmetry**: for all $u,v \in V$, $f(u,v) = -f(v,u)$.
  3. **flow conservation**: for all $u \in V - \{s,t\}$,

$$\sum_{u \in V} f(u,v) = 0.$$

- The **value of a flow** $f$ is defined as

$$|f| = \sum_{v \in V} f(s,v).$$

## References


[1] F. Harary. *Graph Theory*. Addison-Wesley, 1969.

[2] R. E. Tarjan and U. Vishkin. Finding biconnected components and computing tree functions in logarithmic parallel time. In *Proceedings of FOCS'84, 25th Annual IEEE Symposium on Foundations of Computer Science*, October 1984.