Characterizing Graphs (1)

Characterizing a class $\mathbf{G}$ by a condition P means proving the equivalence $G \in \mathbf{G}$ iff G satisfies P. That is, P is both a *necessary* and *sufficient* condition for membership in $\mathbf{G}$. Many questions asked in graph theory are related to characterization.

Example. What graphs are 2-colorable (have chromatic number at most 2)? We know that these are exactly the bipartite graphs. One might be satisfied with this observation or continue and try to arrive at another characterization (which perhaps is even more useful in designing algorithms for detecting such graphs).

Characterizing Graphs (2)

The **length** of a walk, path, cycle, or trail is the number of edges, and such an object is called *even (odd)* if it has *even (odd)* length. A vertex is called *even (odd)* if it has *even (odd)* degree, and a graph is said to be *even (odd)* if the degrees of all vertices are *even (odd)*.

Theorem 1.2.18. A graph is bipartite iff it has no odd cycle.

Proof: *Necessity.* ($\Rightarrow$) Every walk in a bipartite graph G alternates between the two partite sets, so there is an even number of edges in the walk between two occurrences of a vertex that occurs at least twice.

Characterizing Graphs (3)

Proof: (cont.) *Sufficiency.* ($\Leftarrow$) Let G be a graph with no odd cycle. For each nontrivial component we construct a bipartition as follows. For such a nontrivial component H, we fix a vertex $u \in V(H)$, and for every $v \in V(H)$ let $f(v)$ be the minimum length of a path from u to v.

Let $X = \{v \in V(H) : f(v) \text{ is even}\}$ and $Y = \{v \in V(H) : f(v) \text{ is odd}\}$. An edge within X or Y would lead to a closed odd walk, and by Lemma 1.2.15 (omitted) a closed odd walk must contain an odd cycle. Since no odd cycles exist, X and Y are independent sets. □

Characterizing Graphs (4)

Theorem 1.2.18 can be utilized in developing algorithms for testing whether a graph is bipartite or not (and is obviously useful for manual proofs as well). To prove that a graph is

not bipartite Find and present an odd cycle.

bipartite Find a bipartition and prove that the two sets are independent.

Eulerian Circuits (1)

The Königsberg Bridge Problem is that of finding a closed trail containing all the edges in the graph. Necessary condition are obviously that all vertex degrees be even and that the all edges belong to the same component. Euler stated (but did not give a proof) that these conditions are sufficient, which they indeed are.

circuit A closed trail.
Eulerian circuit A circuit containing all the edges. If such a circuit exists, the graph is said to be *Eulerian*.

Eulerian Circuits (2)

The proof of the following lemma shows an important technique of proof, *extremality*.

Lemma 1.2.25. If every vertex of a graph G has degree at least 2, then G contains a cycle.

Proof: Let P be a path in G this is not contained in a longer path, and let u be an endpoint of P. Since u has degree at least 2, it follows that u is incident with an edge that is not in P, which, by the initial assumption, must have both endpoints among vertices of P and therefore belong to a cycle. □

Eulerian Circuits (3)

Theorem 1.2.26. A graph G is Eulerian iff it has at most one nontrivial component and all its vertices are even.

Proof: *Necessity.* Obvious, observing that each passage through a vertex uses two incident edges.

Sufficiency. We use induction on the number of edges, m.
Basis step: $m = 0$. Obvious.
Induction step: $m \geq 1$. Each vertex of the nontrivial component of G has degree at least 2, and (by Lemma 1.2.25) has a cycle C. Let G' be the graph obtained from G by deleting $E(C)$.

Eulerian Circuits (4)

Proof: (cont.) Since C has 0 or 2 edges at each vertex, each component of G' is an even graph with fewer than m vertices and by the induction hypothesis has an Eulerian circuit. These may now be combined into an Eulerian circuit of G by traversing C and detouring along Eulerian circuits of the aforementioned components. See the figure on [Wes, p. 28]. □

This proof reveals another property of even graphs (an even graph has a cycle, deletion of a cycle leaves an even graph).

Theorem 1.2.27. Every even graph decomposes into cycles.

TONCAS

We have seen some examples of characterizations, where "The Obvious Necessary Conditions are Also Sufficient". The mnemonic **TONCAS** is used for such theorems throughout [Wes].

Maximum and Maximal

It is important to distinguish between the adjectives *maximum* and *maximal*.

maximum Of largest size.
maximal No larger one contains this one.

The path in the proof of Lemma 1.2.25 is then maximal. These terms are used for many other objects, such as cliques (and independent sets), connected subgraphs, etc. Note, however, that when we use these terms to describe numbers rather than containment, they can be used interchangingly: maximum vertex degree = maximal vertex degree.

Vertex Degree

We denote the degree of a vertex v in a graph G by $d_G(v)$, or simply $d(v)$. (Note that each loop counts twice.) The maximum degree is denoted by $\Delta(G)$ and the minimum degree by $\delta(G)$.

regular A graph for which $\Delta(G) = \delta(G)$ $(= k)$. To point out the common degree, it is called k-regular.
neighborhood The set of vertices adjacent to a given vertex; denoted by $N_G(v)$ or $N(v)$.
order of graph The number of vertices; denoted by $n(G)$.
size of graph The number of edges; denoted by $e(G)$.

The Handshaking Lemma

The following formula—an essential tool—is called the "First Theorem of Graph Theory", the "Handshaking Lemma", or the "Degree-Sum Formula". The proof technique is *counting two ways*.

Theorem 1.3.3. For a graph G,

$$\sum_{v \in V(G)} d(v) = 2e(G).$$

Proof: Summing the degrees counts each edge twice, since each edge has two endpoints. □

Corollary 1.3.5. Every graph has an even number of vertices of odd degree. No graph of odd order is regular with odd degree.

Hypercubes

The n-**dimensional cube** or n-**cube**, denoted by Q_n, is the simple graph whose vertices are the n-tuples with entries from $\{0, 1\}$ and whose edges are the pairs of tuples that differ in exactly one position (that is, their *Hamming distance* is 1).

Such a *hypercube* is a regular bipartite graph.

Theorem 1.3.9. If $k > 0$, then a k-regular bipartite graph has the same number of vertices in each partite set.

Proof: Let X and Y be two partite sets. Counting the edges with endpoints in X yields $e(G) = k|X|$, and with endpoints in Y yields $e(G) = k|Y|$. Therefore $k|X| = k|Y|$, so $|X| = |Y|$ when $k > 0$. □

Establishing Bijections (1)

A technique for counting a set is to *establish a bijection from it to a set of known size.*

Example. Counting the 6-cycles of the Petersen graph by establishing a one-to-one correspondence between these and the claws (there are obviously 10 claws).

Since G has girth 5, every 6-cycle F is an induced subgraph, and each vertex of F has one neighbor outside F. Since nonadjacent neighbors have exactly one common neighbor (Theorem 1.1.38), opposite vertices on F have a common neighbor outside F.

Establishing Bijections (2)

Example. (cont.) The three vertices that can be identified in this way are all distinct, since otherwise we would have a vertex with degree at least $2 + 2 = 4$. It follows that the graph induced by the vertices not in F is a claw.

In the other direction, starting from a claw H, we show that the vertices not in H induce a 6-cycle. There are six edges with one endpoint in H and one outside H. Since the girth of the Petersen graph is 5, the endpoints outside H must be distinct. Therefore, $G - V(H)$ is 2-regular and—knowing the girth of G—form a 6-cycle.

Vertex-Deleted Subgraphs

Subgraphs obtained by deleting a vertex are called **vertex-deleted subgraphs**. The **Reconstruction Conjecture** is an important open problem in graph theory.

Conjecture 1.3.12. If G is a simple graph with at least three vertices, the G is uniquely determined by the list of (isomorphism classes of) vertex-deleted subgraphs.

Extremal Problems (1)

An **extremal problem** asks for the minimum or maximum value of a function over a class of objects. Proving that β is the minimum—changing "$\geq$" to "$\leq$" yields the criteria for a maximum—of $f(G)$ for graphs in a class $\mathbf{G}$ requires showing two things:

1. $f(G) \geq \beta$ for all $G \in \mathbf{G}$.
2. $f(G) = \beta$ for some $G \in \mathbf{G}$.

Example. The minimum number of edges in a connected graph with n vertices is $n-1$ (see [Wes, Proposition 1.3.13]).

Extremal Problems (2)

Theorem 1.3.15. If G is a simple n-vertex graph with $\delta(G) \geq (n-1)/2$, then G is connected.

Proof: We prove that two vertices u and v have a common neighbor if they are not adjacent. Since G is simple, we have $|N(u)| \geq \delta(G) \geq (n-1)/2$, and analogously for v. When $u \not\leftrightarrow v$, $|N(u) \cup N(v)| \leq n-2$, since neither u nor v are in the union. Now

$$\begin{aligned} |N(u) \cap N(v)| &= |N(u)| + |N(v)| - |N(u) \cup N(v)| \\ &\geq (n-1)/2 + (n-1)/2 - (n-2) = 1, \end{aligned}$$

which completes the proof. □

Extremal Problems (3)

A result is **best possible** or **sharp** when there is some aspect of it that cannot be strengthened without the statement becoming false. The following example shows that the result in Theorem 1.3.15 is sharp (the *floor* of x, $\lfloor x \rfloor$, is the largest integer less than or equal to x, and the *ceiling* of x, $\lceil x \rceil$, is the smallest integer greater than or equal to x).

Example. Let G be the n-vertex graph with components isomorphic to $K_{\lfloor n/2 \rfloor}$ and $K_{\lceil n/2 \rceil}$. This graph has $\delta(G) = \lfloor n/2 \rfloor - 1$ and is disconnected.

We have solved an extremal problem: The maximum value of $\delta(G)$ among disconnected n-vertex simple graphs is $\lfloor n/2 \rfloor - 1$.

Optimization Problems (1)

In extremal problems, we find an optimum over a class of graphs. The problem of seeking extremes in a single graph—such as the maximum size of an independent set—is called an **optimization problem**. **Algorithms** are needed to solve instances of such problems. Proofs may also be *algorithmic*.

Theorem 1.3.19. Every loopless graph G has a bipartite subgraph with at least $e(G)/2$ edges.

Optimization Problems (2)

Proof: We partition G into two arbitrary sets X and Y, and consider the vertices with one endpoint in X and the other in Y (and call this graph H). If H contains fewer than half the edges of G incident to a vertex v, then we move v to the other partite set to increase the number of vertices of the bipartite graph.

This switching process must terminate at some point where $d_H(v) \geq d_G(v)/2$ for all v. Summing and applying the degree-sum formula yields $e(H) \geq e(G)/2$. □

Note: This algorithm only gives a *local maximum* for the number of edges in bipartite subgraphs (cf. maximal vs. maximum).

H-free Graphs (1)

In a class with n students, how many pairs of friends can there be if no two friends have a common third friend? In graph theory terms, how many edges can there be in a graph with no triangle.

A graph is said to be H-**free** if it has no induced subgraph isomorphic to H.

Theorem 1.3.23. The maximum number of edges in an n-vertex triangle-free simple graph is $\lfloor n^2/4 \rfloor$.

Proof: Let G be an n-vertex triangle-free simple graph. Let x be a vertex of maximum degree, with $k = d(x)$. There are no edges between neighbors of x.

H-free Graphs (2)

Proof: (cont.) Therefore, summing the degrees of x and its *non*neighbors counts at least one endpoint of every edge:

$$(n-k)k \geq \sum_{v \notin N(x)} d(v) \geq e(G).$$

We may now utilize the fact that $(n-k)k$ is the size of the complete bipartite graph $K_{n-k,k}$, and use a switching argument similar to that in the proof of Theorem 1.3.19. This gives the maximum of $(n-k)k$ for $k = \lfloor n/2 \rfloor$ or $k = \lceil n/2 \rceil$. The maximum is $\lfloor n^2/4 \rfloor$ and is achieved by $K_{\lfloor n/2 \rfloor, \lceil n/2 \rceil}$. □

Note: Calculus is *not* used to maximize $(n-k)k$.

Directed Graphs (1)

In a **directed graph** or **digraph**, each edge is an *ordered* pair of vertices. The first vertex of the ordered pair is the **tail** (from), and the second is the **head** (to).

Many of the concepts considered earlier carry over to the case directed graphs. We shall accentuate the main differences between directed and undirected graphs.

If there is an edge from u to v, written uv, then v is a **successor** of u, and u is a **predecessor** of v; we also write $u \rightarrow v$.

Directed Graphs (2)

The **underlying graph** of a digraph D is the graph obtained by treating the edges of D as unordered pairs.

A digraph is a *path* if its underlying graph is a path and one endpoint can be reached from the other by following the directions of the edges. By merging the endpoints of a path, we get a *cycle*.

In the adjacency matrix of a directed graph, the entry $a_{i,j}$ is the number of edges in G with tail v_i and head v_j. In the incidence matrix of a loopless digraph, the entry $m_{i,j}$ is 1, 0, or -1, if v_i is the tail, not an endpoint, or the head of e_j, respectively.

Example. See [Wes, Example 1.4.11].

Directed Graphs (3)

There are two variants of connectedness for digraphs.

weakly connected The underlying graph is connected.
strongly connected For each *ordered* pair of vertices, (u, v), there is a path from u to v.
strong component Maximal strongly connected subgraph.

Vertex Degrees

For a vertex v of a digraph, we have the following definitions:

outdegree The number of edges with tail v; denoted by $d^+(v)$.
indegree The number of edges with head v; denoted by $d^-(v)$.
successor set Defined as $N^+(v) := \{x \in V(G) : v \to x\}$.
predecessor set Defined as $N^-(v) := \{x \in V(G) : x \to v\}$.

The parameters $\delta(G)$ and $\Delta(G)$ now split into $\delta^-(G)$, $\delta^+(G)$, $\Delta^-(G)$, and $\Delta^+(G)$, with the obvious meanings.

Theorem 1.4.18. (Degree-Sum Formula) For a digraph G,

$$\sum_{v \in V(G)} d^+(v) = \sum_{v \in V(G)} d^-(v) = e(G).$$

Eulerian Digraphs

Theorem 1.4.24. A digraph is Eulerian iff $d^+(v) = d^-(v)$ for each vertex v and the underlying graph has at most one nontrivial component.

Proof: Omitted (left for the tutorials). □

Note: Every Eulerian digraph with no isolated vertices is strongly connected, although the characterization in Theorem 1.4.24 states that being weakly connected is sufficient.

Orientations and Tournaments (1)

There is a graph underlying each directed graph. One may also proceed in the opposite direction.

orientation A digraph D obtained from a graph G by choosing an orientation ($x \to y$ or $y \to x$) for each edge $xy \in E(G)$.
oriented graph An orientation of a simple graph.
tournament An orientation of a complete graph.

The number of oriented graphs with n vertices is $3^{\binom{n}{2}}$; the number of tournaments is $2^{\binom{n}{2}}$.

Orientations and Tournaments (2)

A tournament may be used to indicate the results of a sports tournament (!) where each participant plays each other participant exactly once. Then, for example, uv indicates that u won v and vu that v won u. The number of wins is then the outdegree of a vertex.

There need not be a clear winner, but one may prove, for example, that there is always a participant x such that, for every other participant z, either x beats z or x beats some team that beats z [Wes, Proposition 1.4.30]. (A *king* is a vertex from which every vertex is reachable by a path of length at most 2.)