Lecture 5: Basics of Graph Theory

Rajat Mittal*

IIT Kanpur

Combinatorial graphs provide a natural way to model connections between different objects. They are very useful in depicting communication networks, social networks and many other kind of networks. For the purpose of this course, graphs will mean combinatorial (or abstract) graphs as opposed to graphs of functions etc. which you might have learnt previously.

Graphs have become such an important tool that a complete field, *Graph Theory*, is devoted to learning about the properties of graphs. In the next few lectures we will learn about graphs, associated concepts and various properties of graphs.

1 Definitions

Depending upon the application, you will find many kind of combinatorial graphs studied in the literature. We will start with the most basic definition, the one we will deal with in this course, and mention the variations later.

A *graph*, $G = (V, E)$, is described by a set of vertices V and edges E which represent the connection between the vertices (set of unordered pairs of vertices). If (u, v) is an element in E then we say that vertex u and vertex v are *connected*. We also say that u is *adjacent* to v (and v is adjacent to u). The vertex set of a graph G is represented by $V(G)$ and the edge set by $E(G)$.

The graph is called *simple* (look at Fig. 1) if there is no loop, i.e., no vertex is connected to itself. There are at most $n(n-1)/2$ edges in a simple graph with n vertices.

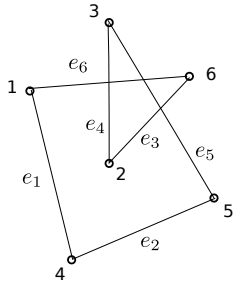

Fig. 1. A simple graph

As stated before, there are many variations possible in the definition of a graph.

If there is a possibility of multiple edges between vertices then it is called a *multigraph*. If the edges are assigned directions, i.e., (u, v) is ordered then it is called a *directed* graph as opposed to an *undirected* graph.

Sometimes edges are given weights (for their *importance*) and then the graph is called a *weighted graph*.

* Thanks to Nitin Saxena for his notes from the previous iteration of the course.

Exercise 1. Think of applications where weighted, directed and multigraphs might be useful.

Graphs are useful in modelling many natural situations. Let us take a look at few of them.

- Graphs are used extensively in modeling social networks. The members of the social group are represented by vertices and their relationship is modeled by the edges between them. If the graph depicts the friendship between people, like Facebook, it is an undirected graph. If it has a relation which has direction (say a is elder to b in a family), then we have a directed graph.
- Another example of a directed graph is how different processes are run on a computer. Suppose a process a can be run only if process b is completed. This is called the *precedence graph* and a directed edge (u, v) shows that u should run before v.
- Let us consider that we want to schedule exams for different courses where some courses have common students. How many time slots do we need? One way to model this is: denote every class by a vertex and join an edge between two classes if some student is common in two classes. Now, we need to assign labels to every vertex (time slots), so that any two vertices having the same label should not be adjacent. This motivates the concept of *coloring*, we will study coloring in detail later.
- Given a map of India, can we color each state so that no two neighboring states have the same color?

  Exercise 2. Show that this problems is similar to previous example, an instance of coloring problem.
- A road network is a weighted graph (with length of path being the weight of the edge). There are natural questions like: are two towns connected using the road network, what is the shortest way to reach from one town to another?

We will study undirected graphs in this course (assume simple graphs if not stated otherwise). Note that many of the concepts described below can be studied for the directed version too.

1.1 Subgraph, degree and some interesting graphs

A graph $H = (V, E')$ is called the *subgraph* of a graph $G = (V, E)$ if the edge set E' is a subset of the edges E in G (note that the vertex set remains the same). In other words, some edges of G are not present in H, but every edge of H is present in G.

The *degree* of a vertex v in a graph G is the number of edges from v in G. The degree of a vertex cannot be greater than $|V| - 1$. By double counting, we can easily prove the following theorem.

Theorem 1. *The sum of the degrees of all the vertices in a graph is equal to twice the number of edges.*

Count the number of pairs $\{(u, v) \in V^2 \mid (u, v) \in E(G)\}$. By looking at the degrees we get $\sum_{u \in V(G)} \deg(u)$. It is also, simply, $2 \cdot |E|$.

Exercise 3. Show that the number of vertices with odd degree is even.

Let us look at a few examples of simple graphs which arise often in practice.

- Complete graph: A graph on n vertices is called a complete graph, K_n, if every pair of vertices is connected. In other words, it has the maximum number of edges, $n(n-1)/2$.

  Exercise 4. What is the degree of a vertex in a complete graph?
- Path graph: A path on n vertices $v_1, v_2, \cdots, v_n$ is the graph with edges $(v_1, v_2), (v_2, v_3), \cdots, (v_{n-1}, v_n)$. It is the simplest graph where every vertex can be reached from any other vertex (called *connected*, we will see the formal definition soon).
- Cycle graph: A cycle on n vertices $v_1, v_2, \cdots, v_n$ is the graph with edges $(v_1, v_2), (v_2, v_3), \cdots, (v_n, v_1)$. Notice that there is one extra edge in a cycle as compared to a path.
- Bipartite graphs: This is a class of graphs. A graph is called bipartite if the vertex set $V(G)$ can be partitioned into two sets, say S and T, such that all edges have one vertex in S and one in T.

  Exercise 5. What is the maximum number of edges in a bipartite graph on n vertices?

  A complete bipartite graph, $K_{m,n}$, is a bipartite graph with two parts S (m vertices) and T (n vertices) such that every vertex in S is connected to every vertex in T.
- Regular graphs: Again, this is a class of graphs. A graph is called d-regular if every vertex has degree d. How many edges are there in d-regular graph on n vertices?

1.2 Representation of graphs as matrices

Adjacency matrix is a representation of a graph in matrix format. Given a graph G, its adjacency matrix A_G is a $|V| \times |V|$ matrix. Its rows and columns are indexed by the vertices of the graph. The (i, j)-th entry of the matrix is one if and only if vertex i is adjacent to vertex j, otherwise it is zero.

The adjacency matrix A_G for the graph in Fig. 1 is,

$$A_G = \begin{pmatrix} 0&0&0&1&0&1 \\ 0&0&1&0&0&1 \\ 0&1&0&0&1&0 \\ 1&0&0&0&1&0 \\ 0&0&1&1&0&0 \\ 1&1&0&0&0&0 \end{pmatrix}$$

Note 1. The adjacency matrix for an undirected graph is always symmetric.

Another matrix representation of a graph is called the *incidence matrix*. Here the rows are indexed by vertices and columns are indexed by edges. An entry (i, e) is one if and only if vertex i is part of edge e, otherwise it is zero. The incidence matrix for the graph in Fig. 1 is,

	e_1	e_2	e_3	e_4	e_5	e_6
v_1	1	0	0	0	0	1
v_2	0	0	1	1	0	0
v_3	0	0	0	0	1	1
v_4	1	1	0	0	0	0
v_5	0	1	0	0	1	0
v_6	0	0	1	0	0	1

Note 2. Here v_i represents the i-th vertex.

Exercise 6. Suppose A is the adjacency and M is the incidence matrix for a graph. Show that $MM^T = A$.

Many properties of graphs can be found by studying the spectrum (eigenvalues) of these matrices. Some of the results are given as extra reading at the end of these notes.

1.3 Graph isomorphism

One of the major problems in theoretical computer science is to figure out if two given graphs are isomorphic. Two graphs, G and H, are called *isomorphic* if there exists a bijection $\pi : V(G) \to V(H)$, s.t., $(u, v) \in E(G)$ if and only if $(\pi(u), \pi(v)) \in E(H)$.

Exercise 7. Given a graph G, its complement $\bar{G}$ is the graph with exactly those edges not present in G. Show that G and H are isomorphic iff $\bar{G}$ and $\bar{H}$ are isomorphic.

A sufficient condition for two graphs to be non-isomorphic is that there degrees are not equal (as a multiset).

Exercise 8. Construct all possible non-isomorphic graphs on four vertices with at most 4 edges.

Exercise 9. Construct two graphs which have same degree set (set of all degrees) but are not isomorphic.

Remember that adjacency matrix depends on the order of the vertices. The problem of graph isomorphism between two graphs G and T is same as, Is there a permutation of row (apply the same on columns) of A_G which makes turns it into A_T?

Let us define a relation (called isomorphism) between two graphs, graphs G and T are related iff G is isomorphic to T.

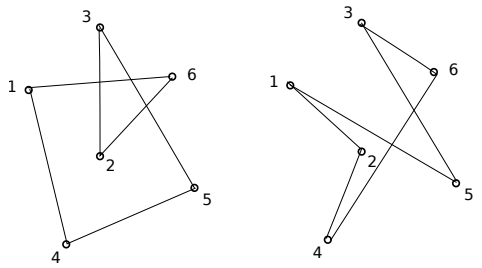

Fig. 2. Are these two graphs isomorphic?

Exercise 10. Show that isomorphism relation is an equivalence relation.

Hence, we get equivalence classes, inside which, each graph is isomorphic to other. Many a times, we don't care about the labels of the vertices. So, we are only interested in studying the equivalence classes.

The interesting graphs discussed above like complete graph, path graph and cycle graph are all equivalence classes. Though there are many equivalence classes in bipartite and regular graphs.

Exercise 11. Construct two bipartite graphs (same number of vertices) which are not isomorphic. Construct two d-regular graphs (same number of vertices) which are not isomorphic.

Look at two 2-regular graphs on 6 vertices, C_6 and two disjoint triangles, clearly they are not isomorphic. In C_6, all vertices are *connected* to each other. This is not true for two disjoint triangles. Let us study this *connection* property in detail.

2 Connectivity

An important question in graph theory is that of connectivity. Given a graph, can we reach from a source vertex s to target vertex t using the edges of the graph?

2.1 Paths and cycles

A *walk* of length k in a graph is a sequence of vertices $x_0, x_1, \cdots, x_k$, where any two consecutive vertices are connected by an edge in the graph. In a walk, it is allowed to take an edge or a vertex multiple times. The *length* of the walk is the number of edges in the walk.

If all the vertices in a walk are distinct, except possibly the first and the last vertex, then it is called a *path*. In some texts, paths are called simple paths and walks are called paths.

A path of length greater than two is called a *cycle* if the first and the last vertex are the same.

Example 1. Consider the simple graph $V(G) = [4]$, $E(G) = \{(1,2),(2,3),(3,4),(4,1),(4,2)\}$. The sequence $(1,2,3,4,1)$ is a walk, path and a cycle.

On the other hand, the sequence $(1,2,4,3,2,1)$ is a walk, but not a path or cycle.

The sequence $(1,2,3,4)$ is a walk and a path, but not a cycle.

Lemma 1. *If there is a walk between two vertices in a graph then there is a path between them.*

Proof. Consider the walk of *least* length, say $P = \{x_0, x_1, \cdots, x_k\}$. If all vertices are distinct then it is a path.

Suppose that there is a vertex v which occurs twice in the walk. If we delete the portion of the walk between the two occurrences of the vertex v, we still get a walk P' from x_0 to x_k.

But P was the smallest walk from x_0 to x_k, this is a contradiction. Thus, P is already a path. □

Exercise 12. Show that if there is a walk with the same first and last vertex and no two consecutive edges are the same, then there is a cycle in the graph.

Let the walk be $P = \{x_0, x_1, \cdots, x_k = x_0\}$. By the hypothesis, it can be seen that $k \geq 3$. As before, if there is a vertex $v \neq x_0$ which occurs twice in the walk, let us look at the portion of the walk from v to v. It has no two consecutive edges and is of smaller length. By induction, there is a cycle in the graph. If there is no repeated vertex except the first and last, P is a cycle.

Note 3. A proof similar to Lem. 1 will not work. The cycle need not be from the first to the last vertex. We can only guarantee a cycle in the entire graph.

A graph is called *connected* if there is a path between every pair of vertices of the graph.

Exercise 13. Define a relation between the vertices of the graph. Two vertices are related if there is a path between them. Show that this is an equivalence relation.

A graph can always be divided into disjoint parts which are connected with in themselves but not connected to each other. These are called the *connected components* of the graph. They are *uniquely* determined.

Exercise 14. Show that if there is path between a particular vertex v and every other vertex of the graph, then the graph is connected.

Consider a graph G and its adjacency matrix A. Since a path of length 1 is an edge, an entry (u,v) of A is 1 iff there is a path of length 1 between u and v.

What about the entries of A^2? Suppose (u,v) entry of A^2 is non-zero. Since all entries of A are positive, there should exist a t such that $A[u,t]$ and $A[t,v]$ is 1.

In other words, there is a walk of length 2. So, an entry of A^2 is non-zero iff there is a walk of length 2 between those vertices. More specifically, the entry (u,v) of A^2 gives the number of walks between u and v of length 2.

Exercise 15. Show that the (u,v) entry of A^k gives the total number of walks from u to v of length k.

2.2 Trees

A graph is called a *tree* if,

- It is connected, and
- There are no cycles in the graph.

Path graph is the simplest example of a tree.

Exercise 16. Give two non-isomorphic trees on $n \geq 3$ vertices.

Tree is a special graph where every pair of vertices have a unique path between them. Let us prove this formally.

Theorem 2. *A graph G is a tree iff there is a* unique *path between every pair of vertices in G.*

Proof. Suppose there is a unique path between every pair of vertices. That means, at least one path between every pair of vertices, implying the graph is connected. Suppose, such a graph has a cycle $x_0, x_1, \cdots, x_k = x_0$. Then there are two distinct paths between x_0 and x_1 (why?). This is a contradiction. Hence, a graph with unique path between every pair is a tree.

For the converse, assume we have a tree but there are two distinct paths between vertices u and v. Say $P = \{u = x_0, x_1, \cdots, x_k = v\}$ and $P' = \{u = y_0, y_1, \cdots, y_k = v\}$. Let i be the first index, s.t., $x_{i+1} \neq y_{i+1}$. Similarly j be the last index, s.t., $x_{j-1} \neq y_{j-1}$.

Exercise 17. Show that i and j exist and $j - i \geq 2$.

Now, consider the walk $x_i, x_{i+1}, \cdots, x_j = y_j, y_{j-1}, \cdots, y_{i+1}, y_i$. This is a walk with first and last vertex being the same and no two consecutive edges are same. By the exercise before, there is a cycle in this tree. This gives us a contradiction. □

Exercise 18. Show that if you remove any edge of a tree, you will get two disjoint connected components of the graph.

Let the edge be (u, v). Look at the connected components of u and v.

Exercise 19. A tree always has a vertex of degree one.

Assume that there is no degree one vertex. Start walking from a vertex s in the tree T. Walk in such a way that on reaching any vertex u you go out taking an edge different from the one used to reach u. Since there is no cycle in T, this walk will never end!

Then using induction on number of vertices, we can show that,

Theorem 3. *A tree on n vertices has $n - 1$ edges.*

Proof. Pick a degree one vertex v in the tree T. Consider the subgraph $T \setminus \{v\}$. You can show that the subgraph is also a tree. By induction, it has $n - 2$ edges. So, T has $n - 1$ edges. □

If a subgraph T of a graph G is a tree, on $V(G)$, then T is called a *spanning tree* of the graph G. If the graph G is connected, we can always construct a spanning tree. Define S_0 to be the initial set containing a particular vertex v. At every stage, construct S_{i+1} by including an edge which connects S_i to some vertex not in S_i.

Exercise 20. Why does this not create a cycle?

The process ends when S_i has all the vertices of the graph. We can proceed with each stage (find a vertex to add to S_i) because the graph is connected.

Spanning trees are important in many applications. They are the smallest structure which preserve the connectivity. If we assign weights/cost to every edge of the graph then we are mostly interested in *minimum weight* spanning tree. You will study algorithms to find minimum weight spanning tree in a graph in future courses.

3 Tours on a graph

Given a connected graph, there is a natural question, is it possible to visit all the vertices (tour the graph) without revisiting a vertex or an edge?

Finding such tours is sometimes easy and sometimes notoriously difficult. We will study two such tours, Eulerian and Hamiltonian in the next subsections.

3.1 Eulerian circuit

A *simple circuit* is a walk where first and last point are the same and no edges are repeated. An *Euler's circuit* is a simple circuit which uses all possible edges of the graph.

Note 4. The vertices can be repeated.

The definition of Eulerian circuits arose from the problem of "Königsberg bridges".

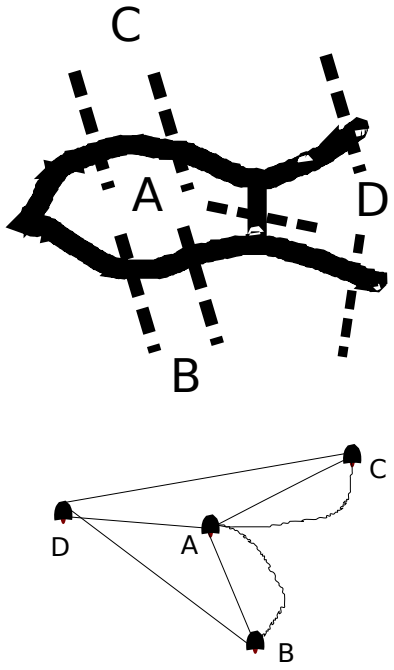

Fig. 3. Königsberg bridges and their graph representation. Dashed lines represent the bridges.

There are four regions connected with seven bridges. Can you go through all the bridges without revisiting any bridge? Converting it into a graph as shown in Fig. 3, the question is equivalent to finding an Eulerian circuit in the graph.

The following theorem gives the answer.

Theorem 4. *A connected graph (not necessarily simple) has an Eulerian circuit iff all vertices have even degree.*

Proof. If the graph has an Eulerian circuit then every vertex has even degree (show it as an exercise). Let us prove that if every vertex has an even degree then there is an Eulerian circuit.

We will construct the Eulerian circuit recursively. Start with an arbitrary vertex, say w, find an edge and move in that direction. Since the degree is even, whenever we arrive at a particular vertex, we have a different edge to leave that vertex too.

Since the number of edges are finite, we will arrive back at the starting vertex (call this cycle C). If all edges are covered then we have found the Eulerian circuit. If not, call the subgraph with edges of C removed as H. All the connected components of H will be connected to cycle C (since the graph is connected). Suppose the connected components are $H_1, H_2, \cdots, H_k$ and connection points are $w_1, w_2, \cdots, w_k$ when traversing the cycle C from w in a particular direction (say anticlockwise).

Note 5. There can be multiple connection points for a connected component, we will pick the first one while going in a particular direction on the cycle C.

Construct the Eulerian circuit in all connected components of H (Note: each vertex there still has even degree). We can use induction on number of edges, since H_i has less number of edges than the original graph. The base case, total 4 edges in graph, is easy.

Then the Eulerian circuit for G will be, go from w to w_1 using edges of C, take the Eulerian circuit of H_1, go from w_1 to w_2, take the Eulerian circuit for H_2, $\cdots$, reach w_k and take the Eulerian circuit for H_k, come back to w using remaining edges of C.

□

Exercise 21. Write the pseudocode for finding an Eulerian circuit in a graph G.

Exercise 22. Show that it is not possible to take an Eulerian walk on the Königsberg bridges.

3.2 Hamiltonian cycle

Similar to Euler circuits, a path $x_0, x_1 \cdots, x_k$ is called a *Hamiltonian path* if it goes through all the vertices of the graph. Remember that in a path we are not allowed to repeat the vertices.

Exercise 23. Show that a complete graph has a Hamiltonian path.

If the first and the last vertex of a Hamiltonian path is same then it is called a *Hamiltonian cycle*. A graph is called *Hamiltonian* if there is a Hamiltonian cycle in it. It is easy to see that if a graph is Hamiltonian, then adding edges to this graph will preserve the Hamiltonian property of the graph.

Exercise 24. Show that $K_{m,n}$ has a Hamiltonian cycle iff $m = n$.

Exercise 25. Show that there are two disjoint paths between any two vertices in a Hamiltonian graph. In other words, removing any vertex still keeps a Hamiltonian graph connected.

It might seem like an easy task to give a necessary and sufficient condition for the existence of Hamiltonian path/cycle like Eulerian circuit, but it turns out to be a really hard problem.

You will study later that finding a Hamiltonian path in a graph is an NP-complete problem (the list of problems which are assumed to be hard for computers to solve efficiently).

Still, there are many sufficient conditions known for existence and non-existence of Hamiltonian path. We will cover one of them here.

From the examples mentioned above, it seems that if there are many edges then the graph has good chances of being a Hamiltonian.

Theorem 5. *A simple graph on n vertices ($n > 3$) is Hamiltonian if every vertex has degree at least $n/2$.*

Proof. The proof is taken from the book *Introduction to Graph Theory* by Douglas West.

Suppose a graph H with the required condition on degree is not Hamiltonian (we will prove by contradiction). Keep adding edges till it remains non-Hamiltonian. Since complete graph is Hamiltonian, there will be a stage when adding any other edge will make the graph Hamiltonian. Call such a graph as G.

G satisfies the degree conditions (since we only added edges) and is not complete. We will show that G has a Hamiltonian cycle, that will give a contradiction.

Consider a non-edge (u, v) in G. Graph G with edge (u, v) is Hamiltonian by assumption and the Hamiltonian cycle contains (u, v) (G is non-Hamiltonian). Let the cycle be $u = v_1, v_2, \cdots, v_n = v$.

Define two sets, S and T as,

$$S = \{i : \ (u, v_{i+1}) \in E_G\} \text{ and } T = \{i : \ (v_i, v) \in E_G\}.$$

If there is an element in $S \cap T$, then Fig. 4 shows how to construct a Hamiltonian cycle in G. The new cycle is $v_1, v_2, \cdots, v_i, v_n, v_{n-1}, \cdots, v_{i+1}, v_1$.

The only thing needed to show is, the intersection $S \cap T$ is not empty. This follows because S, T have cardinality at least $n/2$ and n is not an element of $S \cup T$.

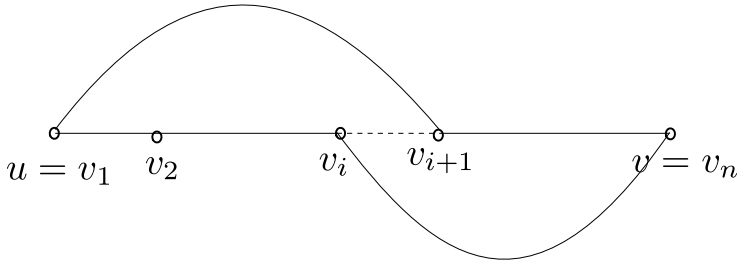

Fig. 4. Getting a Hamiltonian cycle in G

Exercise 26. Why are these conditions enough to show $S \cap T$ is not empty?

$$|T \cap S| + |T \cup S| = |T| + |S|.$$

□

Exercise 27. Show that the condition in Thm. 5 is not necessary?

Take the simple cycle as counterexample.

The degree bound for Hamiltonian property is tight. Take two K_n's and join them at a vertex (they will share a common vertex). In this case, total number of vertices is $2n-1$ and the degree is $n-1 = \lfloor (2n-1)/2 \rfloor$. The graph is not Hamiltonian because removing the shared vertex disconnects the graph.

4 Extra reading: spectral properties of adjacency matrix

The eigenvalues and eigenvectors (spectral properties) of the adjacency matrix provide us with a lot of information about the graphs. We will see a few examples below. First, a lemma which will help us in proving results about eigenvalues.

Lemma 2 (Matrix shift). *Suppose $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ are the eigenvalues of M. Then the eigenvalues of $\lambda I + M$ are $\{\lambda + \lambda_1, \lambda + \lambda_2, \cdots, \lambda + \lambda_n\}$, where I is the identity matrix. The converse is also true.*

Proof. Exercise in linear algebra. □

Exercise 28. The eigenvalues of a symmetric real matrix are real.

Hence, adjacency matrices have real eigenvalues.

Remember that a graph is called *regular* if every vertex has the same degree d. In this case, d is called the degree of the graph.

Theorem 6. *The maximum eigenvalue of the adjacency matrix of a regular graph G is d, the degree of the graph.*

Proof. By using Lemma 2, it is sufficient to prove that all eigenvalues of $dI - A_G$ are greater than zero.

For the sake of contradiction, assume that $M = dI - A_G$ has a negative eigenvalue μ with eigenvector u. Then $u^T M u = \mu u^T u < 0$. We will prove that for every vector v, $v^T M v \geq 0$, hence get a contradiction.

Let v_i be the i-th entry of v and M_{ij} be the (i,j)-th entry of M. Then,

$$\begin{aligned} v^T M v &= \sum_{i,j} M_{ij} v_i v_j \\ &= \sum_i d \cdot (v_i)^2 - \sum_{(i,j)\in E_G} v_i v_j \\ &= \sum_{(i,j)\in E_G} (v_i - v_j)^2 \geq 0 \quad [\text{Count } v_i^2]. \end{aligned} \tag{1}$$

Note 6. The term $v^T M v$ is known as the *quadratic form* of M.

To prove that d is an eigenvalue, notice that every row of the matrix A_G sums up to d.

Exercise 29. What is the eigenvector corresponding to the eigenvalue d ?

□

A graph is called *bipartite* if the vertex set can be divided into two parts A and B, s.t., there are no edges inside A and no edges inside B. A regular bipartite graph can be characterized by its eigenvalues.

Theorem 7. *The minimum eigenvalue of the adjacency matrix of a d-regular, connected graph is greater than or equal to $-d$.*

It is bipartite if and only if the minimum eigenvalue is $-d$.

Proof. By using Lemma 2, it is sufficient to prove that all eigenvalues of $M := dI + A_G$ are greater than or equal to zero. We will prove that: There will be an eigenvalue 0 iff the graph is bipartite.

Again we will notice the quadratic form $v^T(dI + A_G)v$,

$$\begin{aligned} v^T M v &= \sum_{i,j} M_{ij} v_i v_j \\ &= \sum_i d(v_i)^2 + \sum_{(i,j)\in E_G} v_i v_j \\ &= \sum_{(i,j)\in E_G} (v_i + v_j)^2 \geq 0 \quad [\text{Count } v_i^2]. \end{aligned} \tag{2}$$

Suppose equality holds in the above equation. Notice that v_i corresponds to the i-th vertex. If for some i, $v_i = 0$ then, by repeatedly using the identities $v_i + v_j = 0$ for $(i,j) \in E_G$, we deduce that v is the zero vector. Thus, for all i, $v_i \neq 0$.

Let us say S_1 is the set of vertices for which v_i is positive and S_2 is the set of vertices for which v_i is negative. Then there are no edges inside S_1 and inside S_2.

This implies that the graph is bipartite, with $V(G)$ partitioned into S_1 and S_2.

Conversely, if the graph is bipartite, then assign 1 to one part and -1 to the other part of the vertices. This will show that the least eigenvalue of $dI + A_G$ is zero (why?). Thus, the least eigenvalue of A_G is $-d$.

□

Exercise 30. Show that the bipartite graph cannot have an odd cycle.

A *complete graph* is a simple graph where every possible edge is present. A complete graph on n vertices is called K_n.

Exercise 31. How many edges are there in a complete graph?

The adjacency matrix of the complete graph K_n is $J - I$, where J is all 1's matrix and I is the identity matrix, both of dimension $n \times n$.

Exercise 32. What are the eigenvalues of the adjacency matrix of the complete graph.

$(n-1)$ once and the rest are -1's.

References

1. K. H. Rosen. Discrete Mathematics and Its Applications. *McGraw-Hill*, 1999.
2. N. L. Biggs. Discrete Mathematics. *Oxford University Press*, 2003.