AN EXPOSITION OF CERTAIN CONCEPTS IN GRAPH THEORY

JONATHAN HANNA, WILLIAM KOYFMAN, SAMUEL MAYLE

Abstract. Graph theory is an important field of modern-day mathematics originating in the 18th century. Although simple aspects of the field permeate the understanding of many individuals, there remain many obscured results that prove not only intriguing but also applicable to tangible systems, especially networks. In this short exposition, we provide a curated assortment of concepts in graph theory and the associated proofs. We build upon definitions defined here, as well as certain algebraic properties and properties regarding linear algebra. The paper also contains visual representations of graphs to facilitate the understanding of particular definitions and proofs.

Contents

1. Introduction

1.1. **Brief Overview.** Graph theory, put simply, is the study of graphs. Graph theory began with what seemed to be a simple problem. In 1736, Euler studied the famous Königsberg bridge problem, which asked whether someone could walk through the city of Königsberg and cross each of the 7 bridges exactly once without getting trapped or skipping a bridge. Again, though it seems rather elementary, it is the mathematical foundation for how we analyze many complex systems.

Graph theory has grown into a major part of modern mathematics and, more recently, computer science. Graphs can model many different things. A graph only contains vertices and edges, but those vertices and edges can represent almost anything that is connected. Vertices might be cities, people, websites, or computers. Edges might be bridges, friendships, links, and a variety of other variables.

Trees are one of the most important types of graphs because they represent connected systems with no loops. This makes them useful whenever the goal is to connect vertices as efficiently as possible. As an example, imagine someone is trying to pick all of their friends up from their respective houses. What is the best way of navigating? A minimum spanning tree then gives the most efficient way to get to each house (defined in subsection 3.3). This is the basic idea behind the "connector problem" (finding the minimum spanning tree, which is covered in subsection 3.3 onward).

The other topic of our paper, spectral graph theory, shifts the focus from the apparent structure of a graph to the algebraic properties of matrices associated with it. Important matrices in spectral graph theory include the adjacency matrix and the Laplacian.

There is far more to graph theory than is covered here. For instance, Ramsey Theory is a field that is deeply intertwined with graphs and continues to produce results as of this year.

Date: May 2026.

2. Basic Concepts

2.1. **What are graphs?** Graphs are, at their heart, mathematical models of parts of the real world. For example, the map of a city can be shown as a graph where intersections are vertices and streets are edges. They can represent chemicals, where edges are bonds and vertices are atoms. They can represent nearly anything.

More formally, a graph is an ordered triple, $G = (V(G), E(G), I_G)$, where $V(G)$ is a nonempty set, $E(G)$ is a set disjoint from $V(G)$, and I_G is an incidence relation that associates each element of $E(G)$ with two unordered elements of $V(G)$. I_G can be thought of as a certain function that states the vertices that are incident to an edge.

2.2. **Terminology.** For these definitions, let us fix a graph G. $V(G)$ is the set of all vertices, $E(G)$ is the set of all edges, and I_G is the incidence function that relates each edge to the vertices it is between.

Definition 2.1. G is considered a **connected** graph if there exists a **path** (sequence of incident vertices and edges) between any two vertices in $V(G)$.

Definition 2.2. The **degree** of a vertex is the number of edges that connects to it.

The **degree sequence** of a graph is the sequence of degrees $(d_1, d_2, \ldots, d_n)$ of the vertices of a graph where n is the **order** (cardinality of $V(G)$).

We let $\Delta(G)$ and $\delta(G)$ denote the largest and smallest degree of a graph G, respectively.

Definition 2.3. An **edge cut** of a graph is the set of edges of the form $[S, \overline{S}]$ in a nontrivial connected graph G where S and $\overline{S}$ are nonempty vertex subsets of G, and $\overline{S}$ is the complement of S in the vertex set of G.

The **edge connectivity**, $\lambda(G)$, of a graph is the value of the minimum number of edges in an edge cut for any valid subsets S and $\overline{S}$.

Definition 2.4. A **vertex cut** of a graph G is a subset of vertices such that the removal of the vertices from the vertex set, as well as any edges that are incident to a removed vertex, results in a disconnected graph.

The **vertex connectivity** $\kappa(G)$ denotes the value such that a set of $\kappa(G)$ vertices exists that is also a vertex in G, and no vertex cut comprised of less than $\kappa(G)$ vertices exists.

Definition 2.5. The **components** are the maximally connected subgraphs. G is connected if and only if it has only one component.

The number of components is denoted as ω or $\omega(G)$.

Definition 2.6. A **cycle** is a path that starts and ends at the same vertex. It otherwise has no repeat vertices. (An important exception is the loop, which occurs when an edge is incident to the same vertex twice. This is not a cycle.)

Definition 2.7. A **subgraph** H is a graph such that $V(H) \subseteq V(G)$, $E(H) \subseteq E(G)$, and I_H is restricted by I_G.

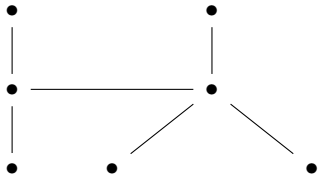
A subgraph is a spanning subgraph if $V(H) = V(G)$.

Definition 2.8. The **complete graph**, K_n, is the graph of n vertices with every possible edge.

Definition 2.9. A **directed graph** D, or digraph (or quiver), is like any other graph, except instead of having edges, it has arcs (arrows). As such, it has an arc set $A(D)$ instead of an edge set, and its incidence relation is ordered rather than unordered.

3. Trees

A **tree** is a certain type of graph that is connected but has no cycles. They are of particular focus in the world of practical graph theory. Their name arrives intuitively from the shape of their graphs. An example of a tree is below. Proofs about trees are often intuitive but require certain preliminary steps to prove rigorously.



A Tree

3.1. **Terminology.** Another set of graphs exists such that the graph has no cycles but is not necessarily connected. These are known as **acyclic graphs**, and each connected component of these graphs is a tree. Often, properties of trees can be extended to each connected component of the graph or the graph as a whole.

Definition 3.1. A spanning tree of a graph is a spanning subgraph that is a tree. $\tau(G)$ is the number of spanning trees in G.

Definition 3.2. A **subtree** is a connected subgraph of a tree. They are always trees. (Disconnected subgraphs of trees have components that are all trees.)

3.2. **Properties of trees.** There are some basic properties of trees that are not only interesting to prove but also act as a basis for further proofs. Those proofs can be found here.

Theorem 3.3. *A simple graph is a tree if and only if any pair of distinct vertices is connected by a single unique path. [BR12, Chapter 4]*

Proof. If T is a tree and there exists more than one path between two vertices, the union of the two paths has a cycle. On the other hand, if a path exists between every pair of vertices, the graph is connected. As all vertices in a cycle are connected by multiple paths, a graph with a single unique path between every pair of vertices cannot have a cycle. Thus, this graph must be a tree. □

Theorem 3.4. *Every connected graph has a spanning tree. [BR12, Chapter 4]*

Proof. If G is the connected graph in question, we can consider L, the set of spanning subgraphs of the graph G. G is a spanning subgraph of G, so L is not an empty set. It will then always be possible to consider T, the spanning subgraph with the fewest edges. T cannot have cycles, because removing any edge from a cycle in T would result in another spanning subgraph with fewer edges[1]. Thus, we know that T is a spanning tree of G. □

Theorem 3.5. *If a tree has n vertices, it must have $n-1$ edges. If a graph is connected and has $n-1$ edges and n vertices, it must be a tree.*

Proof. We can observe that the number of edges of a tree is 1 less than the number of vertices when $n = 1$ or $n = 2$ by examining all possible cases.

The only trees for 1 and 2 vertices, respectively.

[1]The removal of an edge from the cycle does not disconnect any vertex of the cycle, and thus does not disconnect the spanning subgraph.

We may now apply induction to prove the same condition for greater values of n. Consider the tree T with n vertices. We assume that, for $n \geq 3$, the property holds on trees with $n-1$ vertices or fewer. Without loss of generality, an edge of the tree may be chosen and removed, creating two components, T_1 and T_2, that are trees. Since each component has at most $n-1$ vertices, the property holds on both of them. As such,

$$m(T) = 1 + m(T_1) + m(T_2) = 1 + n(T_1) - 1 + n(T_2) - 1 = n(T) - 1$$

and the property holds for T. This completes the proof by induction on n.

If a connected graph has $n-1$ edges and n vertices, it must have a spanning tree (by Theorem 3.4), and this spanning tree also has n vertices and $n-1$ edges. Thus, the spanning tree must be identical to the connected graph, making the graph a tree. □

3.3. **Algorithms and Applications Regarding Trees.**

Definition 3.6. A **weighted graph** is a graph in which each edge has an assigned value, its **weight**. If an edge, denoted by e, belongs to a graph, we denote its weight by $w(e)$. For example, this weight can represent the amount of traffic on a road in a navigation app.
The **total weight** W of a weighted tree T is:

$$w(T) = \sum_{e \in E(T)} w(e) \tag{3.7}$$

3.3.1. *Minimum Spanning Tree (MST).*

Definition 3.8. Let G be a connected weighted graph. A **minimum spanning tree (MST)**, or as Balakrishnan and Ranganathan define, the **minimum-weight spanning tree**, is a spanning tree (as defined previously) T of G such that for every spanning tree S of G, $w(T) \leq w(S)$ [BR12, Ch. 4].
A minimum spanning tree must satisfy two conditions: First, it is a spanning tree, and therefore every vertex of G is connected by T and T has no cycles. In addition, $w(T)$ is minimized.

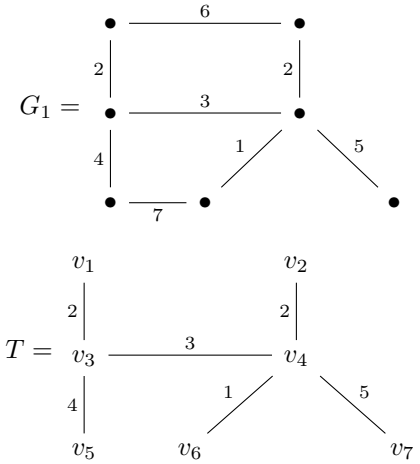


FIGURE 1. The weighted graph G_1 and a spanning tree T of G_1.

In the example above, T is a spanning tree of G_1. It contains all seven vertices of G_1, is connected, and has no cycles, fulfilling the criteria of a spanning tree. Since T has 7 vertices and 6 edges, this agrees with the fact that every tree with n vertices has $(n-1)$ edges, the tree edge theorem.

The total weight (via summing each edge's weight) of T is

$$w(T) = 2 + 3 + 4 + 2 + 1 + 5 = 17 \tag{3.9}$$

The two edges that belong to G_1 and not to T, namely, connecting v_1 to v_2 with weight 6 and the edge connecting v_5 to v_6 with weight 7, are removed. Adding either of these edges would create a cycle and would violate the definition of a spanning tree; in addition, they are not necessary for connectivity.

Example 3.1. *We can also see how a graph may have more than one spanning tree.*

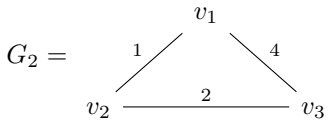


FIGURE 2. The weighted graph G_2.

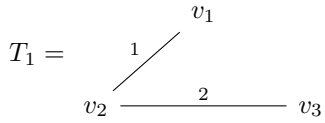


FIGURE 3. A spanning tree of G_2, with total weight $w(T_1) = 1 + 2 = 3$.

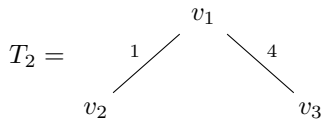


FIGURE 4. A spanning tree of G_2, with total weight $w(T_2) = 1 + 4 = 5$.

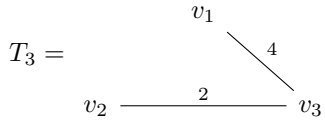


FIGURE 5. A spanning tree of G_2, with total weight $w(T_3) = 2 + 4 = 6$.

Thus, there are three different spanning trees, with the minimum spanning tree being T_1 with a weight of 3.

Theorem 3.10. *A graph can have multiple spanning trees, but there will always be* ***a unique minimum spanning tree*** *for a graph with distinct weights.*

Proof. Suppose, for contradiction, that G has two different minimum spanning trees, T and S. Since T and S are different, there is some edge that is in one tree but not the other. Let e be the edge with minimum weight among those edges. Without loss of generality, assume

$$e \in T \quad \text{and} \quad e \notin S.$$

Now add e to S. Since S is already a tree, adding one edge creates exactly one cycle.
This cycle must contain some edge f that is in S but not in T. If it did not, then every edge in the cycle would also be in T, which would mean T contains a cycle. But T is a tree, so this is impossible. Thus,

$$f \in S \quad \text{and} \quad f \notin T.$$

So both e and f are edges where T and S differ. Since e was chosen to be the edge with minimum weight, and all edge weights are distinct, we have

$$w(e) < w(f).$$

Now, replace f with e in S. Take the tree S, add the edge e, and remove the edge f. That is, consider

$$S' = S \cup \{e\} - \{f\}.$$

This is still a spanning tree, but its total weight is

$$w(S') = w(S) + w(e) - w(f).$$

Since $w(e) < w(f)$, we get

$$w(S') < w(S).$$

This contradicts the fact that S was a minimum spanning tree. Therefore, G cannot have two different minimum spanning trees. So the minimum spanning tree is unique.

□

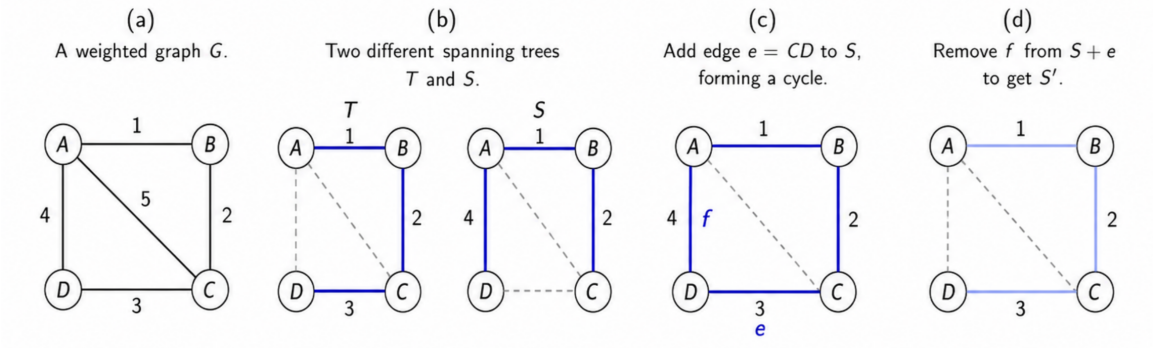


FIGURE 6. An illustration of the uniqueness proof for minimum spanning trees.

3.3.2. *Kruskal's Algorithm.*

Definition 3.11. Kruskal's algorithm is defined as follows. Let G be a simple connected weighted graph with an edge set of $E(G) = \{e_1, \ldots, e_m\}$. Now, pick the edge with the minimum weight of the whole graph. This process is repeated on the set of **eligible edges**, those being edges that fulfill two conditions. The first is that the edge has not already been selected, and the second is that its addition to the set of previously selected edges would not create a cycle [CLRS09, Ch. 23].

Equivalently, Kruskal's algorithm sorts the edges by increasing weight and then examines them in order.

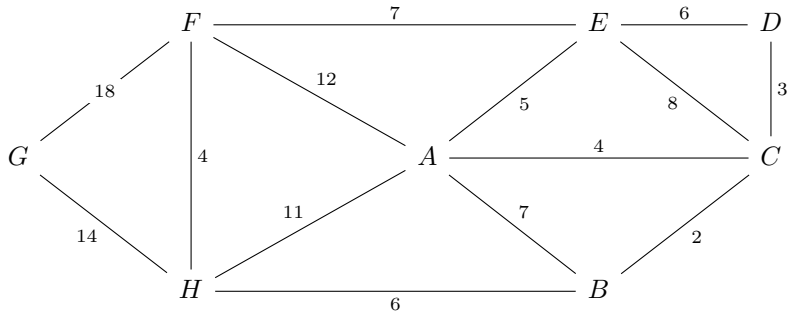


FIGURE 7. The weighted graph G_3 used to illustrate Kruskal's algorithm and Prim's algorithm.

For the graph G_3, Kruskal's algorithm considers edges in increasing order of weight.

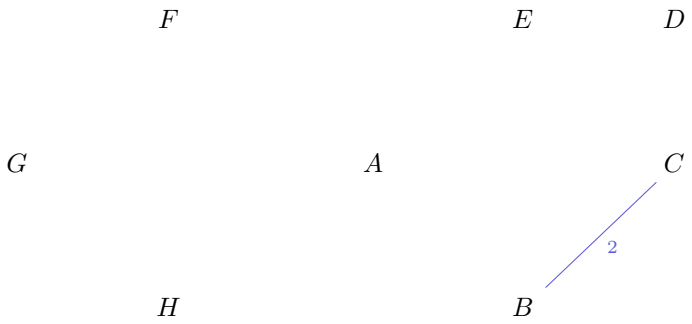


FIGURE 8. Kruskal's algorithm first chooses BC, which has weight 2.

We then proceed with the edge CD with weight 3. Then it selects AC and FH, both with weight 4.

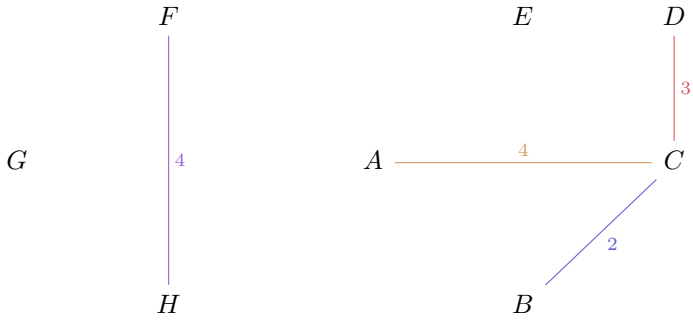


FIGURE 9. Kruskal's algorithm then chooses CD, AC, and FH.

Following this, we choose the edge AE with a weight of 5. The edges ED and HB are the next minimal edges with weight 6, but ED would create the cycle $E - A - C - D - E$ and is therefore not eligible to be picked. The edge HB, on the other hand, does not create a cycle, and it is added.

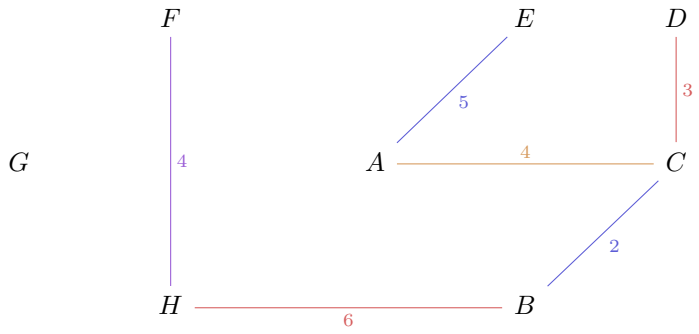


FIGURE 10. Kruskal's algorithm then adds AE and HB.

The last vertex G is not connected yet. The minimal edge that connects it and thus all vertices is HG with a weight of 14, so it is added. The final tree has 8 vertices and 7 edges, so it is a spanning tree.

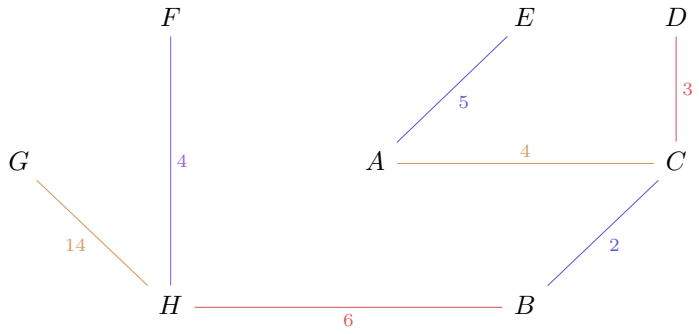


FIGURE 11. The spanning tree produced by Kruskal's algorithm.

Theorem 3.12. *The tree produced by Kruskal's algorithm is the minimum spanning tree.*

Proof. Kruskal's algorithm is an algorithm because it gives a finite step-by-step process: list the edges by weight, examine them in order, and add an edge exactly when it does not create a cycle. The result is a tree because the chosen edges are always acyclic, and since G is connected, the process continues until all vertices are connected, producing a connected acyclic spanning subgraph. Let G be a simple connected graph containing n vertices with an edge set $E(G)$. Then, let T' be the tree produced by Kruskal's algorithm and $E(T')$ be its associated edge set.

Suppose the edges of T' are chosen in the order

$$e_1, e_2, \ldots, e_{n-1}.$$

Thus

$$E(T') = \{e_1, e_2, \ldots, e_{n-1}\}.$$

Assume for the sake of contradiction that T' is not a minimum spanning tree.

Among the various minimum spanning trees of G, we select one, call it T_0, with edge set $E(T_0)$ that shares the maximum number of edges with T' without being the same spanning tree.

$$e_1, e_2, \ldots, e_{k-1} \in E(T_0)$$

while holding,

$$e_k \notin E(T_0).$$

Now, add e_k to T_0. Since T_0 is a tree, adding one edge will create exactly one cycle. Call this cycle C_1. This cycle cannot consist only of edges from T'. If it did, then T' would contain a cycle, which is impossible because T' is a tree. Therefore, C_1 contains some edge f such that

$$f \in E(T_0)$$

and

$$f \notin E(T').$$

Now, a new tree T_1 is defined as

$$T_1 = T_0 + \{e_k\} - \{f\}$$

Since f lies on the cycle created by adding e_k, removing f breaks the cycle and keeps the graph connected. Therefore, T_1 is also a spanning tree.

Next, compare the weights of e_k and f. When Kruskal's algorithm chose e_k, the edges

$$e_1, e_2, \ldots, e_{k-1}$$

had already been chosen. Also, adding f to these edges would not create a cycle, because

$$e_1, e_2, \ldots, e_{k-1}, f \in E(T_0).$$

and T_0 is a tree. So f must have been an available edge at that step.

Since Kruskal's algorithm chose e_k as the available edge of smallest weight,

$$w(e_k) \leq w(f).$$

Therefore

$$w(T_1) = w(T_0) + w(e_k) - w(f).$$

Using $w(e_k) \leq w(f)$, we get

$$w(T_1) \leq w(T_0).$$

But T_0 is a minimum spanning tree, so no spanning tree can have a smaller weight than the weight of T_0. Hence

$$w(T_1) = w(T_0),$$

and T_1 is also a minimum spanning tree.

However, T_1 contains

$$e_1, e_2, \ldots, e_k,$$

so T_1 shared common edges with T' for a longer initial segment than T_0 did. This contradicts the way we chose T_0, because we chose the maximum initial segment.

Therefore, our initial assumption was false, and T' must be a minimum spanning tree. □

3.3.3. *Prim's Algorithm.*

Definition 3.13. In **Prim's algorithm**, let G be a connected weighted graph. Choose a **starting vertex**. Let S be the set of vertices already included in the growing tree. Among all edges with one endpoint in S and one endpoint in $V(G) \setminus S$, choose an edge of minimum weight. Add this edge and its new endpoint to the tree. Repeat until all vertices have been included [CLRS09, Ch. 23].

Prim's algorithm differs from Kruskal's algorithm in how it grows the tree. Kruskal's algorithm may have several components at intermediate stages, while Prim's algorithm always keeps one connected tree. Starting at a chosen vertex, it repeatedly adds the edge of minimum weight that connects the current tree to a new vertex. We apply Prim's algorithm to G_3, beginning at the vertex A.

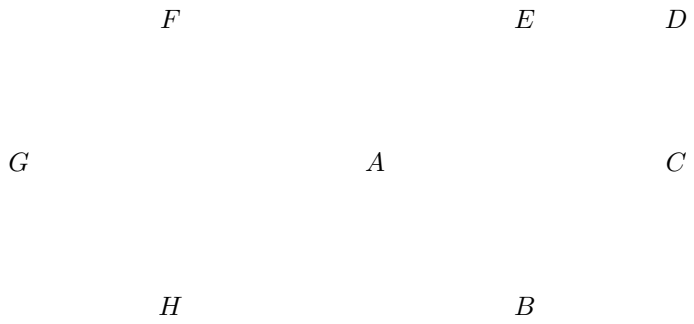


FIGURE 12. Prim step 0: start with the single vertex A.

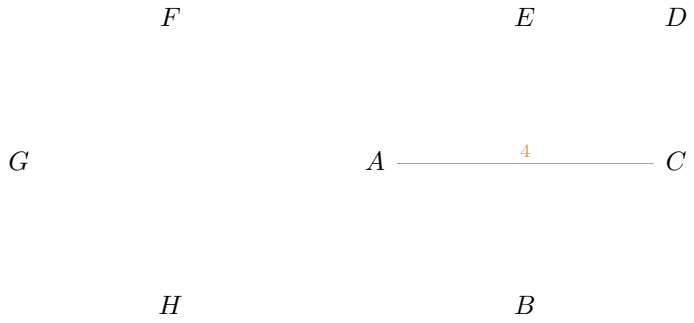


FIGURE 13. Prim step 1: add AC, the edge with the least weight connected to the vertex we picked, A.

Now, we look for the minimum-weight edge leaving the current vertex set, which contains A, C. The least edge leaving it is CB with weight 2, followed by CD with weight 3.

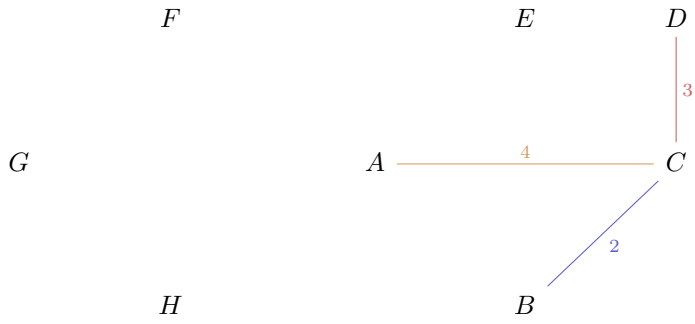


FIGURE 14. Prim steps 2 and 3: add CB with weight 2, followed by CD with weight 3.

The next least edge connecting the current tree to a new vertex is AE, of weight 5.

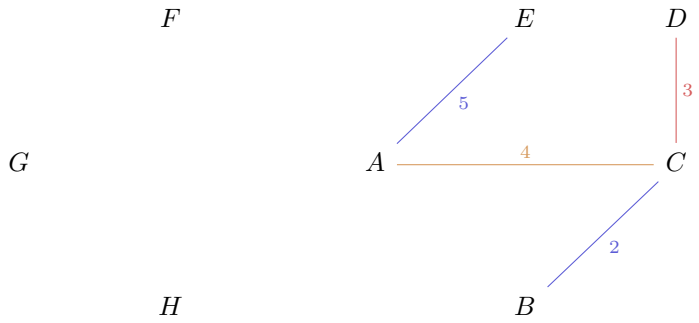


FIGURE 15. Prim step 4: add AE, of weight 5.

The next least edge leaving the current tree is BH, with weight 6. Then HF, of weight 4, connects F to the tree.

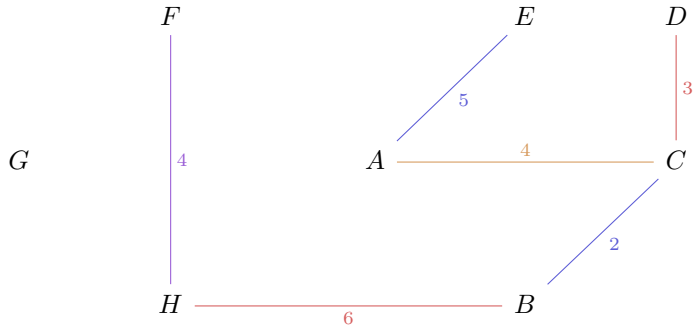


FIGURE 16. Prim steps 5 and 6: add BH, with weight 6, and then HF, with weight 4.

Finally, the only vertex not yet included is G. The least edge connecting G to the tree is HG, with weight 14.

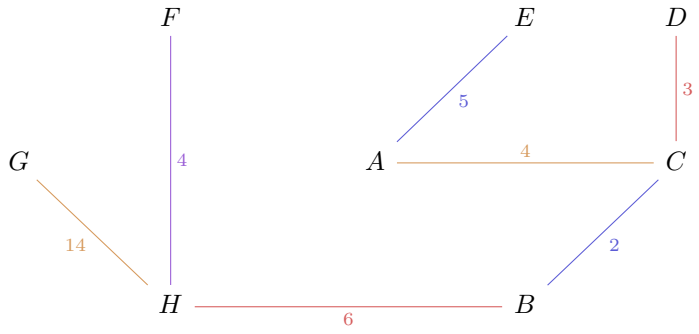


FIGURE 17. Prim step 7: add HG, with weight 14. The spanning tree produced by Prim's algorithm has the same total weight as the tree produced by Kruskal's algorithm.

Theorem 3.14. *The spanning tree produced by Prim's algorithm is a minimum spanning tree.*

Proof. At any stage of Prim's algorithm, let S be the set of vertices already included in the current tree. Prim chooses the minimum-weight edge crossing the cut from S to $V \setminus S$, adding one new vertex, so this cannot create a cycle. By the cut property, this edge is safe: it is contained in some minimum spanning tree that extends the edges chosen so far. Therefore, after each step, the current tree can still be extended to a minimum spanning tree. When the algorithm stops, all vertices have been included, so the chosen edges form a spanning tree. Since those edges can be extended to a minimum spanning tree but already form a full spanning tree, the tree produced by Prim's algorithm must itself be a minimum spanning tree. □

4. Spectral Graph Theory

4.1. **Important Concepts in Linear Algebra.** Linear algebra, in this writing, consists mainly of determinants and eigenvalues, both aspects of matrices.

Definition 4.1. An n by m **matrix** is a set of numbers, with n rows and m columns. For example, here is a 2 by 3 matrix:

$$A = \begin{bmatrix} 1 & 2 & 4 \\ 2 & 8 & 9 \end{bmatrix}$$

When a_{ij} is mentioned, that refers to the term in the ith row and jth column. The **transpose** of a matrix, A^T is flipping it, so that that a_{ij} in the original matrix A becomes a_{ji} in A^T. For example, for the matrix above, the transpose is:

$$\begin{bmatrix} 1 & 2 \\ 2 & 8 \\ 4 & 9 \end{bmatrix}$$

The **dot product** of two vectors is determined by multiplying corresponding terms between vectors and then taking the sum of the products. For example,

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \cdot \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = 1 \cdot 4 + 2 \cdot 5 + 3 \cdot 6 = 32$$

When two matrices AB are multiplied, every entry a_{ij} is equal to the jth column of B dotted with the ith row of A. For example:

$$AA^T = \begin{bmatrix} 1 & 2 & 4 \\ 2 & 8 & 9 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 8 \\ 4 & 9 \end{bmatrix} = \begin{bmatrix} 1 \cdot 1 + 2 \cdot 2 + 4 \cdot 4 & 2 \cdot 1 + 8 \cdot 2 + 9 \cdot 4 \\ 2 \cdot 1 + 8 \cdot 2 + 9 \cdot 4 & 2 \cdot 2 + 8 \cdot 8 + 9 \cdot 9 \end{bmatrix} = \begin{bmatrix} 21 & 54 \\ 54 & 149 \end{bmatrix}$$

The **Identity matrix**, I_n, is the square matrix of size n where the entries along the diagonal are 1 and everything else is 0. When it is multiplied by x, shown by xI_n, every entry along the diagonal is x

Definition 4.2. The **determinant** of a square matrix is, conceptually, the volume of the parallelepiped (multi-dimensional parallelogram) with the vectors as its sides (Fig 1). For the small 2 by 2 matrix, $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$, the determinant is $ad - bc$. For larger matrices, though, the determinant can be found in several ways, the simplest for being Laplace expansion. Laplace expansion involves writing the determinant of a matrix as the sum of several determinants of smaller matrices. Specifically, one takes an entry (typically from the first row) and then multiplies it by the determinant of the matrix left after removing that entry's row and column, then alternately subtracts and adds the products of the same process for all the entries in the row. For example, taking the determinant of the matrix below, shown by

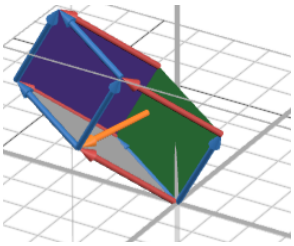


FIGURE 18. The parallelogram (green) from vectors (1,3,5) and (-1,-4,2)(blue). The third vector (4,-3,4)(red) makes it a parallelepiped (with gray and purple) and the volume of that parallelepiped (given the base) is determined solely by the distance(orange) of the red vector from the parallelogram.

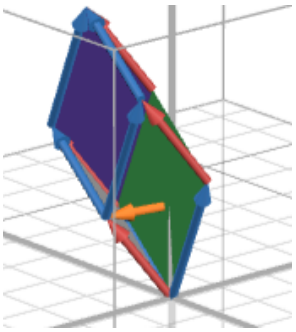


FIGURE 19. A second parallelepiped. The blue vectors are the same, as are the meanings of the various pieces. Here though, the red third vector is (2,-1,3)

$$\det \begin{bmatrix} 1 & 1 & 3 \\ 5 & 6 & 5 \\ 2 & 3 & 4 \end{bmatrix}, \text{ or } \begin{vmatrix} 1 & 1 & 3 \\ 5 & 6 & 5 \\ 2 & 3 & 4 \end{vmatrix}$$

gives $1\cdot\begin{vmatrix} 6 & 5 \\ 3 & 4 \end{vmatrix} - 1\cdot\begin{vmatrix} 5 & 5 \\ 2 & 4 \end{vmatrix} + 3\cdot\begin{vmatrix} 5 & 6 \\ 2 & 3 \end{vmatrix}$. This leads to $1(6\cdot 4 - 5\cdot 3) - 1(5\cdot 4 - 5\cdot 2) + 3(5\cdot 3 - 6\cdot 2) = 8$

4.1.1. *Cauchy–Binet.*

Theorem 4.3. *Additivity of Determinants:*

$$\det \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2m} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} & \dots & a_{nn} \end{bmatrix} + \det \begin{bmatrix} a_{11} & a_{12} & \dots & b_{1m} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & b_{2m} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & b_{nm} & \dots & a_{nn} \end{bmatrix} =$$

$$= \det \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} + b_{1m} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2m} + b_{2m} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} + b_{nm} & \dots & a_{nn} \end{bmatrix}$$

Proof. This will remain true as long as only one row is changed. This, again, is easiest thought of conceptually. When the a vector is absent, the parallelepiped is instead a plane in $n-1$ dimensions (see Fig 1). Then, when that vector is added, the volume of the now parallelepiped (for a constant base) depends solely on the distance of that vector from the base. (like in A= base· height).

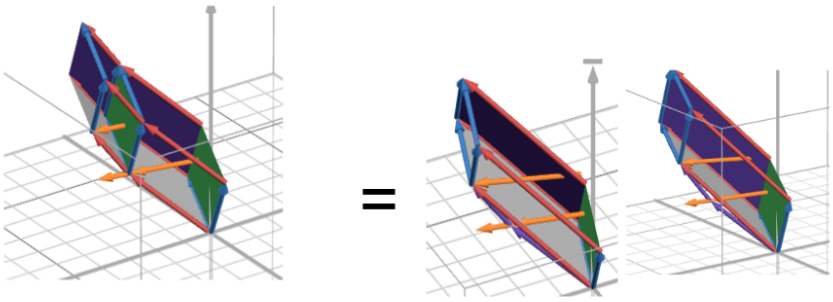


FIGURE 20. (Left) Putting the previous parallelepipeds together. The distance for the second is also shown shifted down, showing how those distances add. (Right)(two images from different angles) The blue vectors are the same, but the red vector is now the sum of the red vectors in the left image(now shown as purple). The initial distance is still shown in the lower orange vectors. While the upper distance(orange) clips through another side, it is equal to the lower distance, showing that in this scenario, when only one vector is changed, the sum of the determinants (left) is equal to the determinant of the matrix with the summed different vectors (right).

Repeating the process with a different vector, the only thing determining the area will still be the distance from the same plane. Therefore, when adding two determinants whose matrices have parallelepipeds with the same base (see Fig 2 and 3) (so the matrices are the same except for one column, call it a for one and b for the other), that is the same as taking the determinant of the matrix where the different column is the combined distances of a and b from the base, which is equal to the determinant of the matrix where the different column is $a + b$. □

Definition 4.4. The **characteristic polynomial** of a matrix is $\det(xI - A)$ for a matrix A.

Definition 4.5. The **principal minor** of order m in a square matrix the determinant of some combination of m rows and the corresponding columns. For example, in the matrix below, a principal minor of order 2 (coming from the bolded values) is $\begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$

$$\begin{bmatrix} \mathbf{1} & 1 & \mathbf{3} \\ 5 & 6 & 5 \\ \mathbf{2} & 3 & \mathbf{4} \end{bmatrix}$$

Theorem 4.6. *In the characteristic polynomial, $x^n + b_1 x^{n-1} + \cdots + b_n$, each b_n is equal to the sum of the principal minors of order $n \cdot (-1)^n$.*

Proof. Given a determinant for a characteristic polynomial, $\det \begin{bmatrix} x-a & -b \\ -c & x-d \end{bmatrix}$, we can set it equivalent to $\det \begin{bmatrix} -a & -b \\ -c & x-d \end{bmatrix} + \det \begin{bmatrix} x & -b \\ 0 & x-d \end{bmatrix}$ per 4.3 Then, we can set those equal to

$$\det \begin{bmatrix} -a & -b \\ -c & -d \end{bmatrix} + \det \begin{bmatrix} -a & 0 \\ -c & x \end{bmatrix} + \det \begin{bmatrix} x & -b \\ 0 & -d \end{bmatrix} + \det \begin{bmatrix} x & 0 \\ 0 & x \end{bmatrix}$$

That shows that the coefficient of x^2 is 1, the coefficient of x^1 is the sum of the principal 1-minors(-d-c) multiplied by $(-1)^1$, and that the coefficient of x^0, b_2 is the sum of the principal (well, only) 2-minor. This process can be done on larger matrices. □

Theorem 4.7. *Where A is an $m \times n$ matrix and B is an $n \times m$ matrix* $\det(xI_n + BA) = x^{n-m}\det(I_m + AB)$

Proof. Given

$$C = \begin{bmatrix} xI_m & A \\ B & I_n \end{bmatrix} \text{and } D = \begin{bmatrix} I_m & 0 \\ -B & xI_n \end{bmatrix}, \text{ we know } \det(CD)=\det(DC). \text{ Therefore,}$$

$$\det \begin{bmatrix} xI_m - AB & xA \\ 0 & xI_n \end{bmatrix} = \det \begin{bmatrix} xI_m & A \\ 0 & xI_n - BA \end{bmatrix} =$$

$$\det(xI_m - AB) \cdot \det(xI_n) = \det(xI_m) \cdot \det(xI_n - BA) = x^n \det(xI_m - AB) = x^m \det(xI_n - BA)$$

Therefore, $\det(xI_n - BA) = x^{n-m}\det(xI_m - AB)$. □

Theorem 4.8. *(Cauchy–Binet) Let A be an $m \times n$ matrix and B be an $n \times m$ matrix. Then, let [n] be the set of columns 1 through n, and $\binom{[n]}{m}$ be the subsets of [n] of size m, then*

$$\det(AB) = \sum_{S \subset \binom{[n]}{m}} \det(A_{[m],S}) \det(B_{S,[m]})$$

In words, this means that when two matrices are multiplied together, their determinant is equal to the product of the determinants of each submatrix, where each $m \times m$ submatrix is obtained by removing columns.

Proof. We begin with theorem 4.7. Given that $\det(xI_n - BA) = x^{n-m}\det(xI_m - AB)$, we can say the sum of the principal minors of size m of BA is equal to the determinant of AB, through $b_m x^{n-m} = x^{n-m}\det(AB)$ so $b_m = \det(AB)$ per 4.6. □

4.1.2. *Eigenvectors.*

Definition 4.9. For a given matrix M, an **eigenvector** is a vector, v such that when a matrix is multiplied by that vector, the vector is multiplied by a scalar, so $Mv = \lambda v$. That scalar is the **eigenvalue**, and will be represented by λ. For example, the matrix $\begin{bmatrix} 2 & 2 \\ 1 & 3 \end{bmatrix}$ has an eigenvector $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ with an eigenvalue of 4 and $\begin{bmatrix} -2 \\ 1 \end{bmatrix}$ with eigenvalue 1. Swapping the rows of a matrix will not change the eigenvalues. All matrices in this section will have eigenvectors.

Definition 4.10. Remember the **characteristic polynomial** of a matrix is $\det(xI - A)$ for a matrix A. Its most interesting property is that its roots are eigenvalues.

4.2. **Spectral Graph Theory Terms and Definitions.**

Definition 4.11. An **adjacency matrix** for some graph G of order n is an $n \times n$ matrix, where $a_{ij} = 1$ if v_i is adjacent to v_j and 0 otherwise.

Definition 4.12. An **ordered incidence matrix**, B, for some graph G with n vertices and m edges, directed in some way, is an $n \times m$ matrix where $a_{ij} = 1$ if the arc (see 2.9) (former edge) a_j starts at vertex v_i, $a_{ij} = -1$ if the arc a_j leads to vertex v_i, and 0 otherwise. Each row (vertex) will have d non-0 entries, where d is the degree of the vertex. Each column (arc) will have 2 non-0 entries, at the two vertices it connects.

Definition 4.13. The **spectrum** of a matrix, $\mathrm{Sp}(A)$, is the set of its eigenvalues and their multiplicities. Typically, the eigenvalues are written in order from greatest to least. In spectral graph theory, spectra are typically taken of symmetric graphs, which ensures that all eigenvalues will be real.

Example: The spectrum of

$$\begin{bmatrix} 4 & \frac{3}{4} & \frac{1}{4} & -\frac{7}{4} \\ 0 & \frac{19}{4} & \frac{1}{4} & -\frac{7}{4} \\ \frac{25}{3} & 0 & 4 & -\frac{25}{3} \\ 5 & \frac{3}{4} & \frac{1}{4} & -\frac{11}{4} \end{bmatrix}$$

is

$$\begin{pmatrix} 4 & 3 & -1 \\ 2 & 1 & 1 \end{pmatrix}.$$

The spectrum of a graph is also equivalent to the sum of its degrees and $2m$.

4.3. **Spectra of Graphs.**

Definition 4.14. A **circulant** matrix is a square matrix of order n, where each row is the prior one, shifted one space to the right. Here is the 3×3 circulant matrix

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_3 & a_1 & a_2 \\ a_2 & a_1 & a_3 \end{bmatrix}$$

Lemma 4.15. *[BR12] Let A be a circulant matrix of order n with first row $(a_1, \ldots, a_n)$. Then, $Sp(A) = \{a_1 + a_2\omega + \cdots + a_n\omega^{n-1}\}$ where ω is a (complex) nth root of 1.*

Proof. The characteristic polynomial of A is $D = \det(xI - A)$, or

$$\det \begin{bmatrix} x - a_1 & -a_2 & \ldots & -a_n \\ -a_n & x - a_1 & \ldots & -a_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ -a_2 & -a_3 & \cdots & x - a_1 \end{bmatrix}$$

Given C_i is the ith column of D, ω is an nth root of 1, and $\lambda_\omega = a_1 + a_2\omega + \cdots + a_n\omega^{n-1}$, replacing C_1 by $C_1 + C_2\omega + \cdots + C_n\omega^{n-1}$ will not change D (because adding multiples of rows to each other does not change the determinant). This leads to the first column being:

$$\begin{bmatrix} x - a_1 - \omega(a_2) - \ldots - \omega^{n-1}(a_n) \\ -a_n + \omega(x - a_1) - \ldots - \omega^{n-1}(a_{n-1}) \\ \vdots \\ -a_2 - \omega(a_3) - \ldots - \omega^{n-1}(x - a_1) \end{bmatrix}$$

Then, substituting in λ_ω into the first column, we get

$$\begin{bmatrix} x - \lambda_\omega \\ \omega(x - \lambda_\omega) \\ \vdots \\ \omega^{n-1}(x - \lambda_\omega) \end{bmatrix}$$

for the first column. The first term is clear, and the second term is, before the λ_ω substitution, $-a_n - (a_1 - x)\omega - a_2\omega^2 - \cdots - a_{n-1}\omega^{n-1}$, which is, because $\omega^n = 1$, equivalent to $-a_n\omega^n - (a_1 - x)\omega - a_2\omega^2 - \cdots - a_{n-1}\omega^{n-1}$, which in turn is equal to $\omega(x - a_n\omega^{n-1} - a_1 - a_2\omega^1 - \cdots - a_{n-1}\omega^{n-2})$, or $\omega(x - \lambda_\omega)$. Because $x - \lambda_\omega$ is a common factor of the column, λ_ω must also be an eigenvalue for all n possible values of ω, making λ_ω run over every eigenvalue. □

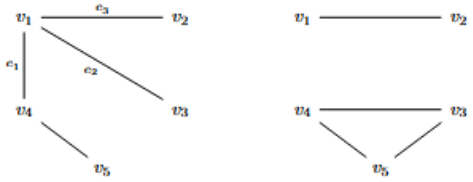


FIGURE 21. Two graphs and their incidence matrices, one with a spanning tree and one without. Each section is separated by a horizontal bar. Note how each section adds to 0

$$B = \begin{bmatrix} 1 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ \hline 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & -1 \end{array}\right]$$

Example 4.16. Spectrum of the Complete Graph (see 2.8) The adjacency matrix of K_n is

$$\begin{bmatrix} 0 & 1 & 1 & \dots & 1 \\ 1 & 0 & 1 & \dots & 1 \\ 1 & 1 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 0 \end{bmatrix}$$

Per the above lemma, $Sp(A) = \lambda_\omega = \omega + \omega^2 + \cdots + \omega^{n-1} = -1$ for $\omega \neq 1$ because the $1 + \omega + \cdots + \omega^{n-1} = 0$.
If, on the other hand, $\omega = 1$, then $\lambda_\omega = 1 + 1^2 + \cdots + 1^{n-1}$, or n-1. Therefore, the spectrum of the complete graph is $\begin{pmatrix} n-1 & -1 \\ 1 & n-1 \end{pmatrix}$.

4.4. **The Laplacian.**

Definition 4.17. The Laplacian, L, is an $n \times n$ matrix, where $L = \text{diag}(d_1, ..., d_n) - A$. The reduced Laplacian, shown by L', is an $(n-1) \times (n-1)$ matrix obtained by removing the ith row and column of L.

Lemma 4.18. *[Pos] The determinant of a part (obtained by removing columns and denoted by S)(effectively a selection of edges) of the reduced incidence matrix,* $\det(B'_S)$*, is* ± 1 *if S is the set of edges of a spanning tree, and is 0 if otherwise.*

Proof. With our $(n-1) \times (n-1)$ matrix, B'_S, first, we assume S does not contain a spanning tree. In that case, it must contain multiple (disconnected) sections, and adding up the rows in a section of B will make a 0 vector, therefore, there will be multiple 0 vectors in B and at least one in B', so the determinant will be 0.

Assuming the converse, if there is a spanning tree, after removing a row (vertex), there must be a vertex with only a 1 or -1, or only one edge e_1 incident, so we add it to the vertex it shares the edge with, so that vertex will only have a 1, and so on. This will end in a matrix with only one non-0 entry per column, and that entry will be 1 (so the unit cube in n-1 dimensions), so the determinant will be ± 1. See Fig 4. □

Lemma 4.19. $L = B \cdot B^T$

Proof. This is most easily thought of conceptually. $B \cdot B^T$ is effectively taking the dot product of every vertex's edges with the edges of every other vertex. If i=j, $a_{ij} = \deg(v_i)$, because they share all of their edges. If $i \neq j$, and if v_i and v_j share x edges, $a_{ij} = -x$, because every edge v_i and v_j share will, when dotted, equal -1 as it will be multiplying 1 and -1, and so that portion will give the adjacency matrix. This will give $B \cdot B^T$=diag $d_1, ...d_n$-A, or L. Removing the ith vertex from B, or B', will similarly lead to $B' \cdot B'^T = L'$ □

Theorem 4.20. *[Pos]* $\tau(G)$*(see 3.1)*$= \det(L')$

Proof. Where B' is an $(n-1) \times m$ matrix that comes from removing one row of the incidence matrix $\det(L') = B' \cdot B'^T$ per the above lemma 4.19, so, per Cauchy-Binet (4.8), we can say

$$\det(L') = \sum_{\substack{S \subset [m] \\ |S|=n-1}} \det(B'_S) \det(B'^T_S) = \sum_{\substack{S \subset [m] \\ |S|=n-1}} \det(B'_S)^2 \tag{4.21}$$

Then, as $\det(B'_S) = \pm 1$ if and only if it is a spanning tree, per 4.18 $\det(B'_S)^2$ effectively adds 1 for every spanning tree, so it is equal to the number of spanning trees.

$$\det(L') = \tau(G)$$

□

4.5. **Graph Energy.**

Definition 4.22. The **energy** of a graph, $\varepsilon(G)$,is defined as the sum of the absolute values of the eigenvalues of a given graph.

Example 4.23. The energy of the complete graph (spectrum found in 4.16), $K_n = |n-1| \cdot 1 + |-1| \cdot (n-1) = 2n-2$

Theorem 4.24. *[BR12, Chapter 4] Where G is a graph of order n and size m,*

$$\varepsilon(G) \leq F(\lambda_1) = \frac{2m}{n} + \sqrt{(n-1)\left[2m - \left(\frac{2m}{n}\right)^2\right]}$$

Proof. Let $Sp(G) = (\lambda_1, \lambda_2, \ldots, \lambda_n)$. Through the Cauchy-Schwarz inequality (for 2 vectors u and v, $(u \cdot v)^2 \leq (u \cdot u)^2 (v \cdot v)^2$), with the two vectors being

$$\left(\begin{bmatrix} |\lambda_2| \\ |\lambda_3| \\ \vdots \\ |\lambda_n| \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}\right)^2 \leq \left(\begin{bmatrix} |\lambda_2| \\ |\lambda_3| \\ \vdots \\ |\lambda_n| \end{bmatrix} \cdot \begin{bmatrix} |\lambda_2| \\ |\lambda_3| \\ \vdots \\ |\lambda_n| \end{bmatrix}\right)\left(\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}\right)$$

$$= \left(\sum_{i=2}^{n} \lambda_i\right)^2 \leq \left(\sum_{i=2}^{n} \lambda_i^2\right)\left(\sum_{i=2}^{n} 1\right).$$

Because $\sum_{i=1}^{n} \lambda_i^2 = 2m$,

$$(\varepsilon(G) - \lambda_1)^2 \leq (2m - \lambda_1^2)(n-1)$$

$$\varepsilon(G) - \lambda_1 \leq \sqrt{(2m - \lambda_1^2)(n-1)}$$

Next, we prove $\lambda_1 \geq \frac{2m}{n}$. For any non-0 vector x, $\lambda_{max} \geq \frac{Ax \cdot x}{x \cdot x}$. Every matrix $n \times n$ (at least those with eigenvectors) can be thought of as three operations: a rotation, so that it is facing along its eigenvectors, a dilation, lengthening each of its eigenvectors by its eigenvalue, and then a second rotation, going to its new shape. Therefore, the most a vector can be lengthened is by λ_{max}, where it is rotated onto the eigenvector with the largest eigenvalue and dilation. When $x = 1$, or the all-1 vector of length n, $\frac{Ax \cdot x}{x \cdot x} = \frac{2m}{n}$ because $Ax \cdot x$ is just the sum of every value in the matrix, or degree

of every vertex, which is $2m$, and $x \cdot x$ is just n, the length of x. Since $\lambda_{max} \geq \frac{2m}{n}$ and because $F(x)$ is strictly decreasing between $\frac{2m}{n}$ and $\sqrt{2m}$ (remember $\sum_{i=1}^{n} \lambda_i^2 = 2m$ so $\lambda_1^2 \leq \sqrt{2m}$), we can state

$$\varepsilon(G) \leq F(\lambda_1) \leq F\left(\frac{2m}{n}\right) = \frac{2m}{n} + \sqrt{(n-1)\left[2m - \left(\frac{2m}{n}\right)^2\right]}$$

□

References

[BR12] B. Balakrishnan and K. Ranganathan. *A Textbook of Graph Theory*. Springer, 2012.

[CLRS09] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. 3rd. MIT Press, 2009.

[Pos] A. Postnikov. *Algebraic Combinatorics, Notes 8*. PDF.