CSC 565: Graph Theory **North Carolina State University**
Fall 2019 **Computer Science**

Lecture 22: November 6, 2019

Lecturer: Don Sheehy <drsheehy@ncsu.edu> *Scribe: Ellis Ackerman, Roshani Narasimhan*

Proving Steinitz's Theorem
With the addition of an additional lemma, we will have the tools to formalize a constructive proof for *Steinitz's Theorem.* (Recall that Steinitz's theorem states that a graph is polytopal iff it is planar and three-connected.)

We will follow the following steps to prove the theorem:

- Use Tutte's algorithm to find a linear embedding of the graph.
- Lift the faces using the Maxwell correspondence.

Lemma: Let G be a planar and 3-connected graph, then either G has K_3 has a non-separating induced cycle, or the dual, G^* has a K_3 as a non-separating induced cycle.

Points to note:

- K_3 is the smallest induced cycle that is possible.
- The dual of a polygon will also be a polygon. In general, the dual of an n-gon will be an n-gon.

Proof. For the sake of a contradiction suppose that both $G = (V, E)$ and its dual, $G^* = (V^*, E^*)$ have no triangle faces. Let $n = |V_G|$, $m = |E_G|$ and f be the number of faces in the embedding of G. Then by Euler's Formula, $n - m + f = 2$, but since we have no triangular faces, every face is bounded by more than three edges. Thus, $2m \geq 4f$ so $m \geq 2f$. Thus, $2 = n - m + f \leq n - 2f + f = n - f$. Now consider G^*. We see that $m \geq 2n$ (since the number of faces and vertices switches in the dual). This leads us to the conclusion $2 = n - m + f \leq n - 2n + f = f - n$. Notice, $2 = n - f$ and $2 = f - n$. We have reached our contradiction. Thus, for a planar and 3-connected graph G, either G or G^* has a triangular face.

□

We are now positioned to prove Steinitz's Theorem.

Theorem 1. *G is polytopal if and only if G is planar and 3-connected.*

Proof. Let G be a planar and 3-connected graph.

Case 1: Additionally, suppose that G has some face which is K_3. Let us embed the graph, via Tutte's Algorithm such that the K_3 face is the outer face. We now lift the embedding from the plane using the Maxwell Correspondence, for everything but the outer face. We now have an an unbounded polyhedron. We now lift the outer-face up, and since in three dimensional space, three points form a flat surface. The outer-face from G becomes the bounding upper face of the polyhedron, and we now have that G is indeed a polytope.

Case 2: Suppose that G has no K_3 face. Then by the Lemma, we know that G^* has a K_3 face. We no proceed to *case 1* and now have a polytope, call it P^*, for G^*. We know that for some polytope, there is a polar-polytope whose graph is its dual. Thus, for P^*, there is a polytope, P, whose graph is $(G^*)^* = G$. The other direction has already been proved, thus we are done. □

Algebraic Graph Theory

To better understand the geometric realizations of graphs, as discussed earlier, and graph morphisms between these realizations, we turn out attention to the algebraic elements of graph theory.

Definition: A *vector space* is a set V and a field F with and addition operation between set elements and scalar multiplication between field and set elements. Formally, the following conditions must be satisfied:

1. $u + v = v + u$ for all $u, v \in V$ (addition commutes).
2. $s(u + v) = su + sv$ for all $u, v \in V$ and $s \in F$ (scalar multiplication distributes).
3. $w + (u + v) = (w + u) + v$ for all $u, v, w \in V$ (addition is associative).
There exists a unique zero vector, $\mathbf{0}$ such that $v + \mathbf{0} = \mathbf{0} + v = v$ for all $v \in V$.
4. There exists a unique additive inverse such that $v + v' = v' + v = \mathbf{0}$ for all $v \in V$, usually we denote the additive inverse for v by $-v$.

Examples of vector spaces include the set of continuous functions and the set of polynomials.

Now consider a function, f, between vector spaces. Let V be a finite dimensional vector space over a field F and U be a finite dimensional vector space over the same field.

Definition: The function $f : V \to U$ is said to be a *linear* if:

1. $f(u + v) = f(u) + f(v)$ for all $u, v \in V$.
2. $f(kv) = kf(v)$ for all $v \in V$ and $k \in F$.

Matrices as maps from one vector space to another: Let us consider matrices to convert a vector in one vector space to a vector in another vector space. Let x be the vector in the U and b be the vector in the vector space V. The matrix A will map the vector x in U to the vector b in V by $Ax = b$. To solve $Ax = b$, we multiply by A^{-1} so that $A^{-1}Ax = A^{-1}b$ and $x = A^{-1}b$. This is essentially transforming b into x, we are trying to invert a linear operation. Note that if we have multiple solutions to map x to b, then A^{-1} won't exist (as solution is not unique), but solutions do exist.

Definition: The *Adjacency Matrix* is the a matrix representation of connectivity in a graph. Consider a graph, $G = (V, E)$ where $n = |V|$. Then the adjacency matrix of G is the matrix,

$$A_G = (a_{ij})_{n \times n} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}.$$

Note that the adjacency matrix is an inefficient data structure to store the graph information. This is because we observe that A_G is symmetric for any graph G. Thus the information we actually need is stored twice. Most real world graphs are sparse and thus the adjacency matrix will predominately contain zeros. (More zeros than ones).

Is the adjacency matrix a graph invariant? Consider the graph, G, below and an isomorphic graph, H, obtained by permuting the vertices of G.

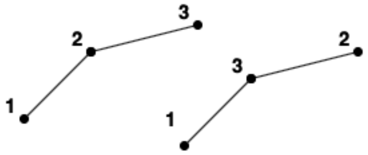

Figure 1: Comparing the adjacency matrices for isomorphic graphs

Clearly, the graphs will not have the same adjacency matrices, since

$$A_G = 010101010 \neq 001001110 = A_H.$$

But if there is a permutation matrix, such that $A_G = P^{-1} A_H P$, then A_G and A_H are said to be similar matrices. In this case, the desired permutation matrix is

$$P = 100001010.$$

While the adjacency matrix is not invariant in the stronger sense, it is invariant across the similarity of matrices. This is not very helpful in practice since finding the permutation matrix is computationally the same as finding an isomorphism between the graphs. Later, we will see an algebraic invariant that we will be able to directly compute, without having to worry about similarity or computing an intermediate step.