3.2 Planar graphs

A *planar graph* is a graph that can be drawn in the plane without crossing. Many results are known for planar graphs (and were likely mentioned in MATH239). Among the most important are the following (some of which assume that the graph is simple, connected and has at least three vertices):

- A planar graph has at most $3n - 6$ edges.
- A planar graph has a vertex of degree at most 5.
- The complete graph K_5 vertices is not planar.
- The complete bipartite graph $K_{3,3}$ is not planar.
- We can test whether a graph is planar in $O(n)$ time (see [HT08] and references therein).
- Every planar graph has a 4-coloring (see [RSST97] and references therein).

We only give a small sample here of how these properties can be useful for algorithms; students interested in more examples should take CS762 (graph-theoretic algorithms).

3.2.1 Clique

Recall that a *clique* is a subset C of vertices that are all mutually adjacent. In other words, it is a subset of vertices that induce a complete graph. We know that K_5 is not a planar graph, so no planar graph can have a clique of size 5 or more. In consequence, there is a very simple polynomial-time algorithm to solve the Clique problem in planar graphs: For any vertex set C with at most four vertices, test whether C forms a clique. Of all sets where the answer is 'YES', return the largest. Since there are $O(n^4)$ sets to test, this algorithm clearly takes polynomial time. (It can actually be implemented in linear time, but no details of this will be given here.)

Theorem 1 *The Clique problem is polynomial in planar graphs.*

3.2.2 Coloring

The status of k-coloring problem is somewhat confusing in planar graphs. As for all graphs, 2-coloring (which is the same as testing whether the graph is bipartite) is polynomial. Also, 4-coloring is polynomial in planar graphs since the answer is always 'yes'. (Finding the coloring is a different question; this is polynomial but not easy [RSST97].) On the other hand, 3-coloring remains NP-hard even in planar graphs. The proof of this is non-trivial (requiring a 'crossing-gadget') and will not be given here.

3.2.3 Maximum Cut

Recall that MaxCut is the NP-hard problem to find a vertex-split $(C, \overline{C})$ (of a given graph $G = (V, E)$) such that as many edges as possible connect C to $\overline{C}$. It turns out that in planar graphs, MaxCut is polynomial as discovered by Hadlock [Had75]. Hadlock's algorithm works by performing a series of reformulations of the problem until we end with one that is clearly polynomial. We will only give the first step here, which is the only one that requires planarity. Figure 3.1(a) shows the graph that we will use as a running example.

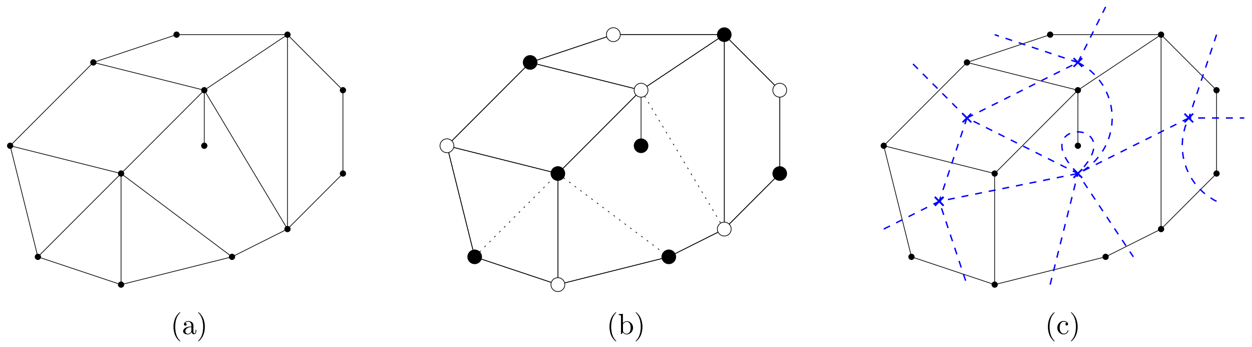

Figure 3.1: (a) A planar graph G with max-cut value $17 = |E| - 3$. (b) Equivalently, after removing three edges the graph is bipartite. (c) The dual graph of the graph in (b) is Eulerian. (The vertex of G^* corresponding to the infinite face is not shown.)

We use an equivalent version of the MaxCut problem: We want to find a minimum cardinality edge-set E' whose removal makes G bipartite. See Figure 3.1(b). Such a set E' is called an *odd circuit cover*, since for every odd circuit (and hence also every odd cycle), at least one edge must be in E'.

For planar graphs, we can use duality to reformulate the problem of finding a minimum odd circuit cover. We first need some definitions and observations. Assume in the following that one particular planar drawing Γ of the planar graph G has been fixed. To avoid some trivialities, we assume that G is connected. A *face* is a maximal connected region of $\mathbb{R}^2 \setminus \Gamma$. The *dual graph* G^* is obtained by defining a vertex v_F for every face F and adding an edge $(v_F, v_{F'})$ whenever the corresponding faces F, F' have an edge in common. (This includes edge (v_F, v_F) if face F is incident to an edge e twice. If this happens then e a *bridge*, i.e., an edge e such that $G \setminus e$ is not connected.) See Figure 3.1(c). We also need the concept of an *Eulerian graph*, which is a connected graph where all vertices have even degree.

Observation 1 *G is bipartite if and only if G^* is Eulerian.*

Proof: For the proof, define a *facial circuit* to be a circuit that bounds a face of G. If G is bipartite, then for any face F the facial circuit must have even length since *all* circuits of G have even length. Each edge e on the circuit gives rise to an edge incident to $\deg(v_F)$ (or two such incidences if e is a bridge), so $\deg(v_F)$ is even. Since G is connected, so is the dual graph, and so G^* is Eulerian.

The other direction holds because every cycle C of G can be "put together" using facial circuits. Formally, let $F_1, \ldots, F_k$ be the faces that are inside C, and let $C_1, \ldots, C_k$ be their facial circuits. Then $\sum_{i=1}^{k} |C_i|$ double-counts all edges that are strictly inside C, and singly-counts every edge on C. Since $|C_i|$ is even, therefore

$$|C| = \underbrace{\sum_{i=1}^{k} |C_i|}_{\text{even}} - \underbrace{2\,\#\{\text{edges strictly inside } C\}}_{\text{even}}$$

is also even. □

We also need to know what deleting an edge corresponds to in the dual graph. The operation of *contracting edge* (v, w) consists of combining the two vertices v, w into one vertex that inherits all edges incident to v and w, except for edge (v, w) itself.

Observation 2 *For any edge e that is not a bridge, deleting e in G corresponds to contracting e in G^*.*

Proof: When we delete e, the two faces F, F' incident to e become one face. (These were different faces because e is not a bridge.) Thus in the dual graph, vertices v_F and $v_{F'}$ become one vertex, which is a contraction. See Figure 3.2 for an example. □

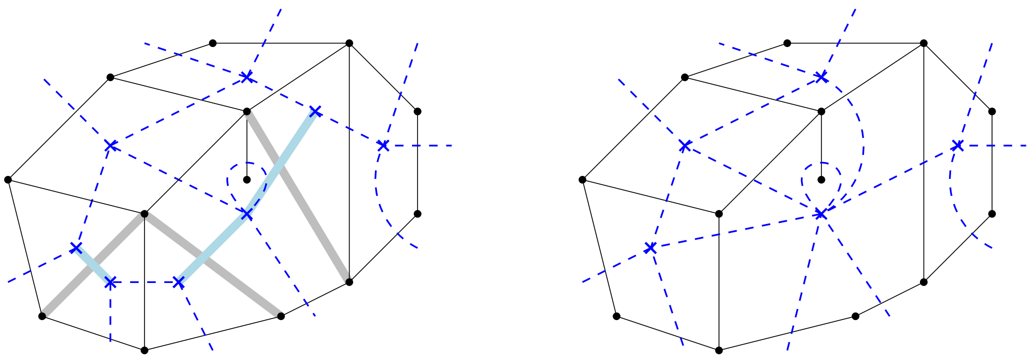

Figure 3.2: The dual graph of Figure 3.1(a). Deleting the three edges shaded edges corresponds to contracting the equivalent edges in the dual graph.

Note that a bridge would never have been part of an odd circuit cover, since it does not belong to any cycle. Therefore with our reformulations we have the following:

Lemma 1 *Fix a connected planar graph G and a planar drawing of G. Finding a maximum cut in G is the same as making G bipartite by deleting a minimum set of edges, which in turn is the same as making G^* Eulerian by contracting a minimum set of edges.*

The problem of finding a minimum set E' of edges such that contracting E' gives an Eulerian graph is called *odd vertex cover*. With a series of further transformations (details omitted), Hadlock showed that odd vertex cover can be solved in polynomial time (for any graph, whether planar or not) via finding a minimum-weight perfect matching. We hence have:

Theorem 2 *MaxCut can be solved in polynomial time in planar graphs.*

Further remarks on duality: The crucial ingredient here was *duality*: The ability to reformulate a problem in a complete different, yet equivalent, way. There are a number of places in algorithm design where duality is a useful tool:

- We have seen here planar graphs and their dual graphs. Whenever one solves a problem in planar graphs, one should think what the corresponding problem in the dual graph is and whether perhaps it is easier to solve. To name just one other example, MaximumFlow (in the special case where source and sink are both on the infinite face) becomes a shortest-path problem when going over to the dual; see [IS79].
- We have earlier seen linear programming problems. For every linear program P, there exists another linear program D (the *dual program*) that has the exact same optimal value, but a complete different set of variables and constraints. This duality can be hugely explored, both for reformulating problems and for better approximation algorithms via the so-called primal-dual method. See any linear programming textbook (for example [Chv83]) for details.
- Finally for any problem that involves points and lines, one can explore *geometric duality*: Every point (x, y) can be mapped to a line with abscissa x and slope y, and vice versa any non-vertical line can be mapped to a point. (This is not the only way in which such a mapping can happen.) See for example [CGL85] for some applications.

To quote from wikipedia (which attributes it to Michael Atiyah): Duality in mathematics is not a theorem, but a principle.

Bibliography

[CGL85] Bernard Chazelle, Leonidas J. Guibas, and D. T. Lee. The power of geometric duality. *BIT*, 25(1):76–90, 1985.

[Chv83] V. Chvátal. *Linear programming*. W. H. Freeman and Company, New York, 1983.

[Had75] F. Hadlock. Finding a maximum cut of a planar graph in polynomial time. *SIAM J. Computing*, 4:221–225, 1975.

[HT08] Bernhard Haeupler and Robert Endre Tarjan. Planarity algorithms via pq-trees (extended abstract). *Electronic Notes in Discrete Mathematics*, 31:143–149, 2008.

[IS79] A. Itai and Y. Shiloach. Maximum flows in planar networks. *SIAM J. Computing*, 8(2):135–150, 1979.

[RSST97] N. Robertson, D. Sanders, P. Seymour, and R. Thomas. The four-colour theorem. *J. Combin. Theory Ser. B*, 70(1):2–44, 1997.