-- --

Counting Embeddings of Planar Graphs Using DFS Trees [1]

Jiazhen Cai

Courant Institute, NYU
New York, NY 10012

ABSTRACT

Previously counting embeddings of planar graphs [5] used *P-Q* trees and was restricted to biconnected graphs. Although the *P-Q* tree approach is conceptually simple, its implementation is complicated. In this paper we solve this problem using DFS trees, which are easy to implement. We also give formulas that count the number of embeddings of general planar graphs (not necessarily connected or biconnected) in $O(n)$ arithmetic steps, where n is the number of vertices of the input graph. Finally, our algorithm can be extended to generate all embeddings of a planar graph in linear time with respect to the output.

Key words. graph, depth first search, embedding, planar graph, articulation point, connected component

AMS(MOS) subject classifications. 68R10, 68Q35, 94C15

1. Introduction

In [14], Wu stated four basic planar graph problems:

1. Decide whether a connected graph G is planar.

2. Find a minimal set of edges the removal of which will render the remaining part of G planar.

3. Give a method of embedding G in the plane in case G is planar.

4. Enumerate and count all possible planar embeddings of G in the plane in case G is planar.

---

1. Part of this work was done while the author was visiting the University of Wisconsin - Madison. The research of this author was partially supported by National Science Foundation grant CCR-9002428.

Wu solved all these problems using systems of algebraic equations. His solutions are elegant, but his implementations are not so efficient. Other solutions to these problems basically follow two different approaches. One uses DFS trees [4, 8]; and the other uses *P*-*Q* trees [3, 5, 9-11].

The *P*-*Q* tree approach is considered to be conceptually simpler, but its implementation is much more complicated. Efficient *P*-*Q* tree solutions have been discovered for all the four problems. Lempel, Even and Cederbaum [10] solved problem 1. Chiba et al solved problems 3 and 4 [5]. These solutions are all linear-time. Recently, Di Battista and Tamassia [6] have claimed an $O(logn)$-time-per-operation solution to the problem of maintaining a planar graph under edge additions, which implies an $O(m \log n)$-time solution to problem 2. Here m is the number of edges and n is the number of vertices of the input graph. On the other hand, the DFS tree approach was used only for problems 1 and 2: a linear-time DFS tree algorithm (the HT algorithm) for problem 1 was given by Hopcroft and Tarjan [8] in 1974, and an $O(m \log n)$-time algorithm for problem 2 was given by Cai, Han, and Tarjan [4] recently. The HT algorithm can also be extended to solve Problem 3, but the modification is complicated.

The previous solutions for the four planar graph problems all consider biconnected graphs only. The extension from biconnected graphs to general graphs is straightforward for problems 1, 2, and 3, but not for problem 4. For connected graphs, Stallmann [12] solved the enumeration version of problem 4 in time linear to the size of the output, but his solution for the counting problem is complicated and cannot be accomplished in polynomial time. For unconnected graphs, we know no published solution for problem 4.

In this paper, we give an $O(n)$-time DFS tree solution for the counting version of problem 4. While the *P*-*Q* tree solution in [5] only counts the embeddings of biconnected graphs, we also solve the interesting combinatorial problem of counting embeddings of general graphs. Our algorithms extend easily to generate one embedding or all embeddings of a planar graph in time linear to the input and output, hence solve problems 3 and 4. Thus, we complete the DFS tree solutions for the four planar graph problems.

The rest of the paper is organized as follows. Section 2 is preliminaries. We solve the counting problem for biconnected graphs in Section 3 and then show how to count embeddings for more general planar graphs in Sections 4 and 5.

2. Preliminaries

Consider an undirected graph $G = (V, E)$ with vertex set V and edge set E. Denote $|V|$ by n and $|E|$ by m. We assume that G has no self loops and has no multiple edges. We can draw a picture H on a surface, which can be either a plane or a sphere, as follows: for each vertex $v \in V$, we draw a distinct node v'; for each edge $(v, w) \in E$, we draw a simple arc connecting the two nodes v' and w'. We call this arc an *embedding* of the edge (v, w). If arcs of H do not cross each other, we say that H is an *embedding* of G. An embedding on the plane is called a *planar embedding*, and an embedding on the sphere is called a *sphere embedding*. It is easy to see that G has a planar embedding iff it has a sphere embedding. If G has an embedding, then we say that G is *planar*. Since we are interested only in graphs with no isolated vertices, we will frequently identify graphs with their edge sets.

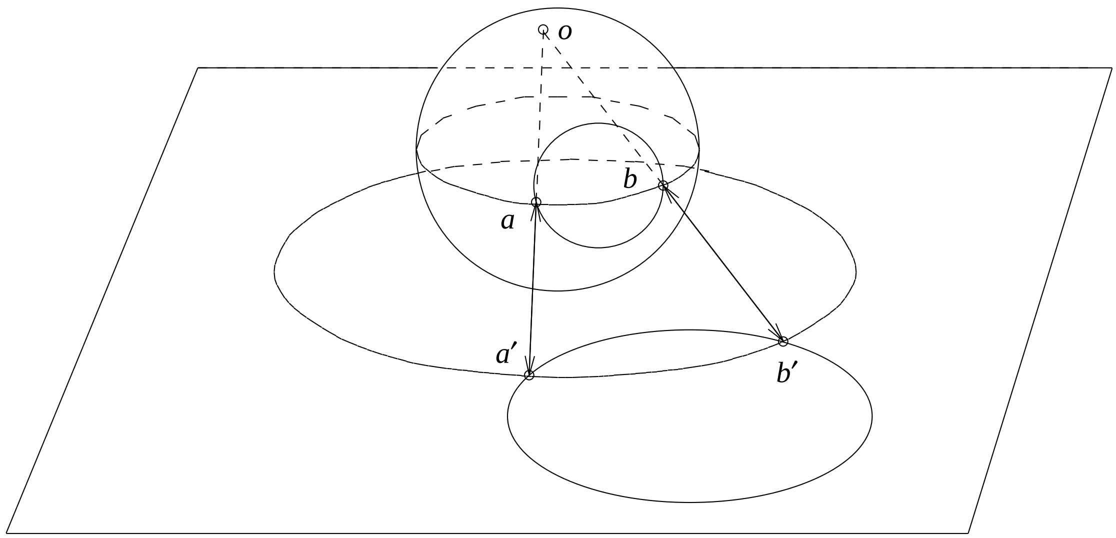


Fig. 1 Sphere projection

One easy transformation between planar embeddings and sphere embeddings is the sphere projection shown in Fig. 1. Under the sphere projection, each point on the sphere, except the projection center o, has a distinct image on the plane, and each point on the plane is the image of some point on the sphere. Let H be a sphere embedding of a graph G with f faces. According to Euler's formula [2], if G has m edges, n vertices and c connected components, then $f = m - n + c + 1$. Using the sphere projection, we can get f topologically different planar embeddings of G from a given sphere embedding of G by selecting the center of projection in different faces. Thus, if G has N sphere embeddings, then it has Nf planar embeddings.

We will represent embeddings by their *planar maps* and *adjacency relations*. A *planar map* M for a given embedding H of G is a mapping from V to lists of E such that for each $v \in V$, $M(v)$ gives the clockwise circular ordering of the edges around v in H. In this case, we say that H and M *match* each other. For connected graphs, sphere embeddings with the same planar map are topologically equivalent. Therefore we need only count planar maps in this case. However, for graphs with more than one connected components, planar maps do not specify the relative positions of the embeddings of different connected components.

Let H be a sphere embedding of G. We define an *adjacency relation* R on the set of faces of the embeddings of different components in H as follows. Let C_1, ..., C_k be the connected components of G, and H_1, ..., H_k be the embeddings of C_1, ..., C_k in H respectively. We say that two embeddings H_i and H_j are *neighbors* of each other in H if there is a face in H whose boundary contains edges from both C_i and C_j. If C_i and C_j are neighbors in H, then there is a face F_i of H_i that contains H_j, and a face F_j of H_j that contains H_i. In this case, we say the two faces F_i and F_j are *adjacent* to each other, and the unordered pair (F_i, F_j) is in R. Thus, in general, a sphere embedding can be specified by a planar map plus an adjacency relation.

The following facts are important to our discussion:

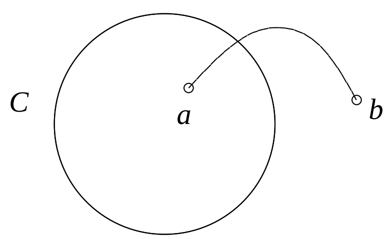


Fig. 2

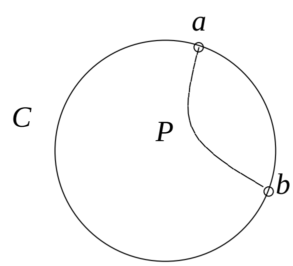


Fig. 3

Observation 1. Let C be a simple closed curve on the plane as in Fig. 2; let a be a point inside C and b be a point outside C. Then any curve that joins a and b will cross C.

Observation 2. Let G_1 be the undirected graph represented by Fig. 3, where P is a path joining the two vertices a and b on cycle C. Then in any embedding of G_1, all the edges of path P are on the same side of the cycle C.

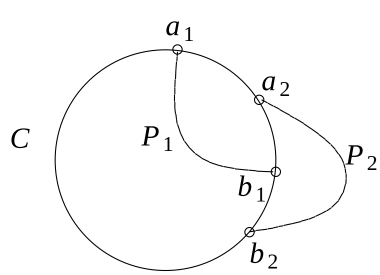


Fig. 4

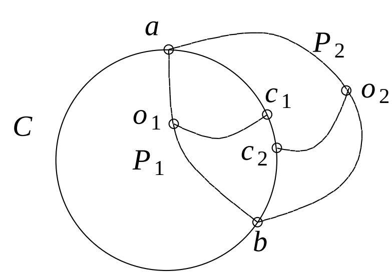


Fig. 5

Observation 3. Let G_2 be the undirected graph represented by Fig. 4, where a_1, a_2, b_1 and b_2 are four distinct vertices that appear in order on C. Then in any embedding of G_2, the two paths P_1 and P_2 are on opposite sides of the cycle C.

Observation 4. Let G_3 be the undirected graph represented by Fig. 5, where a, c_1, c_2 and b are vertices that appear in order on C, and c_1 and c_2 may be the same. Then in any embedding of

G_3, the two subgraphs P_1 (containing paths from o_1 to a, b and c_1) and P_2 (containing paths from o_2 to a, b and c_2) are on opposite sides of the cycle C.

All four observations above are intuitively obvious and can be proved by the Jordan Curve Theorem [7, 13].

3. Number of embeddings for biconnected graphs

We first discuss how to count planar maps of biconnected graphs. We will reduce this problem into a sequence of successively simpler problems before we eventually solve it.

In this section we assume that $G = (V, E)$ is given in its DFS representations [1], where $V = \{1, ..., n\}$ is the set of DFS numbers of the vertices in G, and E is partitioned into a set of tree edges T and a set of back edges B. If $[v, w]$ is a tree edge, then $v < w$. If $[v, w]$ is a back edge, then $w < v$, and there is a tree path in T from w to v. In either case, we say that $[v, w]$ *leaves* v and *enters* w, and is *connected* to v and w.

We define *successor*s for both vertices and edges. If $[v, w]$ is a tree edge, then w is a *successor* of v. If $[v, w]$ is a tree edge, and $[w, x]$ is any edge, then $[w, x]$ is a *successor* of $[v, w]$. Back edges have no successors. We also define descendants and ancestors for both vertices and edges. A *descendant* of vertex (resp. edge) x is defined recursively as either x itself or a successor of a descendant of x. If y is a descendant of x, then x is an *ancestor* of y. If y is a successor of x, then x is a *predecessor* of y.

In this section, we also assume that G is a biconnected graph with at least two edges. Then each tree edge has at least one successor, and T forms a tree with only one edge leaving the root.

Let $e = [v, w] \in E$. Let Y be the set of vertices y such that there exists a back edge $[x, y]$ that is a descendant of e. Then Y is not empty. We define $low_1(e)$ to be the smallest integer in Y, and $low_2(e)$ to be the second smallest integer in $Y \cup \{n+1\}$. The two mappings low_1 and low_2 can be computed in $O(m)$ time during the depth-first-search on G [8]. Since G is biconnected, it has no articulation points. Thus, if v is not the root of T, then $low_1(e) < v$. [1]

As in [8], we define the function ϕ on E as follows.

$$\phi(e) = \begin{cases} 2\, low_1(e) & \text{if } low_2(e) \geq v, \ where\ e = [v, w] \\ 2\, low_1(e) + 1 & \text{otherwise} \end{cases}$$

For each vertex $v \in V$, we arrange all the edges leaving v into a list $\Phi(v)$ in increasing order by their ϕ values. The ordering Φ can be computed in $O(m)$ time using a bucket sort. The first edge in $\Phi(v)$ is called the *reference edge* of v, denoted by $e_{v,ref}$. We use E_0 to represent the set of all non-reference edges in E.

For $e = [v, w] \in E$, we define $S(e)$, the *segment* of e, to be the subgraph of G that consists of all the descendants of e. We use $ATT(e)$ to denote the set of back edges $[x, y]$ in $S(e)$ such that y is an ancestor of v. Each back edge in $ATT(e)$ is called an *attachment* of e. Thus, if $[x, y]$ is an attachment of e, then $low_1(e) \le y \le v$. If $low_1(e) < y < v$, then we say that $[x, y]$ is *normal*. Otherwise we say that $[x, y]$ is *special*.

For each edge $e = [v, w] \in E$, we define $cycle(e)$ as follows: if e is a back edge, then $cycle(e) = \{e\} \cup \{e'$: e' belongs to the tree path from w to $v\}$; if e is a tree edge, then $cycle(e) = cycle(e_{w,ref})$. Since we assume that G is a biconnected graph with more than one edge, then for any edge $e = [v, w] \in E$, $cycle(e)$ is defined. The only edge on $cycle(e)$ that enters v is denoted by $e_{v,in}$. If v is not the root, then $e_{v,in}$ is the only tree edge entering v. Each embedding C_e of $cycle(e)$ is a simple closed curve, which divides the plane (or sphere) into two regions. When we travel on C_e along the direction of its edges, we see one region on the left hand side and the other region on the right hand side. We use $sub(e)$ to denote the subgraph $S(e) \cup cycle(e)$. It is easy to see that the vertex $low_1(e)$ is always on $cycle(e)$, and $sub(e) - S(e) = \{e'$: e' belongs to the tree path from $low_1(e)$ to $v\}$. If e is the only tree edge leaving the root, then $sub(e)$ is the whole graph.

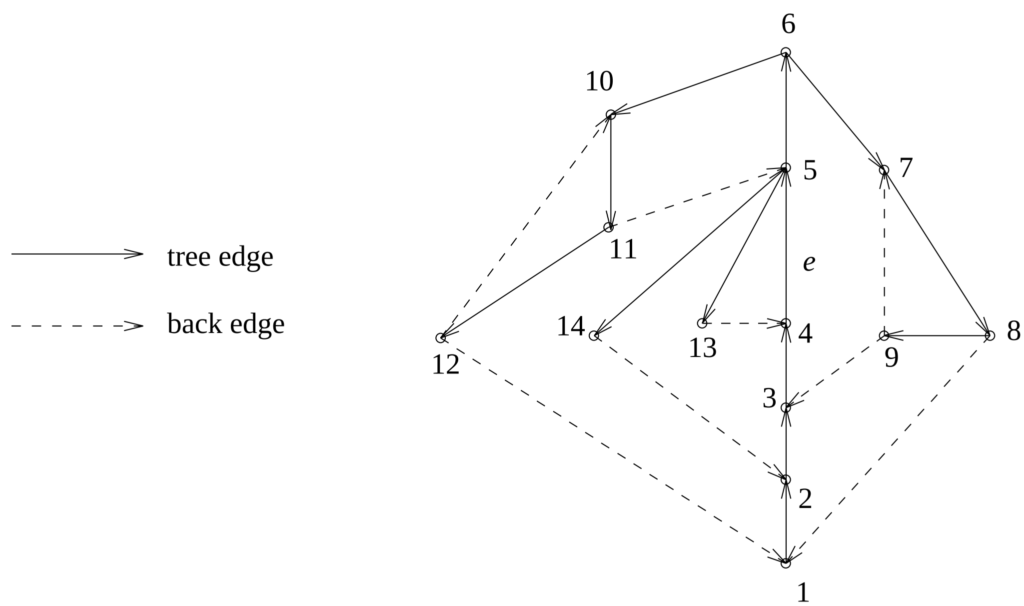


Fig. 6

Fig. 6 illustrates some of these definitions, where $e = [4, 5]$; $low_1(e) = 1$; $low_2(e) = 2$; $cycle(e) = \{ [4, 5], [5, 6], [6, 7], [7, 8], [8, 1], [1, 2], [2, 3], [3, 4] \}$; $S(e)$ contains all the edges in the graph except [1, 2], [2, 3], [3, 4]; $sub(e)$ is the whole graph; $ATT(e) = \{[8, 1], [9, 3], [12, 1], [14, 2], [13, 4]\}$.

3.1. Partial maps

Let H be an embedding of G. Let M be the planar map of H. For each $v \in V$, we assume that the list $M(v)$ starts from the edge $e_{v,in}$. For any vertex v in V and any two edges e_i and e_j connected to v, if e_i appears before e_j in $M(v)$, then we say that e_i is *embedded on the left of* e_j, and e_j is *embedded on the right of* e_i in H.

A mapping M' from V to lists of edges in E is call a *partial map* of G if there is a planar map M of G such that for each $v \in V$, $M'(v)$ can be obtained from $M(v)$ by deleting all the edges entering v. In this case, we say that M is an *extension* of M'. If H is an embedding that matches M, we also say that H and M' match each other. The following lemma establishes the one-to-one correspondence between planar maps and partial maps.

LEMMA 1. *If M' is a partial map of G, then there is a unique planar map M of G that is an extension of M'.*

Proof Let H be an embedding of G that matches M'. Let M be the planar map of H. We show that M is uniquely determined by M'.

Let *label* be a numbering of back edges from 1 to $|B|$ such that for any $v \in V$, for any two edges e_i and e_j leaving v, and for any two back edges $t_i \in S(e_i)$ and $t_j \in S(e_j)$, if $M'(v) = [..., e_i, ..., e_j, ...]$, then $label(t_i) < label(t_j)$. It is clear that *label* is uniquely determined by M'.

Let $v \in V$. Let e be an edge leaving v. Let $in(e)$ be the set of back edges in $S(e)$ entering v, not including $e_{v,in}$. Let $back(e)$ be the unique back edge on $cycle(e)$. Consider an edge $t \in in(e)$. By the definition of *label*, we know that t is embedded on the left of e in H iff $label(t) < label(back(e))$. Thus, the position of t in $M(v)$ relative to e is uniquely determined by M'.

Then consider two edges t_1 and t_2 in $in(e)$ such that either $label(t_1) < label(t_2) < label(back(e))$ or $label(back(e)) < label(t_1) < label(t_2)$. Again by the definition of *label*, we know that t_1 is embedded on the right of t_2. Thus, for any two edges in $in(e) \cup \{e\}$, their relative positions in $M(v)$ are uniquely determined by the mapping *label*.

Now consider any two edges e_i and e_j in $M'(v)$ such that e_i appears before e_j in $M'(v)$. Since G is biconnected, then all edges in $in(e_i) \cup \{e_i\}$ are embedded on the left of all the edges in $in(e_j) \cup \{e_j\}$ in H. Thus $M(v)$ is uniquely determined by *label*. ■

Therefore, to count planar maps, we need only to count partial maps.

The above proof also suggests a simple linear-time algorithm that builds a planar map M from a partial map M'. First we compute the mappings *label*, *back*, and *in* in a depth-first-search on G, which takes $O(n)$ time (recall that for a planar graph, $m = O(n)$.) Then for each edge $e = [v, w] \in E$, we split $in(e)$ into two lists $L_e = [l_1, ..., l_i]$ and $R_e = [r_1, ..., r_j]$ such that $label(r_1) > ... > label(r_j) > label(back(e)) > label(l_1) > ... > label(l_i)$. This can be done again in $O(n)$ time using a bucket sort. For each v in V, let $M'(v) = [e_1, ..., e_k]$. Then $M(v) = [e_{v,in}] + L_{e_1} + [e_1] + R_{e_1} + ... + L_{e_k} + [e_k] + R_{e_k}$, where + is the list concatenation.

3.2. Singular edges

We call an edge $e = [v, w]$ in E_0 *singular* if $low_2(e) \geq v$. A set of all singular edges leaving the same vertex and having the same low_1 value is called a *singular set*. We have,

LEMMA 2. *Let M' be a partial map of G. Let $e_i = [v, w_i]$ and $e_j = [v, w_j]$ be two edges on the same side of $e_{v,ref}$ in $M'(v)$. If $\phi(e_i) = \phi(e_j)$ then both e_i and e_j are singular.*

Proof We prove this lemma by contradiction. Suppose one of e_i and e_j, say e_i, is not singular. Then $low_2(e_i) < v$. Since $\phi(e_i) = \phi(e_j)$, then $low_2(e_j) < v$ also. By Observation 4, $S(e_i)$ and $S(e_j)$ cannot be embedded on the same side of $cycle(e_{v,ref})$. Therefore e_i and e_j cannot be embedded on the same side of $e_{v,ref}$, a contradiction. ■

LEMMA 3. *Let $e_i = [v, w_i]$ and $e_j = [v, w_j]$ be two edges in a singular set. Let M' be any partial map of G. Let M'_1 be a mapping obtained from M' by switching the positions of the two edges e_i and e_j in $M'(v)$. Then M'_1 is also a partial map of G.*

Proof Let H be an embedding of G that matches M'. Since e_i and e_j are in the same singular set, then $low_1(e_i) = low_1(e_j)$. Also, v and $low_1(e_i)$ are the only two vertices that are shared by $S(e_i)$, $S(e_j)$ and the rest of G. Therefore, either one of $S(e_i)$ and $S(e_j)$ can be re-embedded into any face in H whose boundary contains the the two vertices v and $low_1(e_i)$. In particular, we can obtain another embedding H' of G from H by switching the positions of the embeddings of $S(e_i)$ and $S(e_j)$. Then M'_1 is the partial map that matches H'. ■

3.3. Feasible maps and valid partitions

If U is a set, and X, Y are two disjoint sets such that $X \cup Y = U$, then we call $[X, Y]$ an *ordered partition* of U. Let $Q = [LL, RR]$ be an ordered partition of E_0. We say that Q is a *valid partition* of E_0 if there exists an embedding H of G such that in H, each edge $[v, w] \in LL$ is embedded on the left of $e_{v,ref}$, and each edge $[v, w] \in RR$ is embedded on the right of $e_{v,ref}$. In this case we say that Q is *derived* from H. If M is a planar map or partial map of G that matches H, we also say that Q is derived from M.

Let M' be a mapping from V to lists of edges in E such that for each $v \in V$, $M'(v)$ is a permutation of the edges leaving v. We call M' a *feasible map* of G if there exists a valid partition $Q = [LL, RR]$ of E_0 so that for all $v \in V$, if $M'(v) = [l_1, ..., l_s, e_{v,ref}, r_1, ..., r_t]$, then

(1) $l_1, ..., l_s \in LL$, and $r_1, ..., r_t \in RR$.

(2) $\phi(l_1) \geq ... \geq \phi(l_s)$ and $\phi(r_1) \leq ... \leq \phi(r_t)$.

LEMMA 4. *A mapping M' from V to lists of edges in E is a partial map of G iff M' is a feasible map of G.*

Proof

⇒ Suppose M' is a partial map. Let H be an embedding of G that matches M', and $Q = [LL, RR]$ be the unique valid partition derived from H. Let $v \in V$. Let $M'(v) = [l_1, ..., l_s, e_{v,ref}, r_1, ..., r_t]$. Then condition (1) in the definition of feasible map is trivially true. To see condition (2) is also true, consider two edges e_i and e_j in $M'(v)$ with $\phi(e_i) > \phi(e_j)$. We need to show that (i) if both e_i and e_j belong to LL, then e_i appears before e_j in $M'(v)$; and (ii) if both of them belong to RR, then e_i appears after e_j in $M'(v)$. Assume that both e_i and e_j are in LL. Then e_i, therefore the whole $S(e_i)$, is embedded on the left of $cycle(e_{v,ref})$. The condition $\phi(e_i) > \phi(e_j)$ implies that there is a back edge $[x, y]$ in $S(e_i)$ such that $low_1(e_j) < y < v$. Since the tree path from $low_1(e_j)$ to v is shared by $cycle(e_{v,ref})$ and $cycle(e_j)$, then $[x, y]$, therefore $S(e_i)$, is embedded on the left of $cycle(e_j)$. Thus e_i appears before e_j in the list $M'(v)$. The discussion for the situation (ii) is similar.

⇐ Suppose M' is a feasible map. Then there exists a valid partition $Q = [LL, RR]$ such that for all $v \in V$, if $M'(v) = [l_1, ..., l_s, e_{v,ref}, r_1, ..., r_t]$, then conditions (1) and (2) are satisfied. Let M be the partial map of G from which Q is derived. By the *only if* part of Lemma 4, M is also a feasible map of G with respect to Q. The conditions (1) and (2) in the definition of feasible map implies that for each $v \in V$, $M'(v)$ can be obtained from $M(v)$ by permuting edges with the same ϕ values within $\{ l_1, ..., l_s\}$ and $\{ r_1, ..., r_t\}$. By Lemma 2 and 3, M' is also a partial map. ■

By Lemma 4, we need only to count feasible maps, which can be constructed easily from valid partitions.

3.4. *SAME* and *DIFF*

Let H be an embedding of G. For convenience, we say that an edge $e = [v, w] \in E_0$ is *red* in H if e is embedded on the left of $e_{v,ref}$, and *blue* otherwise. We partition E_0 into equivalence classes called *group*s. Two edges in E_0 are in the same group iff they have the same color in each embedding of G. We call the set of such groups *SAME*. We further organize these groups into *pairs*. Two groups W and Z in *SAME* are put into one (unordered) pair (W, Z) iff the color of the edges in W is always different than the color of the edges in Z. We call the set of such pairs *DIFF*. We will show in Section 3.6 that the two sets *SAME* and *DIFF* can be computed in $O(n)$ time during planarity testing.

Let $Q = [LL, RR]$ be an ordered partition of E_0. We say that Q is *consistent with SAME* if each group in *SAME* is totally contained in either LL or RR. We say that Q is *consistent with DIFF* if for each pair $(W, Z) \in DIFF$, one of the two groups W and Z is contained in LL and the other is contained in RR.

By the definition of *DIFF* and *SAME*, any valid partition of E_0 is consistent with *SAME* and *DIFF*. We will further prove that any ordered partition of E_0 consistent with *SAME* and *DIFF* is valid. For this we need some more definitions and lemmas.

Let $e = [v, w]$ be a tree edge. Let $\Phi(w) = [e_1, ..., e_k]$. Let $Q = [LL, RR]$ be an ordered partition of E_0. For $i = 1, ..., k$, let $G_i = sub(e_1) \cup ... \cup sub(e_i)$. Let H_i be an embedding of G_i. We say that H_i is *conformable* to Q (w.r.t. e) if around each vertex $u \geq w$ in H_i, all the edges embedded on the left of $e_{u,ref}$ belong to LL and all the edges embedded on the right of $e_{u,ref}$ belong to RR. By convention, any embedding of $sub(e)$ is conformable to Q (w.r.t. e) if e is a back edge.

Let $[x, y]$ be an attachment of e not on $cycle(e)$. Let $[a, b]$ be the nearest ancestor of $[x, y]$ such that a is on $cycle(e)$. We call $[a, b]$ the *root* of $[x, y]$ (w.r.t. e), denoted by $root([x, y])$. We prove the following lemma.

LEMMA 5. *If $[x, y]$ is an attachments of e in G_{i-1} not on $cycle(e)$, and $low_1(e_i) < y$, then there is a pair (W, Z) in DIFF such that $e_i \in W$ and $root([x, y]) \in Z$, where $1 < i \leq k$.*

Proof Let $[a, b] = root([x, y])$. Let W be the group in *SAME* containing e_i, and Z be the group in *SAME* containing $[a, b]$. Let P_1 be the simple directed path in $sub(e)$ whose first edge is $[a, b]$ and whose last edge is $[x, y]$. Let P_2 be a simple directed path in $sub(e)$ whose first edge is e_i and whose last vertex is $low_1(e_i)$. Consider two cases.

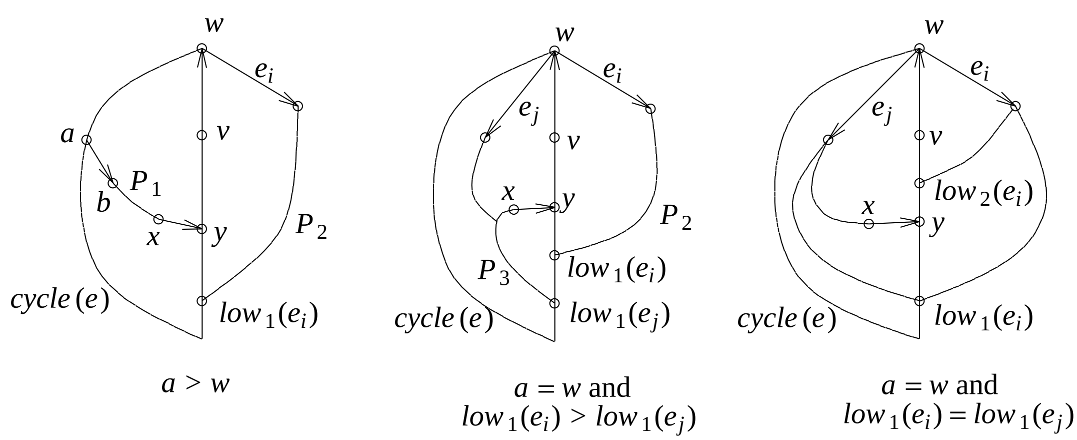


Fig. 7

Case 1. $a > w$. By Observation 3, P_1 and P_2 cannot be embedded on the same side of $cycle(e)$ in any embedding of G (see Fig. 7). Therefore $(W, Z) \in DIFF$.

Case 2. $a = w$. In this case, $[a, b] = e_j$ for some $1 < j < i$, and $low_1(e_j) \leq low_1(e_i) < y$. Then there must be an undirected simple path P_3 in $sub(e_j)$ between $low_1(e_j)$ and y that contains x. If $low_1(e_j) < low_1(e_i) < y$, then P_3 and P_2 cannot be embedded on the same side of $cycle(e)$ by Observation 3. If $low_1(e_j) = low_1(e_i)$, then $low_2(e_j) \leq y < w$. Therefore $low_2(e_i) < w$ (recall that $\phi(e_i) \geq \phi(e_j)$). Thus $S(e_i)$ and $S(e_j)$ cannot be embedded on the same side of $cycle(e)$ by Observation 4. In either case, e_i and e_j cannot be embedded on the same side of $cycle(e)$, and therefor $(W, Z) \in DIFF$. ■

LEMMA 6. *Let $Q = [LL, RR]$ be an ordered partition of E_0 consistent with SAME. Let H_e be an embedding of $sub(e)$ conformable to Q. If $e \in LL$ (RR), then all the normal attachments of e are embedded on the left (right) hand side of $cycle(e)$ in H_e.*

Proof Assume wlog that $e \in LL$. Let $[x, y]$ be a normal attachment of e. Let $[a, b] = root([a, b])$. Let P_1 be the simple directed path whose first edge is $[a, b]$ and whose last edge is $[x, y]$. Let e' be the predecessor of e. Note that the tree path from $low_1(e)$ to v is shared by $cycle(e)$ and $cycle(e')$. By Observation 2, P_1 and e are always on the same side of $cycle(e')$ in any embedding of G. Thus, if e is embedded on the left (right) of $cycle(e')$, then $[a, b]$ must be embedded on the left (right) of $cycle(e)$. This means that e and $[a, b]$ are in the same group of *SAME*. Since Q is consistant with *SAME*, and $e \in LL$, then $[a, b] \in LL$. Since H_e is conformable to Q, then $[a, b]$, and therefore $[x, y]$, are embedded on the left hand side of $cycle(e)$. ■

Now we prove the main lemma of this subsection.

LEMMA 7. *An ordered partition* $Q = [LL, RR]$ *of* E_0 *is valid if it is consistent with SAME and DIFF.*

Proof Assume Q is consistent with *SAME* and *DIFF*. To see that Q is valid, we show that there exists a planar embedding of G from which Q can be derived. For this purpose, we show by induction that for all $e = [v, w] \in E$, we can construct an embedding H_e of $sub(e)$ that is conformable to Q.

If e is a back edge, then any embedding of $sub(e)$ is conformable to Q by convention.

Next we assume that $e = [v, w]$ is a tree edge with $\Phi(w) = [e_1, ..., e_k]$, and for each $i = 1, ..., k$, there is a planar embedding H_{e_i} of $sub(e_i)$ that is conformable to Q (w.r.t. e_i).

To construct H_e, we first let $H_1 = H_{e_1}$. Then for $i = 2, ..., k$, we add H_{e_i} into H_{i-1} to get H_i. As a result, we will have $H_e = H_k$.

Consider adding H_{e_i} to H_{i-1}, where $1 < i \le k$. Assume inductively that H_{i-1} is conformable to Q (w.r.t. e). Also assume wlog that $e_i \in LL$. By Lemma 6, all the normal attachments of e_i are embedded on the left of $cycle(e_i)$ in H_{e_i}. Thus, with the sphere projection, we can transform H_{e_i} into a planar embedding of $sub(e_i)$ in which the tree path from $low_1(e_i)$ to w borders the outer face.

If there is no attachment of e embedded on the left of $cycle(e)$ in H_{i-1}, we can embed H_{e_i} to the left of $cycle(e)$ in the face whose boundary contains the tree path from $low_1(e)$ to w. Otherwise, let $[x, y]$ be one of the highest attachments of e embedded on the left of $cycle(e)$ in H_{i-1} (We say an attachment $[x, y]$ is *higher* than another attachment $[x', y']$ if $y > y'$.) Let $[a, b] = root([x, y])$. By induction hypothesis, H_{i-1} is conformable to Q. Therefore $[a, b] \in LL$. Since $e_i \in LL$ also, there can be no pair (W, Z) in *DIFF* such that $e_i \in W$ and $[a, b] \in Z$. By Lemma 5, $low_1(e_i) \ge y$. Then we can embed H_{e_i} into H_{i-1} on the left side of $cycle(e)$ in the face whose boundary contains the tree path from y to w. In this way, e_i is embedded on the left of $e_1, ..., e_{i-1}$, and H_i is conformable to Q. ■

According to Lemmas 1 , 4, and 7, all planar maps of G can be easily generated from the function ϕ and the two sets *SAME* and *DIFF* as follows:

1. Generate valid partitions using Lemma 7;

2. For each valid partition generated in 1, generate partial maps using Lemma 4;

3. For each partial map generated in 2, construct a planar map using the method described at the end of Section 3.1.

3.5. Counting planar maps

To count the number of planar maps, we further simplifier the problem as follows. We arbitrarily select a representative from each singular set. If M' is a feasible map, and M'' is obtained from M' by deleting all non-representative singular edges, then we say M'' is a *reduced map* from M', and M' is *generated* from M''. Similarly, if Q is a valid partition, and Q' is obtained from Q by deleting all non-representative singular edges, then Q' is called a *reduced partition*. If M'' is a reduced map from M', Q' is a reduced partition from Q, and Q is derived from M', then we also say that Q' is *derived* from M'', and M'' is *constructed* from Q'. It is not difficult to see that from each reduced map, we can derive a unique reduced partition, and from each reduced partition, we can construct a unique reduced map. Thus, to count feasible maps, we can first count reduced partitions, then count the feasible maps that can be generated from each reduced map.

To count reduced partitions, let $SAME'$ and $DIFF'$ be obtained from $SAME$ and $DIFF$ respectively by deleting all the non-representative singular edges. A pair $[W, Z]$ in $DIFF'$ is *trivial* if either W or Z is empty. By Lemma 7, it is easy to see that if $[L, R]$ is an ordered partition of $SAME'$ such that neither L nor R contains groups from the same nontrivial pair in $DIFF'$, then $[\bigcup_{W \in L} W, \bigcup_{W \in R} W]$ is a reduced partition. Let d be the number of nontrivial pairs in $DIFF'$, and s be the number of nonempty sets in $SAME'$ that are not contained in any of the nontrivial pairs in $DIFF'$. Then there are 2^{d+s} reduced partitions and therefore 2^{d+s} reduced maps.

Next we consider the number of feasible maps that can be generated from each reduced map. Let $singular(e)$ be the singular set containing e, and $same(e)$ be the group in $SAME$ containing e. Immediately from Lemma 3 and its proof we have,

LEMMA 8.

(i) Let e be a singular edge. If $|same(e)| > 1$, then $singular(e) \subseteq same(e)$.

(ii) Let e_1 and e_2 be two edges in the same singular set. Then the unordered pair $(same(e_1), same(e_2))$ is not in $DIFF$. ■

We say that a singular edge e is *bound* if $singular(e) \subseteq same(e)$, and *free* otherwise. We can construct a feasible map M' from a reduced map M'' by inserting non-representative singular edges as follows. Let $e = [v, w]$ be a representative singular edge, and let $g(e) = |singular(e)|$. If e is bound, then all the edges in $singular(e)$ must be inserted consecutively in the same side of $e_{v,ref}$ in $M'(v)$. Therefore we replace e in $M''(v)$ by any of the $g(e)!$ permutations of $singular(e)$. If e is free, then the edges in $singular(e)$ can appear in different sides of $e_{v,ref}$ in $M'(v)$ by Lemma 8. Therefore we divide $singular(e)$ into two parts S_1 and S_2, assuming S_1 contains e. Then we replace e by a permutation of S_1, and insert a permutation of S_2 into the other side of $e_{v,ref}$ in $M''(v)$ in the position determined by the condition (2) in the definition of feasible maps. In this case, we have $\frac{(g(e)+1)!}{2}$ different choices.

Now let RS be the set of representative singular edges. For all $x \in RS$, define $h(x) = g(x)!$ if x is bound, and $\frac{(g(x)+1)!}{2}$ otherwise. Then from each reduced map, we can generate $\prod_{x \in RS} h(x)$ different partial maps. By Lemma 1, we have

THEOREM 1. *The total number of planar maps of G is*

$$2^{d+s} \prod_{x \in RS} h(x) \qquad \blacksquare$$

The remaining question is how to compute the two sets *SAME* and *DIFF* efficiently.

3.6. Compute the sets *SAME* and *DIFF*

Now we show how to compute the two sets *SAME* and *DIFF* in linear time during planarity testing. The planarity testing algorithm we will use in this section is a variant of the HT algorithm reported in [4] and is summarized in the next section for the reader's convenience.

3.6.1. Planarity testing

As before, we assume that G is a biconnected graph with more than one edge. Then the tree edges in T form a single tree with only one tree edge leaving the root. Denote this tree edge by e_0. Since $sub(e_0)$ is the whole graph, then we can determine the planarity of G with a procedure that can determine the planarity of $sub(e)$ for all $e \in E$.

We say that an edge e is *planar* if $sub(e)$ is planar. To determine the planarity of an edge e, we consider two cases. If e is a back edge, then $sub(e) = cycle(e)$, which is always planar. Otherwise, e is a tree edge having at least one successor. In this case we first determine the planarity of each of its successors. If all these successors are planar, then we determine the planarity of e based on the structure of its attachments. Following are the details.

3.6.1.1. Structure of attachments

The planarity of an edge $e = [v, w]$ directly depends on the structure of its attachments. If e is planar, we partition the edges of $ATT(e)$ into *block*s as follows. We put two back edges of $ATT(e)$ in the same block if they are on the same side of $cycle(e)$ in every embedding of $sub(e)$. Two blocks *interlace* each other if they are on opposite sides of $cycle(e)$ in every embedding of *sub*(e). By this definition, each block of *ATT*(e) can interlace at most one other block.

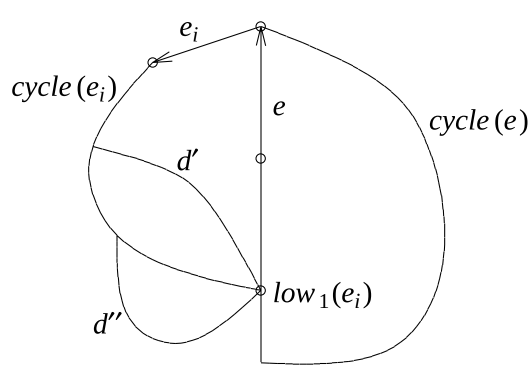
The back edge on *cycle*(e) is the only attachment of e that will not be embedded on either side of *cycle*(e). By convention, this back edge forms a block by itself, called the *neutral block* of e, which does not interlace other blocks of *ATT*(e).

In Fig. 6, *ATT*(e) can be divided into four blocks: B_1 = {[8, 1]}, B_2 = {[12, 1], [14, 2]}, B_3 ={[9, 3]}, and B_4 = {[13, 4]}. B_1 is neutral. B_2 and B_3 are interlacing.

A block of attachments of e is *normal* if it contains some normal attachment of e. Otherwise we say that it is *special*. We say that *sub*(e) is *strongly planar* w.r.t. e if e is planar and if all the normal blocks of *ATT*(e) can be embedded on the same side of *cycle*(e). If *sub*(e) is strongly planar (w.r.t. e), then we say that e is strongly planar. We have,

LEMMA 9. *Let* $e = [v, w] \in T$, *and* e_i *be a successor of* e *such that* $e_i \neq e_{w,ref}$. *Then* e_i *is strongly planar iff the subgraph* $S(e_i) \cup cycle(e)$ *is planar.* ■

Note that in an embedding of $S(e_i) \cup cycle(e)$, the special blocks of e_i do not have to be on the same side of $cycle(e_i)$, see Fig. 8.



The two special attachments d' and d'' of e_i can be on different sides of $cycle(e_i)$, although they are on the same side of $cycle(e)$.

Fig. 8

We represent a block of back edges $K = \{[v_1, w_1], [v_2, w_2], ..., [v_t, w_t]\}$ by a list $L = [w_1, w_2, ..., w_t]$, where $w_1 \leq w_2 \leq ... \leq w_t$. Frequently, we will identify blocks with their list representations. Define $first(K) = first(L) = w_1$, and $last(K) = last(L) = w_t$. If L is empty, we define $first(K) = first(L) = n + 1$, and $last(K) = last(L) = 0$. We can further organize the blocks of *ATT*(e) as follows: if two blocks X and Y interlace, we put them into a pair $[X, Y]$, assuming $last(X) \geq last(Y)$; if a nonempty block X does not interlace any other block, we form a pair $[X, [\,]]$.

Let $[X_1, Y_1]$ and $[X_2, Y_2]$ be two pairs of interlacing blocks. We say $[X_1, Y_1] \leq [X_2, Y_2]$ iff $last(X_1) \leq min(first(X_2), first(Y_2))$. We say a list of interlacing pairs $[q_1,...,q_s]$ is *well-ordered* if $q_1 \leq \cdots \leq q_s$. Empty lists or lists of one pair are well-ordered by convention. In [4] we proved that all the interlacing pairs of $ATT(e)$ can be organized into a well-ordered list $[p_1, ..., p_t]$. We call this list $att(e)$.

In Fig. 6, $att(e) = [p_1, p_2, p_3]$, where p_1 = [[1], [ ]], p_2 = [[3], [1, 2]], and p_3 = [[4], [ ]].

3.6.1.2. Compute *att*(*e*)

Now we are ready to compute $att(e)$. The planarity of e will be decided at the same time.

Consider an edge $e = [v, w] \in E$. If e is a back edge, then its only attachment is e itself. Therefore $att(e) = [[[w], [\ ]]]$. Otherwise, let $\Phi(w) = [e_1, ..., e_k]$. We first recursively compute $att(e_i)$ for each e_i in $\Phi(w)$, then compute $att(e)$ in four steps:

Algorithm A

Step 1 For $i = 1, ..., k$, delete all occurrences of w appearing in blocks within $att(e_i)$. Because these occurrences appear together at the end of the blocks that are contained in the last pairs of $att(e_i)$ only, a simple list traversal suffices to delete all these occurrences in time $O(1 + \textit{number of deletions})$. After this, initialize $att(e)$ to be $att(e_1)$.

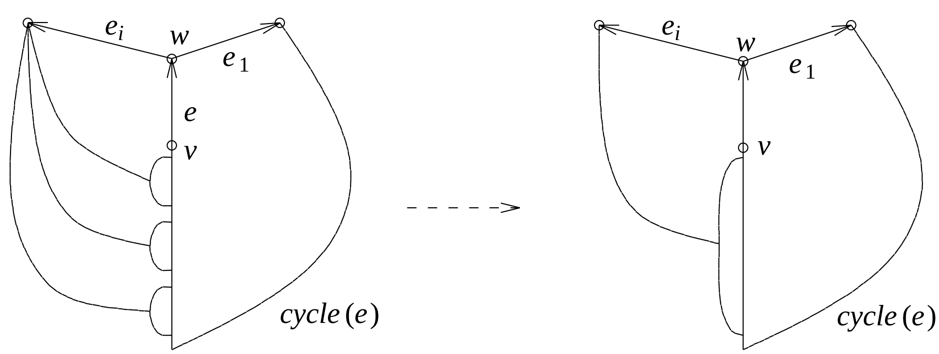


Fig. 9

Step 2. For $i = 2, ..., k$, merge all the blocks of $att(e_i)$ into one intermediate block B_i. See Fig. 9.

According to Lemma 9, this step can be done only if the normal blocks of $att(e_i)$ do not interlace. (If they interlace, the graph is not planar, and the computation fails.) To merge a series of blocks, simply concatenate their ordered list representations (such concatenation is order preserving).

Step 3. Merge blocks in $att(e)$. See Fig. 10.

By Observation 3, all blocks D in $att(e)$ with $last(D) > low_1(e_2)$ must be merged into one block B_1. (If any two of these blocks interlace, the graph is not planar, and the computation fails.)

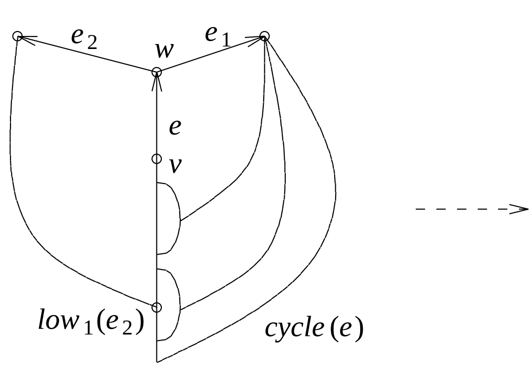


Fig. 10

This is achieved by merging from the high end of *att* (*e*). This step turns *att* (*e*) into a list of pairs $p_1 \le \cdots \le p_h$ with only p_h possibly having a block *D* with $last(D) > low_1(e_2)$.

Step 4. For $i = 2, ..., k$, add blocks B_i into *att* (*e*).

To process B_i, consider the last pair P : [*X*, *Y*] of *att* (*e*). Consider three cases:

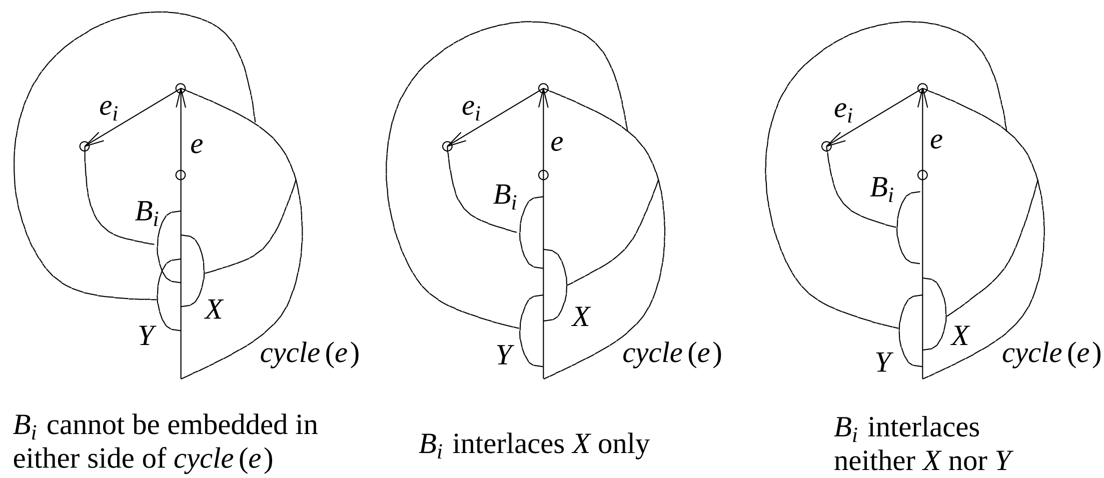


Fig. 11

i. If B_i cannot be embedded on either side of *cycle* (*e*), then *G* is not planar, and the computation of *att* (*e*) fails.

ii. If B_i interlaces *X* only, then merge B_i into *Y*. Next, switch *X* and *Y* if $last(X) < last(Y)$.

iii. If B_i interlaces neither *X* nor *Y*, then add $[B_i, [\,]]$ to the high end of *att* (*e*); $P := [B_i, [\,]]$.

By the following lemma, testing whether B_i interlaces *X* or *Y* takes $O(1)$ time. Also by that lemma, it is not possible that B_i interlaces *Y* only, since $last(X) \ge last(Y)$ (see Fig. 11).

LEMMA 10. *B_i and D can be embedded on the same side of cycle* (*e*) *iff* $low_1(e_i) \ge last(D)$, *where D = X or D = Y.* ■

In [4] we proved that

THEOREM 2.

1. Algorithm A computes $att(e)$ successfully iff e is planar.

2. If e is planar, then Algorithm A computes $att(e)$ correctly. ■

3.6.2. Compute the sets *SAME* and *DIFF*

Next we augment Algorithm A so as to compute the two sets *SAME* and *DIFF* during the planarity testing.

Let $e \in E$ be an edge of G. Let e_a an attachment of e not on $cycle(e)$. Then $root(e_a)$ and e_a are embedded on the same side of $cycle(e)$ in any embedding of G. Thus, for each non-neutral block X of e, there is a unique group in *SAME* that contains the roots of the attachments in X. We call this group $buddy(X)$. It is easy to see that if $[X, Y]$ is a pair of nonempty interlacing blocks of $ATT(e)$, then $(buddy(X), buddy(Y))$ is a pair in *DIFF*. Furthermore, in the proof of Lemma 6, we notice that if e_a is normal, and if $e \in E_0$, then $root(e_a)$ and e belong to the same group in *SAME*. Thus, if X is a normal block of e, and $e \in E_0$, then $buddy(X)$ also contains e. For convenience, we further extend the definition of *buddy* as follows. If $[X, Y]$ is a pair in $ATT(e)$ such that $Y = [\ ]$, and $(buddy(X), U) \in DIFF$, then define $buddy(Y) = U$. According to these observations , we can compute the two sets *SAME* and *DIFF* with the following enhancement to Algorithm A.

Enhancement B

1. Initialization. For all $e \in B$, let $buddy([e]) = \emptyset$. Let $SAME = \{ \{e\} : e \in E_0\}$ and $DIFF = \{(\{e\}, \emptyset) : e \in E_0\}$;

2. In step 2 of Algorithm A, for $i = 2, ..., k$, before we merge $att(e_i)$, we initialize $buddy(B_i)$ to be $\{e_i\}$. For each pair $[X, Y]$ or $[Y, X]$ in $att(e_i)$ such that X is normal w.r.t. e_i, let U be the set such that $(buddy(B_i), U) \in DIFF$; in *SAME*, merge $buddy(X)$ into $buddy(B_i)$ and merge $buddy(Y)$ into U; in *DIFF*, merge the two pairs $(buddy(B_i), U)$ and $(buddy(X), buddy(Y))$ into one pair $(buddy(B_i) \cup buddy(X), U \cup buddy(Y))$.

3. In step 3, let $[X, Y]$ be the last pair in the list $att(e_1)$ before merging. For each pair $[X_1, Y_1]$ in $att(e_1)$ merged into $[X, Y]$, do the following: in *SAME*, merge $buddy(X_1)$ into $buddy(X)$ and merge $buddy(Y_1)$ into $buddy(Y)$; in *DIFF*, merge the two pairs $(buddy(X), buddy(Y))$ and $(buddy(X_1), buddy(Y_1))$ into one pair $(buddy(X) \cup buddy(X_1), buddy(Y) \cup buddy(Y_1))$.

4. In step 4, for $i = 2, ..., k$, let U be the set such that $(buddy(B_i), U) \in DIFF$. If $[B_i, Z]$ becomes the top pair of $att(e)$, where $Z = [\ ]$, then let $buddy(Z) = U$. If B_i is merged into Y, then in *SAME*, merge $buddy(B_i)$ into $buddy(Y)$, and merge U into $buddy(X)$; in *DIFF*, merge the two pairs $(buddy(B_i), U)$ and $(buddy(X), buddy(Y))$ into one pair $(U \cup buddy(X), buddy(B_i) \cup buddy(Y))$.

■

One way to prove the correctness of Enhancement B is to prove that,

(i) if an ordered partition $P = [LL, RR]$ of E_0 is valid, then it is consistent with the two sets *SAME* and *DIFF* computed by Enhancement B; and

(ii) if an ordered partition $P = [LL, RR]$ of E_0 is consistent with the two sets *SAME* and *DIFF* computed by Enhancement B, then it is valid.

We see that (i) is true because in the enhancement code, two edges are put in the same group of *SAME* only if they have the same color in each embedding of G, and two groups form a pair in *DIFF* only if they always have different colors. The assertion (ii) is basically the same as Lemma 7, except that the two sets *SAME* and *DIFF* here are computed by Enhancement B, not given by their definitions. Since the proof of Lemma 7 is based on Lemma 5 and Lemma 6, then we need only to prove these two lemmas under the new condition.

LEMMA 11. *Lemma 5 remains true if the two sets SAME and DIFF are computed by Enhancement B.*

Proof Consider the attachment $[x, y]$ given in Lemma 5. Let $[a, b] = root([x, y])$. Let $[X, Y]$ be the top pair of blocks in $att(e)$ in Step 4 of the planarity testing. Since $att(e)$ is well-ordered, then $[x, y]$ is contained in either X or Y. If $[x, y] \in Y$, then $low_1(e_i) < last(Y)$, and G is not planar. Thus $[x, y] \in X$, and $low_1(e_i) < last(X)$. Therefore B_i is merged into Y in Step 4. Then $root([x, y]) \in buddy(X)$, $e_i \in buddy(Y)$, and $(buddy(X), buddy(Y)) \in DIFF$. ■

LEMMA 12. *Lemma 6 remains true if the two sets SAME and DIFF are computed by the Enhancement B.*

Proof Consider the edge e, the embedding H_e and the partition Q given in Lemma 6. Assume wlog that $e \in LL$. Let $[x, y]$ be a normal attachment of e. We need to show that $[x, y]$ is embedded on the left hand side of $cycle(e)$ in H_e. Let $[a, b] = root([x, y])$. Let e' be the predecessor of e. Let X be the block of attachment in $ATT(e')$ that contains $[x, y]$. Then Enhancement B will put both e and $[a, b]$ into $buddy(X)$. This means that e and $[a, b]$ are in the same group of *SAME*. Since we assume $e \in LL$, then $[a, b] \in LL$. Since H_e is conformable to Q, then $[a, b]$, and therefore $[x, y]$, are embedded on the left hand side of $cycle(e)$ in H_e. ■

As a result of Lemma 11 and Lemma 12, Lemma 7 remains true for the two sets *SAME* and *DIFF* computed by our Enhancement B. Therefore we have,

THEOREM 3. *If G is planar, then Algorithm A with Enhancement B computes the sets SAME and DIFF correctly.* ■

4. Number of embeddings for connected components

Next consider a connected graph G with several biconnected components. Suppose we know the number of embeddings of each biconnected component. We discuss how to find the total number of embeddings of G. This problem was previously considered by Stallmann [12], but his solution is complicated and not efficient. In this section we will give a simple closed formula for this problem that is computable in $O(n)$ arithmetic steps.

We start with the simple situation that G has two biconnected components G_1 and G_2 sharing an articulation point a. Suppose there are m_1 edges connected to a in G_1 and m_2 such edges in G_2. Let H_1 be an embedding of G_1 on a sphere S_1, and H_2 be an embedding of G_2 on another sphere S_2. Imagine that S_1 and S_2 are balloons. To combine H_1 and H_2 into a sphere embedding of G, we choose a face F_1 of H_1 and a face F_2 of H_2 such that their boundaries contain a. Make a hole on F_1 so that a is the only point shared by the boundaries of the hole and F_1. Do the same thing with F_2. Glue these two holes on their boundaries, making sure that the two embeddings of a are put together. Blowing the combined balloon into a sphere gives an embedding of G. There are m_1 faces in H_1 whose boundaries contain a, and there are m_2 such faces in H_2. Thus we can get $m_1 m_2$ different sphere embeddings of G by combining H_1 and H_2.

The above method of combining sphere embeddings can be generalized to get sphere embeddings of graphs with more biconnected components and more articulation points. But counting the number of embeddings becomes more complicated in the general case. For graphs with one articulation point, we have the following result:

LEMMA 13. *Let G be a planar graph consisting of j biconnected components $G_1,...,G_j$ sharing an articulation point a. For each $i = 1,...,j$, let m_i be the number of edges connected to a in G_i, and k_i be the number of different sphere embeddings of G_i. Then for $j > 2$, the total number of different sphere embeddings of G is:*

$$k_1 k_2 \cdots k_j m_1 m_2 \cdots m_j (A-1)(A-2) \cdots (A-j+2)$$

where $A = m_1 + \cdots + m_j$.

Proof We need only to prove the following assertion: for a fixed group of embeddings H_1, ..., H_j of $G_1,...,G_j$, we can obtain $m_1 m_2 \cdots m_j (A-1)(A-2) \cdots (A-j+2)$ different embeddings of G by gluing balloons. We call this set of embeddings of G an $E_{m_1,...,m_j}$ set.

We prove the assertion by induction on A. The basis is trivial, when $m_1 = ... = m_j = 1$. Now we assume that the assertion is true for any $A < k$, where $k > j$. Consider the case when $A = k$. Then there exists some $i = 1, ..., j$ such that $m_i > 1$. We assume wlog that $m_1 > 1$. For each $i = 1$, ..., j, let $e_{i,1},...,e_{i,m_j}$ be the clockwise sequence of edges around a in H_i. We divide an $E_{m_1,...,m_j}$ set into j groups:

Group 1 contains all the embeddings such that $e_{1,1}$ is followed by $e_{1,2}$;

Group 2 contains all the embeddings such that $e_{1,1}$ is followed by $e_{2,l}$, where $l = 1,...,m_2$;

. . .

Group j contains all the embeddings such that $e_{1,1}$ is followed by $e_{j,l}$, where $l = 1,...,m_j$.

In Group 1, if we glue the two edges $e_{1,1}$ and $e_{1,2}$ together in each embedding, we get an $E_{m_1-1,m_2,...,m_j}$ set. By the induction hypothesis, the size of Group 1 is

$$(m_1-1)m_2 \cdots m_j(A-2) \cdots (A-j+1)$$

For each i = 2, ..., j, we divide Group i into m_i subgroups, so that in every embedding of the l-th subgroup, $e_{1,1}$ is followed by $e_{i,l}$. By gluing the two edges $e_{1,1}$ and $e_{i,l}$ together in each of the embedding in the l-th subgroup, we get an $E_{m_1+m_i-1,m_2,...,m_{i-1},m_{i+1},...,m_j}$ set, which has the size

$$(m_1+m_i-1)m_2 \cdots m_{i-1}m_{i+1} \cdots m_j(A-2) \cdots (A-j+2)$$

Therefore the size of Group i is

$$m_i[(m_1+m_i-1)m_2 \cdots m_{i-1}m_{i+1} \cdots m_j(A-2) \cdots (A-j+2)]$$
$$= (m_1+m_i-1)m_2 \cdots m_j(A-2) \cdots (A-j+2)$$

Adding the sizes of Group 1, ..., Group j, we see that the size of $E_{m_1,m_2,...,m_j}$ is

$$m_1m_2 \cdots m_j(A-1) \cdots (A-j+2) \qquad \blacksquare$$

Now consider a connected graph G with more than one articulation point. To count the number of embeddings, we first choose one articulation point a. Let G_a be the subgraph of G that consists of all the biconnected components sharing a. Using Lemma 13, we can count the number of embeddings of the subgraph G_a. Then we treat G_a as one biconnected component, and solve the remaining problem recursively. The result is summarized in the following theorem:

THEOREM 4. *Let G be a planar graph. Let* Γ *be the set of biconnected components of G, let* Θ *be the set of articulation points of G. For each biconnected component C in* Γ*, let* k_C *be the number of sphere embeddings of C. For each* $a \in \Theta$*, let* Γ_a *be the set of biconnected components of G sharing a, and let* A_a *be the number of edges connected to a. For each* $a \in \Theta$ *and each component* $C \in \Gamma_a$*, let* $m_{C,a}$ *be the number of edges in C connected to a. Then the total number of sphere embeddings of G is*

$$\prod_{C\in\Gamma} k_C \prod_{a\in\Theta} \Big(\prod_{C\in\Gamma_a} m_{C,a} \prod_{i=1}^{|\Gamma_a|-2} (A_a-i) \Big) \qquad \blacksquare$$

The analysis in this section also suggests a recursive procedure that generates all planar maps of G without repetition from the planar maps of the biconnected components of G.

5. Counting Embeddings for Unconnected Graphs

Finally, we consider how to count the embeddings of graphs having several connected components, given the number of embedding of each of the connected components.

THEOREM 5. *Let G be a planar graph consists of c connected components* $C_1, ..., C_c$, *where* $c > 1$. *If for* $i = 1, ..., c$, C_i *has* n_i *sphere embeddings each having* f_i *faces, then the number of sphere embeddings of G is*

$$(1+\sum_{i=1}^{c}(f_i-1))^{c-2}\prod_{i=1}^{c}n_i f_i$$

Proof For each $i = 1, ..., c$, we choose a fixed embedding H_i of C_i. We denote the set of these embeddings by Δ. We call the embeddings in Δ *subembeddings* in order to distinguish them from the embeddings of G. Very similarly to the description in Section 4, we can combine the subembeddings in Δ into an embedding of G by gluing balloons. The main difference is that in this case the holes made should not touch the boundary of any face. Let Ψ be the set of all embeddings of G that can be obtained from Δ this way. We need to prove that

$$|\Psi| = (1+\sum_{i=1}^{c}(f_i-1))^{c-2}\prod_{i=1}^{c}f_i \qquad (*)$$

We prove the claim by induction on c, the total number of connected components of G. For $c = 2$, the claim is obviously true. Then we assume that the claim is true for all $c < k$, where $k > 2$. We want to show that the claim is also true for $c = k$. We partition Ψ into c-1 groups $\Psi_1, ..., \Psi_{c-1}$, such that for $i = 1, ..., c-1$, group Ψ_i contains the embeddings H of G in which H_1 is the neighbor of exactly i other subembeddings in Δ (recall that two subembeddings H_s and H_t are neighbors of each other in H if there is a face in H whose boundary contains edges from both H_s and H_t.) We further divide Ψ_i into $\binom{c-1}{i}$ subgroups such that in all embeddings of each subgroup, H_1 has the same set of neighbors. Consider one such subgroup $\Psi_{i,P}$ in which H_1 has the set of neighbors $P = \{H_{t_1},...,H_{t_i}\}$. Let $Q = \{H_2,...,H_c\} - P$. An embedding in $\Psi_{i,P}$ can be obtained in two stages. First we combine H_1 and all the subembeddings in P into one embedding X. Since for each $j = t_1, ..., t_i$, each of the f_j faces of H_j can be adjacent to each of the f_1 faces of H_1, then we have the number c_1 of different choices in the first stage is $(f_1 f_{t_1})...(f_1 f_{t_i})$. Next we combine X and the subembeddings in Q into an embedding Y in $\Psi_{i,P}$. Since subembeddings in Q are not neighbors of H_1, then we can treat X as a component with $\sum_{H_s \in P}(f_s-1)$ faces. Applying (*) inductively, we find that the number c_2 of different choices in the second stage is

$$(1+(\sum_{H_s\in P}(f_s-1)-1)+\sum_{H_s\in Q}(f_s-1))^{c-i-2}\sum_{H_s\in P}(f_s-1)\prod_{H_s\in Q}f_s$$

$$=(\sum_{j=2}^{c}(f_s-1))^{c-i-2}\sum_{H_s\in P}(f_s-1)\prod_{H_s\in Q}f_s$$

Thus the size of subgroup $\Psi_{i,P}$ is

$$c_1 c_2$$

$$= (\sum_{j=2}^{c}(f_s-1))^{c-i-2} \sum_{H_s \in P}(f_s-1) \prod_{H_s \in Q} f_s \prod_{H_s \in P}(f_1 f_s)$$

$$= \sum_{H_s \in P}(f_s-1) (\sum_{j=2}^{c}(f_j-1))^{c-i-2} f_1^{i-1} \prod_{j=1}^{c} f_j$$

Therefor the size of Ψ_i is

$$\sum_{\substack{P \subseteq \{H_2, \ldots, H_c\} \\ |P|=i}} |\Psi_{i,P}|$$

$$= \sum_{\substack{P \subseteq \{H_2, \ldots, H_c\} \\ |P|=i}} (\sum_{H_s \in P}(f_s-1) (\sum_{j=2}^{c}(f_j-1))^{c-i-2} f_1^{i-1} \prod_{j=1}^{c} f_j)$$

$$= (\sum_{\substack{P \subseteq \{H_2, \ldots, H_c\} \\ |P|=i}} \sum_{H_s \in P}(f_s-1)) (\sum_{j=2}^{c}(f_j-1))^{c-i-2} f_1^{i-1} \prod_{j=1}^{c} f_j$$

$$= \binom{c-2}{i-1} \sum_{j=2}^{c}(f_j-1) (\sum_{j=2}^{c}(f_j-1))^{c-i-2} f_1^{i-1} \prod_{j=1}^{c} f_j$$

$$= \binom{c-2}{i-1} (\sum_{j=2}^{c}(f_j-1))^{c-i-1} f_1^{i-1} \prod_{j=1}^{c} f_j$$

Finally, the size of C is

$$\sum_{i=1}^{c-1} |\Psi_i|$$

$$= \sum_{i=1}^{c-1} (\binom{c-2}{i-1} (\sum_{j=2}^{c}(f_j-1))^{c-i-1} f_1^{i-1} \prod_{j=1}^{c} f_j)$$

$$= \sum_{i=0}^{c-2} (\binom{c-2}{i} (\sum_{j=2}^{c}(f_j-1))^{c-i-2} f_1^{i}) \prod_{j=1}^{c} f_j$$

$$= (1 + \sum_{j=1}^{c}(f_j-1))^{c-2} \prod_{j=1}^{c} f_j \qquad \blacksquare$$

From the above discussion, it is not difficult to give a recursive procedure that generates all the adjacency relations on the set of faces of the subembeddings in Δ.

Acknowledgment I thank Tarjan for introducing me to the counting problems solved in this paper. I also thank Matthias Stallmann for pointing out that my previous formula for counting embeddings of connected graphs with two biconnected components does not easily extend to the general case. That comment inspired my discovery of Lemma 13.

References

1. Aho, A., Hopcroft, J., and Ullman, J., *Design and Analysis of Computer Algorithms,* Addison-Wesley, 1974.

2. Berge, C., *The Theory of Graphs and its Applications*, Methuem, London, 1964. trans. by Alision Doig

3. Booth, K. S. and Lueker, G. S., ‘‘Testing for the Consecutive Ones Property, Interval Graphs, and Graph Planarity Using PQ-tree Algorithms,’’ *Journal of Computer and System Science*, vol. 13, no. 3, pp. 335-379, 1976.

4. Cai, J., Han, X., and Tarjan, R. E., *An* $m \log n$ *Algorithm for the Maximal Planar Subgraph Problem*, Department of Computer Science, Princeton University. Technical Report

5. Chiba, N., Nishizeki, T., Abe, S., and Ozawa, T., ‘‘A Linear Algorithm for Embedding Planar Graphs Using *PQ*-trees,’’ *Journal of Computer and System Sciences*, vol. 30, no. 1, pp. 54-76, 1985.

6. Di Battista, G. and Tamassia, R., ‘‘Incremental Planarity Testing (Extended Abstract),’’ in *Proc. 30th Anual I.E.E.E. Symposium on Foundations of Computer Science*, pp. 436-441, 1989.

7. Hall, D. and Spencer, G., *Elementary Topology,* Wiley, New York, 1955.

8. Hopcroft, J. and Tarjan, R., ‘‘Efficient Planarity Testing,’’ *JACM*, vol. 21, no. 4, pp. 549-568, October, 1974.

9. Jayakumar, R., Thulasiraman, K., and Swamy, M. N. S., ‘‘$O(n^2)$ Algorithms for Graph Planarization,’’ *IEEE Transactions on CAD*, vol. 8, no. 3, pp. 257-267, March, 1989.

10. Lempel, A., Even, S, and Cederbaun, I., ‘‘An Algorithm for Planarity Testing of Graphs,’’ in *Theory of Graphs, International Symposium*, pp. 215-232, Rome, July, 1966.

11. Stallman, M., *Using PQ-trees for Planar Embedding Problems,* North Carolina State University, December 1985. Technical Report

12. Stallman, M., *Enumerating the Embeddings of a PLanar Graph,* North Carolina State University, March, 1989. Preliminary Draft

13. Thron, W.T., *Introduction to the Theory of Functions of a Complex Variable.,* Wiley, New York, 1953.

14. Wu, W., ‘‘On the Planar Imbedding of Linear Graphs,’’ *J. Sys. Sci. & Math. Scis.*, vol. 5, no. 4, pp. 290-302, Institute of Systems Science, Academia Sinica, Beijing, 1985.