JOURNAL OF ALGORITHMS 10, 187–211 (1989)

The Hamiltonian Cycle Problem Is Linear-Time Solvable for 4-Connected Planar Graphs

NORISHIGE CHIBA

Department of Computer Science, Faculty of Engineering, Iwate University, Morioka 020, Japan

AND

TAKAO NISHIZEKI

Department of Electrical Communications, Faculty of Engineering, Tohoku University, Sendai 980, Japan

Received October 18, 1986; accepted April 6, 1988

An algorithm is presented for finding a Hamiltonian cycle in 4-connected planar graphs. The algorithm uses linear time and storage space, while the previously best one given by Gouyou-Beauchamps uses $O(n^3)$ time and space, where n is the number of vertices in a graph.

1. INTRODUCTION

A Hamiltonian cycle (path) of a graph G is a simple cycle (path) which contains all the vertices of G. The Hamiltonian cycle problem asks whether a given graph contains a Hamiltonian cycle. It is NP-complete even for 3-connected planar graphs [3, 6]. However, the problem becomes polynomial-time solvable for 4-connected planar graphs: Tutte proved that such a graph necessarily contains a Hamiltonian cycle [9, 10]; and, moreover, Gouyou-Beauchamps [4], based on Tutte's proof, gave an $O(n^3)$ algorithm which actually finds a Hamiltonian cycle in such a graph. Throughout the paper n denotes the number of vertices in a graph.

In this paper we give a linear algorithm for finding a Hamiltonian cycle in 4-connected planar graphs. This linear algorithm improves Gouyou-Beauchamps' $O(n^3)$ and Asano, Kikuchi, and Saito's linear algorithms [1]; the last works only for 4-connected maximal planar graphs.

0196-6774/89 $3.00

Gouyou-Beauchamps' algorithm, as well as ours, used a "divide-and-conquer" method: they first decompose a graph into several small subgraphs, then recursively solve the subproblems, and finally combine the subsolutions into a solution for the whole graph. Since the decomposed subgraphs might not be (edge-)disjoint, it was nontrivial to verify even the polynomial boundedness of the algorithm. Indeed he needed a lengthy argument to prove the $O(n^3)$ bound. In contrast, Thomassen somehow remarks in [8] that a polynomial bounded algorithm can easily be extracted from his short proof for Tutte's theorem. This remark seems to be misled from a flaw in his proof. The proof misses an argument on an "ε-bridge" in Tutte's proof which produces nondisjoint subgraphs [7, 4]. Although his proof can be completed, the proof as well as Tutte's suffers from the same algorithmic difficulty, which Gouyou-Beauchamps struggled to resolve [2].

In this paper we first present our version of a proof for Tutte's theorem, based on Thomassen's proof but avoiding the decomposition into nondisjoint subgraphs. The proof is constructive and yields a simple algorithm for our purpose. It is rather straightforward to show that at most $O(n)$ recursive calls occur during one execution of the algorithm. To the contrast $O(n^2)$ recursive calls occur in Gouyou-Beauchamp's algorithm. Thus our algorithm clearly runs in $O(n^2)$ time, since one step of "divide-and-conquer" can be done in $O(n)$ time. Furthermore, we show that, using a sophisticated method to decompose a graph, one can implement the algorithm to run in linear time.

2. Proof

In this section we define some terms and present our version of a proof for Tutte's theorem.

We first define some terms, related to a graph decomposition, which are variants of those in [5]. Let $G = (V, E)$ be a 2-connected simple undirected graph with vertex set V and edge set E. A pair $\{x, y\}$ of vertices x and y is a *separation pair* if G contains two subgraphs $G_1' = (V_1, E_1')$ and $G_2' = (V_2, E_2')$ satisfying the following conditions:

(a) $V = V_1 \cup V_2$, $V_1 \cap V_2 = \{x, y\}$;

(b) $E = E_1' \cup E_2'$, $E_1' \cap E_2' = \varnothing$,

$$|E_1'| \geq 2, \qquad |E_2'| \geq 2.$$

A graph G is *3-connected* if G is 2-connected and has no separation pair. For a separation pair $\{x, y\}$, $G_1 = (V_1, E_1' \cup \{(x, y)\})$ and $G_2 = (V_2, E_2' \cup \{(x, y)\})$ are called *split graphs* of G, where (x, y) denotes an edge joining x and y. We sometimes say "split G_1 from G" when constructing G_2 from G. Note that in our definition no multiple edges are produced in G_1 or G_2. The edge (x, y) in G_1 or G_2 is called a *virtual edge* no matter whether it originally exists in G or is newly added to G_1 or G_2. Dividing a graph G into two split graphs G_1 and G_2 is called *splitting*. Reassembling the two split graphs G_1 and G_2 into G is called *merging*. Merging is the inverse of splitting.

Analogously we define a separation triple as follows. A set $\{x, y, z\}$ of three vertices is a *separation triple* of G if G has two subgraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ satisfying the following conditions:

(a) $V = V_1 \cup V_2$, $V_1 \cap V_2 = \{x, y, z\}$,

$|V_1| \geq 4$, $|V_2| \geq 4$;

(b) $E = E_1 \cup E_2$, $E_1 \cap E_2 = \varnothing$.

A graph G is *4-connected* if G is 3-connected and has no separation triples. The terms "3-connected" and "4-connected" are equivalent to "vertically 3-connected" and "vertically 4-connected" in [9, 10], respectively.

If G' is a subgraph of G, then $G - G'$ denotes the graph obtained from G by deleting all the vertices in G' together with all the edges incident to them. If V' is the vertex set of G', $G - G'$ is denoted by $G - V'$.

We are now ready to present Thomassen's results.

LEMMA 1 [Thomassen]. *Let G be a 2-connected plane graph with the outer facial cycle Z. Let s and $e = (a, b)$ be a vertex and an edge, both on Z, and let t be any vertex of G distinct from s. Then G has a path P going from s to t through e such that*

(i) *each component of $G - P$ is adjacent to at most three vertices of P, and*

(ii) *each component of $G - P$ is adjacent to at most two vertices of P if it contains a vertex of Z.*

Clearly Lemma 1 implies that a 4-connected planar graph G has a Hamiltonian cycle: let s and t be two adjacent vertices on Z and let $e \neq (s, t)$ be an edge on Z, then the path P joining s and t through e, assurred by Lemma 1, must be a Hamiltonian path of G, so $P + (s, t)$ must be a Hamiltonian cycle of G. Thus an algorithm for finding the s-t path P immediately yields an algorithm for finding a Hamiltonian cycle. In this paper we give such a *linear* algorithm for finding the s-t path P in a 4-connected plane graph.

Although the original graph G is 4-connected, the subgraphs into which G is decomposed by our algorithm are no longer 4-connected. However, they inherit some favorable property, which we call "internally 4-connected." Intuitively a graph is internally 4-connected if it contains no separation pair or triple in the interior. We define formally the term as follows.

Let G be a 2-connected plane graph with outer facial cycle Z. Let s and t be two distinct vertices on Z, and let $e = (a, b)$ be an arbitrary edge on Z such that $e \neq (s, t)$. Interchanging the roles of s and t or a and b and also mirroring the plane embedding of G if necessary, one may assume without loss of generality that $t \neq a, b$ and vertices s, a, b, and t appear clockwise on Z in this order. (See Fig. 1(a).) Note that possibly $s = a$ as shown in Fig. 1(b). Let r be the vertex on Z counterclockwise next to s, and let $f = (r, s)$ be the edge joining r and s. If vertices x and y are on Z, P_{xy} denotes the "outer" path going from x to y clockwise on Z. Then a plane 2-connected graph G is *internally 4-connected with respect to* (s, t, e) if G satisfies the two conditions:

(a) If $\{x, y\}$ is a separation pair, then (i) every component of $G - \{x, y\}$ contains at least one vertex of Z and (ii) each of the three paths P_{sa}, P_{bt}, and P_{ts} contains at most one of x and y.

(b) If $\{x, y, z\}$ is a separation triple, then every component of $G - \{x, y, z\}$ contains at least one vertex of Z.

Condition (a) (i) above implies that both x and y of the separation pair must lie on Z and that $G - \{x, y\}$ contains exactly two components, while condition (b) implies that at least two of x, y, and z of the separation triple must lie on Z. Figs. 1(c), (d), and (e) show separation pairs violating condition (a), while Fig. 1(f) shows a separation triple violating condition (b). It should be noted that the property of internal 4-connectivity depends on the planar embedding of G as well as the choice of s, t, and e.

An internally 4-connected graph is diagramatically illustrated in Fig. 2(a). A separation pair $\{x, y\}$ is called *vertical* if either $x \in P_{sa} - s$ and $y \in P_{br}$ or $x = s$ and $y \in P_{bt} - t$. In Fig. 2(a) $\{x, y\}$ is one of the vertical separation pairs. In the algorithm, G is decomposed into two subgraphs with respect to a vertical separation pair. (Such a decomposition is called a Type I reduction for the time being, and will be formally defined later.) If $\{x, y\}$ is a non-vertical separation pair, one of x and y must be in $P_{bt} - t$ and the other in $P_{tr} - t$, and such a pair is implicitly said to be *horizontal*. In Fig. 2(a) $\{y', z'\}$ is a horizontal separation pair.

Let G be an internally 4-connected plane graph having no vertical separation pair. (Figure 3(a) illustrates such a graph G with $s \neq a$, while Fig. 3(b) illustrates G with $s = a$.) Then G is decomposed into several types of subgraphs, called G_b, G_g^2, G_u^3, and G_g^4, which are defined below. (If $s = a$, G is decomposed into G_b.)

Let C_b be the block (i.e., 2-connected component) of $G - P_{sa}$ which contains t. (In Figs. 3(a) and (b) C_b are cross-hatched.) Then C_b must

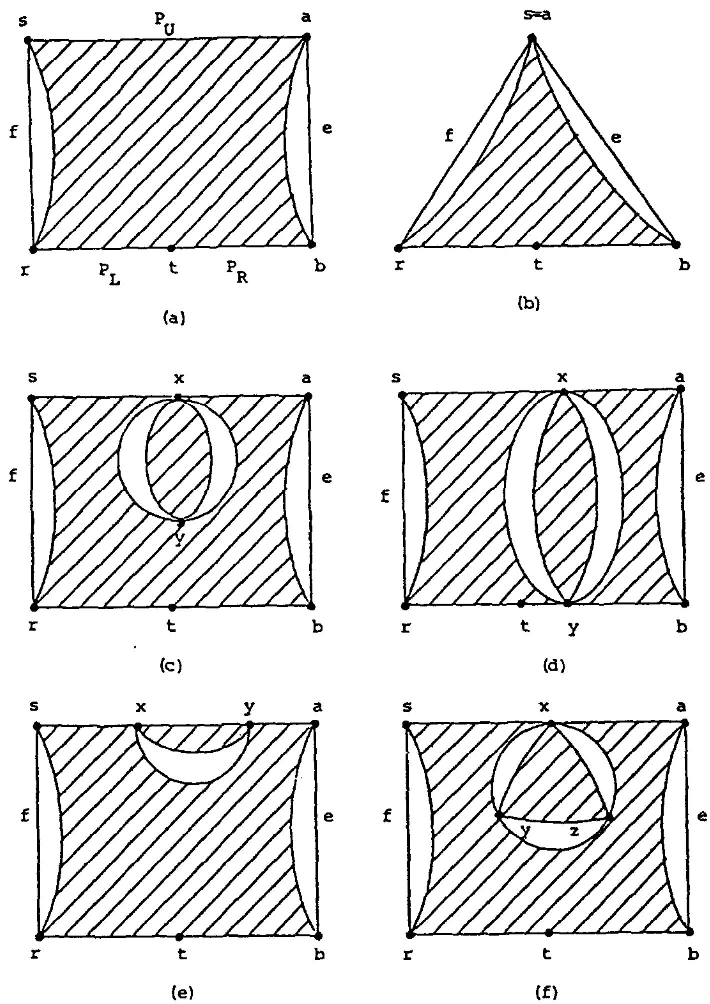

FIG. 1. Illustrations for notations and conditions (a) and (b): (a) shows G with $s \neq a$; (b) shows G with $s = a$; (c), (d), and (e) show separation pairs violating conditions (a); and (f) shows a separation triple $\{x, y, z\}$ violating condition (b).

entirely contain P_{br}, otherwise G would contain a vertical separation pair. Repeat splitting of C_b at every separation pair such that one of the two split graphs entirely contains P_{br}. (In case of Fig. 3(a), C_b is split into four components, as illustrated in Fig. 3(c).) The resulting components are called G_b or G_g^2. That is, the component containing P_{br} is called G_b, while each of the others is called G_g^2, where $g = (x, y)$ is a virtual edge contained in the

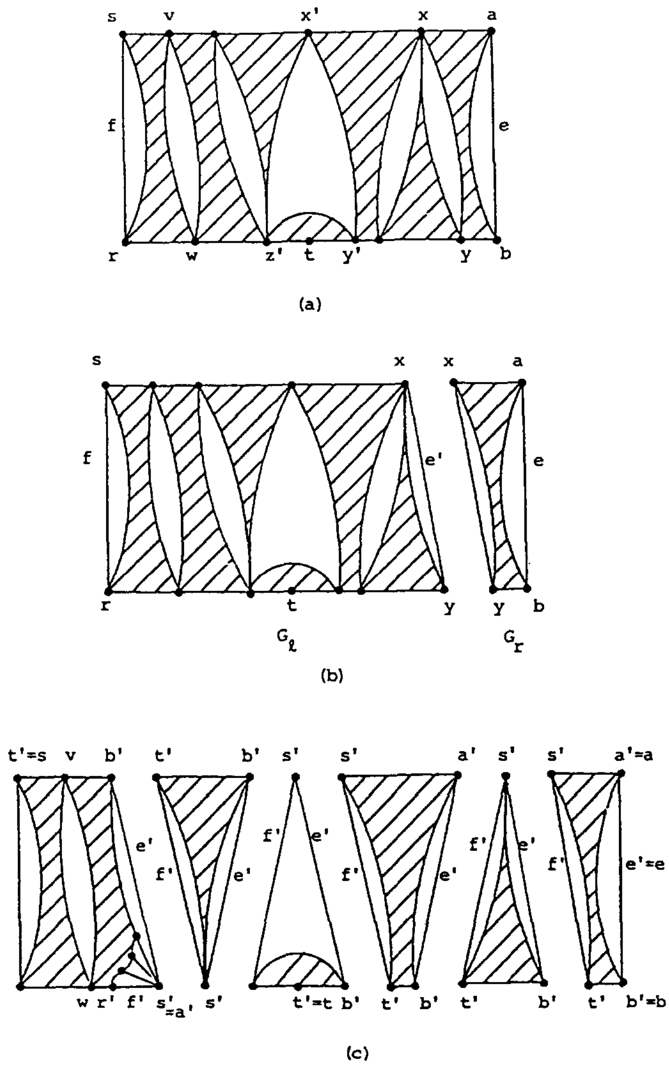

FIG. 2. Type I reduction for an internally 4-connected graph G having vertical separation pairs: (a) G; (b) G_l and G_r; (c) the split graphs.

component. Here we may assume by possibly interchanging the roles of x and y that $x \neq b$.

We next define G_u^3 for each cut vertex u of $G - P_{sa}$ contained in C_b. (There are six u's in Fig. 3(a).) Let C_u be the maximal subgraph of $G - P_{sa}$ which can be separated from C_b at u. Let C_u^3 be the subgraph of G induced by the vertices of C_u and the vertices on P_{sa} which are adjacent to C_u. Note

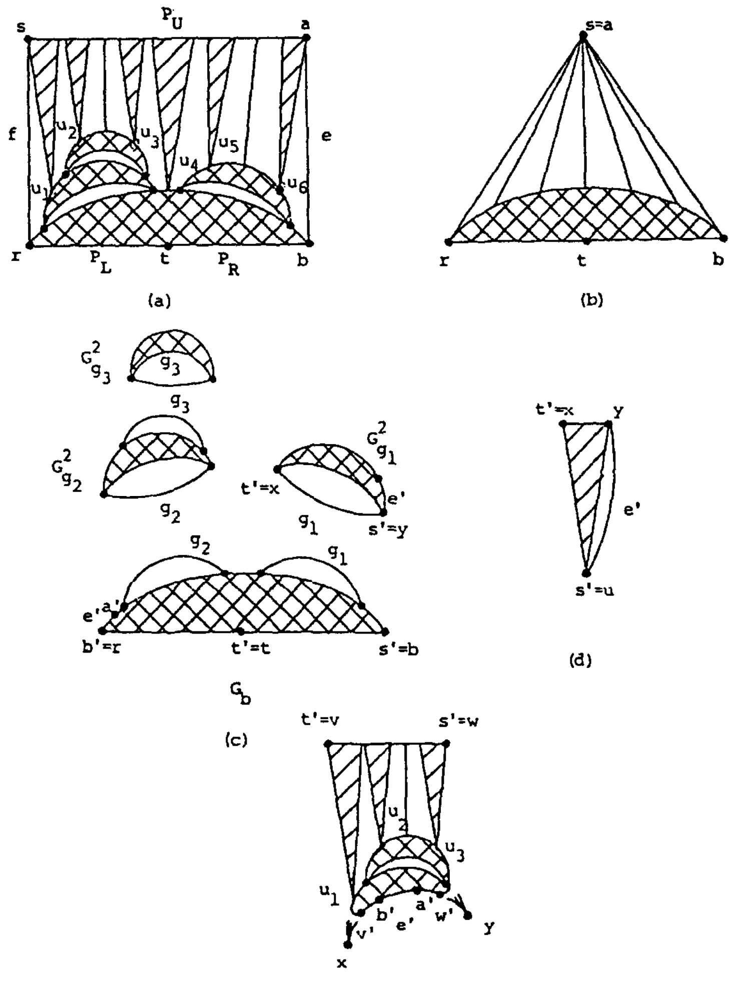

FIG. 3. Type II reduction for an internally 4-connected graph G having no vertical separation pair: (a) G with $s \neq a$; (b) G with $s = a$; (c) G_b and G_g^2, where g_1, g_2, and g_3 are virtual edges; (d) G_u^3; (e) G_g^4.

that there is no vertex of degree 2 on $P_{sa} - s - a$; otherwise an illegal separation pair would exist. Let x (resp. y) be the vertex of $P_{sa} \cap C_u^3$ which is nearest to s (resp. a) along P_{sa}. Clearly $x \neq y$. Add a new edge $e' = (u, y)$ to C_u^3 if it does not exist, and let G_u^3 be the resulting graph. Figure 3(d) illustrates G_u^3.

Finally we define G_g^4 for a virtual edge $g = (x, y)$. Schematically G_g^4 is a graph formed by merging G_g^2 with all the G_g^2, and C_u^3 drawn above G_g^2.

Formally we constructively define G_g^4 as follows (see Fig. 3(e)):

(1) $C_g^4 := G_g^2$;

(2) while C_g^4 has a virtual edge $g' \neq g$, iterate to merge $G_{g'}^2$ into C_g^4;

(3) for each cut vertex $u(\neq x, y)$ of $G - P_{sa}$ contained in C_g^4, merge C_u^3 to C_g^4;

(4) construct the subgraph of G induced by the vertices of C_g^4 together with the vertices of P_{sa} adjacent to C_g^4, and redefine C_g^4 as the subgraph;

(5) $G_g^4 := C_g^4 - x - y$.

Figure 3(e) illustrates $G_{g_2}^4$ for virtual edge g_2. Let v (resp. w) be the vertex of G_g^4 that appears first (resp. last) on P_{sa}. Note that G_g^4 must contain at least two vertices of P_{sa} and hence v and w are distinct. Let w' (resp. v') be the vertex which is adjacent to y (resp. x) and appears last (resp. first) on outer path P_{wv} of G_g^4.

We now have the following lemma.

LEMMA 2. *Let G be an internally 4-connected graph containing no vertical separation pair. Then all the decomposed graphs G_b, G_g^2, G_u^3, and G_g^4 are internally 4-connected with respect to (s', t', e') if s', t', and e' are defined as follows*:

(a) *Case G_b. Let $s' = b$, and $t' = t$. If $t \neq r$, let e' be the edge clockwise incident to r on the outer facial cycle Z' of G_b; if $t = r$, let e' be the edge counterclockwise incident to b.*

(b) *Case G_g^2. Let $s' = y$, $t' = x$, and let e' be the edge counterclockwise incident to y on the outer facial cycle of G_g^2.*

(c) *Case G_u^3. Let $s' = u$, $t' = x$, and $e' = (u, y)$.*

(d) *Case G_g^4. Let $s' = w$ and $t' = v$. If $w' \neq v'$, let $e' = (a', b')$ be an arbitrary edge on outer path $P_{w'v'}$; if $w' = v'$, let e' be an arbitrary edge on P_{wv} incident to w' $(= v')$.*

Proof. Since the proof for cases (a)–(c) is trivial, we verify only the case (d). Suppose that G_g^4 is not internally 4-connected with respect to (s', t', e'). Then G_g^4 must have one of the following three:

(i) a cut vertex v_c;

(ii) a separation pair $\{v_{p1}, v_{p2}\}$ such that both v_{p1} and v_{p2} are contained in path $P_{wa'}$, $P_{b'v}$, or P_{vw}; and

(iii) a separation triple $\{v_{t1}, v_{t2}, v_{t3}\}$ for which one of the components of $G_g^4 - \{v_{t1}, v_{t2}, v_{t3}\}$ contains no vertex of the outer facial cycle Z' of G_g^4.

In either case G would contain a separation triple $\{v_c, x, y\}$, $\{v_{p1}, v_{p2}, x\}$, $\{v_{p1}, v_{p2}, y\}$ or $\{v_{t1}, v_{t2}, v_{t3}\}$ such that the deletion of the triple from G produces a component containing no vertex of Z.

This contradicts the internally 4-connectedness of G. □

Lemma 3 below claims that a plane graph internally 4-connected with respect to (s, t, e) has a Hamiltonian path joining s and t. Although the claim is implied by Lemma 1, we give a constructive proof, from which a simple algorithm for finding a Hamiltonian path follows immediately.

LEMMA 3. *Let G be a plane graph having an outer facial cycle Z. Let s and t be two distinct vertices on Z and let e ($\neq (s, t)$) be an edge on Z. If G is internally 4-connected with respect to (s, t, e), then G has a Hamiltonian path $P(G, s, t, e)$ which connects s and t and contains e. Moreover, if G has no vertical separation pair, then $P(G, s, t, e)$ does not contain edge $f = (s, r)$.*

Proof. The proof is by induction on the number $|V|$ of vertices of a graph G. If $|V| = 3$, the claim is clearly true, so we assume that $|V| > 3$. There are two cases to consider.

Case 1. There exists a vertical separation pair $\{x, y\}$ (see Fig. 2).

Denote by G_l and G_r the two $\{x, y\}$-split graphs. One may assume that G_r contains e. Among all vertical separation pairs of G, we choose $\{x, y\}$ such that G_r has the smallest number of vertices. We call such a separation pair the *rightmost* separation pair.

First consider the case $t \in G_l$ as shown in Fig. 2. Let $e' = (x, y)$ be a virtual edge. Then clearly G_l is internally 4-connected with respect to (s, t, e'), while G_r is internally 4-connected with respect to (x, y, e) (or with respect to (y, x, e) if $y = b$). Therefore by the inductive hypothesis G_l has a Hamiltonian path $P(G_l, s, t, e')$ and G_r has $P(G_r, x, y, e)$ (or $P(G_r, y, x, e)$). Thus G has Hamiltonian path

$$P(G, s, t, e) = P(G_l, s, t, e') + P(G_r, x, y, e) - e'$$

or

$$P(G_l, s, t, e') + P(G_r, y, x, e) - e'.$$

Next consider the case $t \notin G_l$. Let $e' = (y, x)$, then G_l and G_r are internally 4-connected with respect to (y, s, e') and (x, t, e), respectively. Here $s \neq x$, since G is internally 4-connected with respect to (s, t, e). Thus G has a Hamiltonian path

$$P(G, s, t, e) = P(G_l, y, s, e') + P(G_r, x, t, e) - e'$$

as desired. Note that $P(G_r, x, t, e)$ does not contain (x, y), since G_r has no vertical separation pair.

Case 2. G has no vertical separation pairs (see Fig. 3).

By the inductive hypothesis and Lemma 2, all the decomposed graphs G_b, G_g^2, G_u^3, and G_g^4 have Hamiltonian paths. From them one can construct a Hamiltonian cycle $P(G, s, t, e)$ of G as follows:

(i) First set P to $P_{sb} + P(G_b, b, t, e')$.

(ii) Then P is modified into a Hamiltonian path $P(G, s, t, e)$ of whole G. Note that in the following modifications edge $f = (s, r)$ is never included in path P.

(ii.1) If there is a virtual edge $g = (x, y)$ in G_b, modify P as follows: if $g \in P$, then let $P := P - g + P(G_g^2, y, x, e')$ (that is, replace g of P by a Hamiltonian path of G_g^2), and subsequently merge G_g^2 into G_b;
if $g \notin P$, then construct G_g^4, and set $P := P - P_{vw} + P(G_g^4, w, v, e')$ (that is, replace the subpath P_{vw} of P by a Hamiltonian path of G_g^4).

(Thomassen's proof misses the counter part of the argument on G_g^4 above.)

(ii.2) Finally, for each u of the cut vertices of $G - P_{sa}$ contained in G_b, set $P := P - P_{xy} + P(G_u^3, u, x, (u, y)) - (u, y)$ (that is, replace the subpath P_{xy} of P by a Hamiltonian path of G_u^3, and delete an extra edge (u, y)).

Clearly the resulting P forms a Hamiltonian path $P(G, s, t, e)$ of whole G. □

EXAMPLE. For an illustration consider G in Fig. 3(a). If $P(G_b, b, t, e')$ contains virtual edge g_1 but not g_2, then $P(G, s, t, e)$ is constructed from Hamiltonian paths of G_b, $G_{g_1}^2$, $G_{u_4}^3$, $G_{u_5}^3$, $G_{u_6}^3$, and $G_{g_2}^4$. If $P(G_b, b, t, e')$ contains both g_1 and g_2 and if $P(G_{g_2}^2, s', t', e')$ does not contain g_3, then $P(G, s, t, e)$ is constructed from Hamiltonian paths of G_b, $G_{g_1}^2$, $G_{g_2}^2$, $G_{g_3}^4$, $G_{u_1}^3$, $G_{u_4}^3$, $G_{u_5}^3$, and $G_{u_6}^3$.

3. ALGORITHM AND $O(n^2)$ BOUND

The proof of Lemma 3 immediately yields a recursive algorithm which finds a Hamiltonian path $P(G, s, t, e)$ in an (internally) 4-connected planar graph G. The reduction performed with respect to a vertical separation pair at Case 1 is called a "Type I reduction" (Fig. 2), while the reduction at

Case 2 is called a "Type II reduction" (Fig. 3). Then the algorithm is as follows.

```
procedure HPATH(G, s, t, e);
  begin
    if G contains exactly three vertices
      then {G is a triangle}
          return a (trivial) Hamiltonian path P(G, s, t, e)
      else if G has a vertical separation pair
            then Type I reduction
            else Type II reduction
  end;
```

As shown below it is rather easy to verify an $O(n^2)$ time bound for the algorithm HPATH. Clearly the running time of the algorithm is dominated by the time required by the graph decomposition: the decomposition of G into G_l and G_r in Type I reductions; and into G_b, G_g^2, G_u^3, and G_g^4 in Type II reductions. Hopcroft and Tarjan have given a linear algorithm which decomposes a given graph into 3-connected components [5]. Using a similar algorithm, the decomposition above can be done in linear time per reduction. (Note that our situation is much easier than theirs since we can use the plane embedding of G.) Furthermore we have the following lemma.

LEMMA 4. *If the input graph G of the algorithm HPATH has n vertices, then there are at most $n - 3$ reductions during one execution of HPATH.*

Proof. Assume that r reductions occured during one execution of HPATH. Then we should verify $r \leq n - 3$. Let $d(i)$, $1 \leq i \leq r$, be the number of smaller graphs into which a graph is decomposed at the ith reduction. That is, $d(i)$ is the number of the recursive calls occured at the reduction. Then $d(i)$ is necessarily 2 if the ith reduction is Type I, while $d(i)$ is 1 if the ith reduction is Type II and none of G_g^2, G_g^4, and G_u^3 is produced. Let r' be the number of reductions with $d(i) = 1$. Recall that G is eventually decomposed into triangles, for which Hamiltonian paths can be found trivially and no more reductions occur. Let t be the number of these final triangles.

Consider the so-called recursive call tree. Each internal node of the tree corresponds to a reduction, its sons to the recursive calls at the reduction, and the leaves to the final triangles. Thus the tree has t leaves and r internal nodes, r' of which have outdegree 1. The trivial fact that the number of internal nodes in a binary tree is one less than the number of leaves implies that the recursive call tree has at most $t - 1$ internal nodes of outdegree 2 or more. Thus we have

$$r \leq t - 1 + r'. \tag{1}$$

A Type I reduction decomposes a graph G into G_l and G_r having two duplicated vertices. The total length (i.e., number of edges) n of Hamiltonian paths in G_l and G_r is one larger than the length $n - 1$ of a Hamiltonian path in G. Thus a Type I reduction increases by one the total length of paths which will be found in the two reduced graphs. In general, the ith reduction increases by at most $d(i) - 1$ the total length of paths which will be found in the $d(i)$ reduced graphs. Conversely, if $d(i) = 1$, then the length of a path which will be found in the reduced graph decreases by at least one. Trivially HPATH initially wishes to find a Hamiltonian path of length $n - 1$. Therefore the total length of paths found in triangles cannot exceed

$$\begin{aligned} n - 1 + \sum_{1 \le i \le r} (d(i) - 1) - r' &= n - 1 + r + t - 1 - r - r' \\ &= n + t - r' - 2. \end{aligned}$$

Since the lengths of the Hamiltonian paths found in the final triangles total $2t$, we have

$$2t \le n + t - r' - 2;$$

that is,

$$t \le n - r' - 2. \tag{2}$$

Combining (1) with (2), we get the claimed bound on r

$$r \le n - 3. \qquad \square$$

Thus it is rather straightforward to implement the algorithm HPATH to run in $O(n^2)$ time.

Remark. Gouyou-Beauchamps' algorithm seems to be more complicated, partly due to the fact that it is based on the original lengthy proof of Tutte's theorem. Furthermore, he needed a lengthy argument to polynomially bound the algorithm. In our terminology, when G has no vertical separation pair as shown in Fig. 3(a), his algorithm decomposes G into G_u^3 and C_b (instead of G_b). If the (not necessarily Hamiltonian) path found in C_b does not pass through vertices in $G_{g_2}^2$, then his algorithm recurses to $G_{g_2}^4$ and then constructs a single path from the two paths in C_b and $G_{g_2}^4$. Thus some edges and vertices in $G_{g_2}^2$ and $G_{g_3}^2$ are contained in both C_b and $G_{g_2}^4$ to which his algorithm recurses. This is the obstruction that makes the analysis of his algorithm quite hard, although he eventually gave an $O(n^3)$ bound.

4. Linear Implementation

In this section, we refine the algorithm HPATH to run in linear time.

We first give a precise implementation of a Type I reduction. In a Type I reduction G is decomposed into two graphs G_l and G_r as in the proof of Lemma 3. In order to make the analysis easy, we decompose G into two or more graphs at once as follows. Continue to split G_l into G_l' and G_r' at the rightmost separation pair $\{x', y'\}$ of G_l while G_l contains vertex t and a vertical separation pair. Further split G_l into G_l' and G_r' if G_l contains a vertical separation pair of form $\{v, y'\}$ for some vertex v. We then apply the algorithm recursively to all the split graphs all together (see Fig. 2(c)). We newly call this operation a *Type I reduction*. The vertical separation pairs involved in the Type I reduction are called *usable*. (The graph G in Fig. 2(a) has five usable vertical separation pairs together with one nonusable pair $\{v, w\}$.)

We then show how to find the usable vertical separation pairs. Here we do not want to spend linear time since reductions occur $O(n)$ times. Making use of the plane embedding, one can do it spending less than linear time. Let $v \in P_{sa}$ and $u \in P_{br}$, then $\{v, u\}$ is a vertical separation pair if and only if v and u are on the same inner facial cycle other than the two inner facial cycles containing edge e or f. Thus traversing all the inner facial cycles passing through a particular vertex v on P_{sa}, one can find every vertex u with which v forms a vertical separation pair. Repeating this procedure for each vertex v on path P_{sa} from a to s, we can find all the vertical separation pairs in the rightmost order. Using an appropriate numbering of vertices on the outer boundary, one can immediately decide whether a given vertex on P_{br} is either on P_{bl} or P_{tr}. Therefore we can know which pairs are usable. Once these usable separation pairs are known,

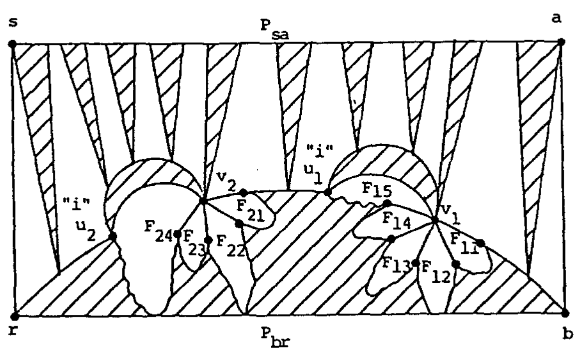

FIG. 4. Finding path P_b.

we can immediately split a graph into the smaller graphs (for example, G in Fig. 2(a) into those in Fig. 2(c)). Thus we have the following lemma.

LEMMA 5. *The time spent by a Type I reduction is proportional to the time for tranversing all the inner facial cycles passing through vertices on P_{sa} (which are called the P_{sa}-cycles from now on).*

We next show how to implement a Type II reduction. Since graphs G_g^2, G_u^3, and G_g^4 are not always disjoint and some of them need not be constructed, we delay the actual constructions until they become necessary.

Let P_b be the "outer" path on the outer facial cycle of G_b counterclockwise going from b to r. If P_b is known, then the graph G_b is easily constructed from G. The path P_b is found by the following procedure (see Fig. 4):

```
procedure PATHPB;
  begin
    color all the vertices and edges of P_sa-cycles "red";
    P_b := b;  {initialize path P_b as a single vertex b}
    SP := Ø; {SP is a list of separation pairs of block C_b}
    v := b;    {v is a possible vertex of a separation pair}
    let F_1 be the face which is incident to b and clockwise next to the outer face Z
    repeat
      let F_1, F_2, ..., F_l be all the inner facial cycles incident to v which are ordered
      clockwise around v;
      for i = 1 to l
        do begin
            let u_0(= v), u_1, ..., u_k be the vertices on F_i ordered clockwise;
            traverse clockwise cycle F_i from u_2 to u_k; {{u_0, u_1} is not a separation pair
            of C_b}
            if a red vertex u is found
              then exit from the for-statement;
          end;
      if there is no red edge (u, v)
          then begin {{u, v} is a separation pair of C_b}
                add {u, v} to SP;
                add a virtual edge (u, v) in the interior of F_i;
                F_i := u_0(= u), v, u_1, ..., u_0
              end;
      P_b := P_b + u;  {P_b proceeds to u}
      v := u; F_1 := F_i
    until v = r  {P_b reaches r}
  end;
```

As in procedure PATHPB one can find path P_b and simultaneously split G into two graphs G_b and G'. (See Fig. 5.) Clearly one execution of PATHPB is done within time proportional to the time spent for traversing all the facial cycles of G_b incident to $P_b - r$. However, in order to achieve a linear bound of HPATH, we need to slightly modify PATHPB so that one

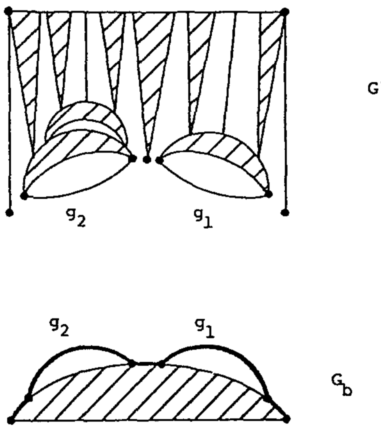

FIG. 5. Splitting G (shown in Fig. 3(a)) into G_b and G'. (Path P_b in G_b is drawn in a thick line.)

need not traverse the facial cycles incident to both P_b and P_{b_r} that have been traversed so far. (In Fig. 4 the procedure PATHPB above traverses faces F_{12}, F_{22}, and F_{24} although they may have been traversed so far.) For the purpose we maintain for each vertex u a list $L(u)$ of traversed faces on which u lies. (Thus in Fig. 4 $F_{12} \in L(v_1)$, $F_{22}, F_{24} \in L(v_2)$, and $F_{24} \in L(u_2)$ if F_{12}, F_{22}, and F_{24} have been traversed so far.) Furthermore we color the vertices on faces traversed so far "green." The procedure AVOID-DUPLICATE below finds all the separation pairs of C_b which lie on faces traversed so far (such as (v_2, u_2) in Fig. 4). More precisely, for all faces f traversed so far, AVOID-DUPLICATE constructs the lists $CV(f)$ of candicate vertices. If $|CV(f)| \geq 2$ and f is a facial cycle in C_b, then the first and the last vertices in $CV(f)$ compose a separation pair of C_b lying on f which should be added to SP in PATHPB.

procedure AVOID-DUPLICATE;
begin
let $F_1, F_2, \ldots, F_l$ be all the P_{sa}-cycles ordered from a to s;
for $i := 1$ **to** l **do**
begin
let $u_1(\in P_{sa}), u_2, \ldots, u_k$ be the vertices on F_i ordered clockwise;
for $j := 1$ **to** k **do**
if u_j is a "green" vertex and has no mark, that is, u_j lies on a cycle traversed so far and has not been traversed by this procedure
then
begin
for each face $f \in L(u_j)$ **do**
$CV(f) := CV(f) + \{u_j\}$;

```
                mark the vertex u_j
            end
        end;
    {we next update lists L(u)}
    for i := 1 to l do
        for each vertex u_j on F_i do
            L(u_j) := L(u_j) + {F_i};
    remove the marks from the "green" vertices on P_sa-cycles;
    initialize all nonempty lists CV(f) to empty ones
end;
```

We execute AVOID-DUPLICATE just before the execution of PATHPB. (In Fig. 4 we have $CV(F_{12}) = \{v_1\}$, $CV(F_{22}\} = \{v_2\}$, and $CV(F_{24}) = \{v_2, u_2\}$.) Every two vertices contained in $CV(f)$ is a separation pair lying on face f traversed so far. Thus in the modified procedure PATHPB we can skip the traverse of faces which have been traversed so far (such as F_{11}, F_{22}, and F_{24} in Fig. 4). Note that after the execution of PATHPB we need to recolor all the vertices u traversed by PATHPB "green" and to update the lists $L(u)$. Hereafter, we call the procedure modified as above PATHPB.

We then find a Hamiltonian path $P(G_b, b, t, e')$ by procedure HPATH(G_b, b, t, e'). The path P of G is first set as $P := P_{sb} + P(G_b, b, t, e')$. When P contains a virtual edge $g = (x, y)$, we need to split G_g^2 from G'. G_g^2 can be split from G' by applying the same procedure PATHPB to G' with setting $v := y$ and $r := x$. Redefine G' as the remaining graph. Replace edge g in P by a Hamiltonian path HPATH(G_g^2, y, x, e') of G_g^2. Repeat these operations while P contains a virtual edge. Then we eventually obtain a path P having no virtual edges. Let G_b' be the graph obtained from G_b by merging all the G_g^2 above. We replace virtual edges g in the outer path P_b of G_b by the outer paths P_{yx} in each of G_g^2 involved in the operations above, and let P_b' be the resulting outer path of G_b'. Clearly the time for constructing G_b and all G_g^2 above is proportional to the time for traversing all the facial cycles in G_b' incident to path $P_b' - r$ except those which are incident to P_{br} and have been traversed so far. We call these traversed facial cycles "P_b'-cycles." (Figure 6 illustrates graph G_b' and path P_b' for graph G shown in Fig. 3(a), assuming that $P(G_b, b, t, e')$ contains virtual edge g_1 but not g_2.)

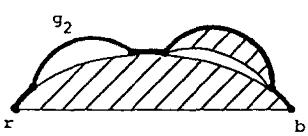

FIG. 6. Graph G_b' and path P_b' (drawn in a thick line).

Other graphs which must be constructed are G_u^3 or G_g^4. We can easily split these graphs from G' by traversing the outer facial cycle of G'. Thus this can be done within time proportional to the time for traversing the P_{sa}-cycles.

Thus we have the following lemma.

LEMMA 6. *The time spent by a Type II reduction is proportional to the time for traversing the P_{sa}- and P_b'-cycles.*

5. LINEAR BOUND

In this section we will show that each of the edges, including virtual edges, is traversed at most constant times during one execution of HPATH, so HPATH runs in linear time. At each stage of the execution of HPATH the set of edges is partitioned into 23 classes defined below. An edge belonging to a certain class at a stage may transit to another class at the next stage. We investigate how an edge transits the classes and show that every edge which is traversed more than some constant times disappears from a graph. (All the edges in a graph remain in the subgraphs decomposed by a Type I reduction. However, some edges disappear after a Type II reduction: all the edges incident to s disappear if $s = a$; and some of the edges incident to vertices on P_{sa} disappear if $s \neq a$; furthermore, the edges incident to x or y in C_g^4 disappear when constructing G_g^4.)

An edge e belongs to exactly two facial cycles F and F'. In order to analyze the time complexity, we regard e as a pair of multiple edges e_F and $e_{F'}$, which belong to F and F', respectively. Thus when face F is traversed, edge e_F is charged but $e_{F'}$ is not. The 23 classes of edges are denoted by $\langle X \rangle$ or $\langle X' \rangle$, where X is one of the 13 characters in sequence $Y, H, f, e, u, n, s, t, a, b, r, U, L, R, B, O$. Each class $\langle X \rangle$ consists of all the edges that are in components with $s \neq a$ and lie on cycles specified below. Class $\langle X^* \rangle$ consists of all the edges that lie on the same type of cycles as $\langle X \rangle$ but are contained in components with $s = a$. Note that although the graph is a single connected component at the first stage, it is decomposed into several components thereafter. When an edge can be in two or more classes, the edge is defined to be in the first class. Thus every edge is in exactly one of the 23 classes. Note that the classes $\langle n^* \rangle$, $\langle a^* \rangle$, and $\langle U^* \rangle$ are all empty:

$\langle Y \rangle, \langle Y^* \rangle$: inner facial cycles incident to all three paths P_{sa}, P_{bt} and P_{tr} (Note that each component contains at most one such cycle. In Fig. 2(a) the cycle containing x', y', and z' is such a cycle. The cycle looks like "Y turned upside down." From now on we simply write "cycle" for "inner facial cycle");

$\langle H\rangle, \langle H^*\rangle$: cycles incident to both paths P_{tr} and P_{bt} in a graph with $r \neq t$ (Such a cycle contains a horizontal separation pair.);

$\langle f\rangle, \langle f^*\rangle$: cycles containing an edge $\mathbf{f} = (s, r)$;

$\langle e\rangle, \langle e^*\rangle$: cycles containing an edge $\mathbf{e}$;

$\langle u\rangle, \langle u^*\rangle$: cycles containing a separation pair usable for a Type I reduction;

$\langle n\rangle$: cycles containing a separation pair $\{v, w\}$ nonusable for a Type I reduction such that $v \in P_{sa}$ and $w \in P_{tr} - t$;

$\langle s\rangle, \langle s^*\rangle$: cycles incident to a vertex $\mathbf{s}$;

$\langle t\rangle, \langle t^*\rangle$: cycles incident to a vertex $\mathbf{t}$;

$\langle a\rangle$: cycles incident to a vertex $\mathbf{a}$;

$\langle b\rangle, \langle b^*\rangle$: cycles incident to a vertex $\mathbf{b}$;

$\langle r\rangle, \langle r^*\rangle$: cycles incident to a vertex $\mathbf{r}$;

$\langle U\rangle$: cycles incident to a path $P_{\mathrm{U}} = P_{sa} - s - a$;

$\langle L\rangle, \langle L^*\rangle$: cycles incident to a path $P_{\mathrm{L}} = P_{tr} - t - r$;

$\langle R\rangle, \langle R^*\rangle$: cycles incident to a path $P_{\mathrm{R}} = P_{bt} - b - t$;

$\langle B\rangle, \langle B^*\rangle$: cycles in G_b' incident to the outer path P_{b}' (It should be noted that this class is defined only before a Type II reduction is applied.);

$\langle O\rangle, \langle O^*\rangle$: the other cycles.

If the edges contained currently in $\langle X\rangle$ were in classes $\langle Y_1\rangle, \langle Y_2\rangle, \ldots,$ or $\langle Y_k\rangle$ before a Type I reduction, then we write Prec I$(X) = \{Y_1, Y_2, \ldots, Y_k\}$. Similarly we define Prec II(X) with respect to a Type II reduction. When a Type I reduction and subsequently a Type II reduction occur, we denote by Prec I.II(X) the set of classes to which the edges currently in $\langle X\rangle$ belonged just before the last two reductions. Similarly define Prec II.I(X). Then we have the following lemmas, which can be proved by an easy but lengthy case study.

LEMMA 7. *A Type I reduction causes the following transitions of the classes*:

$\langle Y\rangle = \varnothing,$

$\mathrm{Prec\,I}(H) = \{H\},$

$\mathrm{Prec\,I}(f) = \{Y, f, u\},$

$\mathrm{Prec\,I}(e) = \{e, u\},$

$\langle u\rangle = \varnothing,$

$\langle n \rangle = \varnothing,$
$\text{Prec I}(s) = \{s, U\},$
$\text{Prec I}(t) = \{t, R\},$
$\text{Prec I}(a) = \{a, U\},$
$\text{Prec I}(b) = \{b, R\},$
$\text{Prec I}(r) = \{r, L\},$
$\text{Prec I}(U) = \{U\},$
$\text{Prec I}(L) = \{L\},$
$\text{Prec I}(R) = \{R\},$
$\text{Prec I}(B) = \{O\},$
$\text{Prec I}(O) = \{O\},$

$\text{Prec I}(Y^*) = \{Y, u, Y^*\},$
$\text{Prec I}(H^*) = \{H, n, H^*\},$
$\text{Prec I}(f^*) = \{Y, f, u, L, Y^*, f^*, u^*\},$
$\text{Prec I}(e^*) = \{Y, e, u, e^*, u^*\},$
$\langle u^* \rangle = \varnothing,$
$\text{Prec I}(s^*) = \{s, t, a, b, U, L, R, s^*\},$
$\text{Prec I}(t^*) = \{s, t, U, R, t^*, R^*\},$
$\text{Prec I}(b^*) = \{a, b, U, R, b^*, R^*\},$
$\text{Prec I}(r^*) = \{r, L, r^*\},$
$\text{Prec I}(L^*) = \{r, L, L^*\},$
$\text{Prec I}(R^*) = \{U, R, R^*\},$
$\text{Prec I}(B^*) = \{O, O^*\},$
$\text{Prec I}(O^*) = \{O, O^*\}.$

Proof. As illustrations, we verify some of the equations above. Just after a Type I reduction, clearly there is no edge in $\langle Y \rangle$, that is, $\langle Y \rangle = \varnothing$. Only the edges in $\langle H \rangle$ become edges in $\langle H \rangle$. Therefore $\text{Prec I}(H) = \{H\}$. We next verify $\text{Prec I}(f) = \{Y, f, u\}$. Since in a Type I reduction a graph is split at usable separation pairs, only edges in $\langle Y \rangle$, $\langle u \rangle$, or $\langle f \rangle$ itself may transit into class $\langle f \rangle$ in a decomposed graph. Consider the cycle of G in Fig. 2(a) which is incident to z' and clockwise next to the outer cycle. The edges on that cycle belonged to $\langle L \rangle$ before the reduction, but at present belong to the inner facial cycle containing $f' = (s', r')$ of the leftmost component in Fig. 2(c). However, $L \notin \text{Prec I}(f)$ since $s' = a'$ in the component. (In fact, $L \in \text{Prec I}(f^*)$ and $L \in \text{Prec I}(s^*)$.) Thus we have verified $\text{Prec I}(f) = \{Y, f, u\}$. □

LEMMA 8. *A Type II reduction causes the following transitions of the classes*:

$\mathrm{Prec\,II}(Y) = \{H, U, H^*\}$,
$\mathrm{Prec\,II}(H) = \{H, U, H^*\}$,
$\mathrm{Prec\,II}(f) = \{s, t, a, b, U, t^*, b^*\}$,
$\mathrm{Prec\,II}(e) = \{r, O, r^*\}$,
$\mathrm{Prec\,II}(u) = \{s, t, a, U, L, R, O, t^*, L^*, R^*\}$,
$\mathrm{Prec\,II}(n) = \{U, R, R^*\}$,
$\mathrm{Prec\,II}(s) = \{a, b, U, b^*\}$,
$\mathrm{Prec\,II}(t) = \{s, t, U, t^*\}$,
$\mathrm{Prec\,II}(a) = \{B, O, B^*\}$,
$\mathrm{Prec\,II}(b) = \{r, O, r^*\}$,
$\mathrm{Prec\,II}(r) = \{U, R, R^*\}$,
$\mathrm{Prec\,II}(U) = \{B, O, B^*\}$,
$\mathrm{Prec\,II}(L) = \{U, R, R^*\}$,
$\mathrm{Prec\,II}(R) = \{L, O, L^*\}$,
$\langle B \rangle = \varnothing$,
$\mathrm{Prec\,II}(O) = \{O, O^*\}$,

$\mathrm{Prec\,II}(Y^*) = \{H, b, U, H^*, b^*\}$,
$\mathrm{Prec\,II}(H^*) = \{U, R, R^*\}$,
$\mathrm{Prec\,II}(f^*) = \{s, t, a, b, r, U, B, O, t^*, b^*\}$,
$\mathrm{Prec\,II}(e^*) = \{e, a, b, U, B, b^*\}$,
$\mathrm{Prec\,II}(u^*) = \{a, U\}$,
$\mathrm{Prec\,II}(s^*) = \{a, b, U, B, O, b^*\}$,
$\mathrm{Prec\,II}(t^*) = \{s, t, r, U, B, t^*\}$,
$\mathrm{Prec\,II}(b^*) = \{a, U, B, B^*\}$,
$\mathrm{Prec\,II}(r^*) = \{U, R, O, R^*\}$,
$\mathrm{Prec\,II}(L^*) = \{U, R, O, R^*\}$,
$\mathrm{Prec\,II}(R^*) = \{U, B, O, B^*\}$,
$\langle B^* \rangle = \varnothing$,
$\mathrm{Prec\,II}(O^*) = \{O, O^*\}$.

Proof. We verify only $\mathrm{Prec\,II}(e) = \{r, O, r^*\}$. As shown in Fig. 3, only G_b and G_g^4 may have an edge in $\langle e \rangle$ since G_g^2 and G_u^3 satisfy $s' = a'$. Clearly such an edge had to be in $\langle r \rangle$, $\langle O \rangle$, or $\langle r^* \rangle$. □

Lemmas 7 and 8 lead us to the following Lemmas 9 and 10.

LEMMA 9. *A pair of successive reductions of Type I and Type II causes the following transitions*:

$\text{Prec I.II}(Y) = \{H, n, U, H^*\},$
$\text{Prec I.II}(H) = \{H, n, U, H^*\},$
$\text{Prec I.II}(f) = \{s, t, a, b, U, R, t^*, b^*, R^*\},$
$\text{Prec I.II}(e) = \{r, L, O, r^*\},$
$\text{Prec I.II}(u) = \{s, t, a, r, U, L, R, O, t^*, L^*, R^*\},$
$\text{Prec I.II}(n) = \{U, R, R^*\},$
$\text{Prec I.II}(s) = \{a, b, U, R, b^*, R^*\},$
$\text{Prec I.II}(t) = \{s, t, U, R, t^*, R^*\},$
$\text{Prec I.II}(a) = \{O, O^*\},$
$\text{Prec I.II}(b) = \{r, L, O, r^*\},$
$\text{Prec I.II}(r) = \{U, R, R^*\},$
$\text{Prec I.II}(U) = \{O, O^*\},$
$\text{Prec I.II}(L) = \{U, R, R^*\},$
$\text{Prec I.II}(R) = \{r, L, O, L^*\},$
$\text{Prec I.II}(B) = \varnothing,$
$\text{Prec I.II}(O) = \{O, O^*\},$

$\text{Prec I.II}(Y^*) = \{H, n, a, b, U, R, H^*, b^*, R^*\},$
$\text{Prec I.II}(H^*) = \{U, R, R^*\},$
$\text{Prec I.II}(f^*) = \{s, t, a, b, r, U, L, R, O, t^*, b^*, R^*\},$
$\text{Prec I.II}(e^*) = \{e, u, a, b, U, R, O, b^*, R^*\},$
$\text{Prec I.II}(u^*) = \{a, U\},$
$\text{Prec I.II}(s^*) = \{a, b, U, R, O, b^*, R^*\},$
$\text{Prec I.II}(t^*) = \{s, t, r, U, L, R, O, t^*, R^*\},$
$\text{Prec I.II}(b^*) = \{a, U, O, O^*\},$
$\text{Prec I.II}(r^*) = \{U, R, O, R^*\},$
$\text{Prec I.II}(L^*) = \{U, R, O, R^*\},$
$\text{Prec I.II}(R^*) = \{U, O, O^*\},$
$\text{Prec I.II}(B^*) = \varnothing,$
$\text{Prec I.II}(O^*) = \{O, O^*\}.$

Proof. By the definitions, we have

$$\mathrm{Prec\,I.II}(X) = \bigcup_{w \in \mathrm{Prec\,II}(X)} \mathrm{Prec\,I}(w).$$

Using this equation and Lemmas 7 and 8 we can easily verify the claimed equations. For example, since $\mathrm{Prec\,II}(Y) = \{H, U, H^*\}$,

$$\begin{aligned}\mathrm{Prec\,I.II}(Y) &= \mathrm{Prec\,I}(H) \cup \mathrm{Prec\,I}(U) \cup \mathrm{Prec\,I}(H^*)\\ &= \{H\} \cup \{U\} \cup \{H, n, H^*\}\\ &= \{H, n, U, H^*\}.\end{aligned}$$ □

LEMMA 10. *A pair of successive reductions of type II and Type I causes the following transitions of the classes*:

$\langle Y \rangle = \varnothing$,
$\mathrm{Prec\,II.I}(H) = \{H, U, H^*\}$,
$\mathrm{Prec\,II.I}(f) = \{H, s, t, a, b, U, L, R, O, H^*, t^*, b^*, L^*, R^*\}$,
$\mathrm{Prec\,II.I}(e) = \{s, t, a, r, U, L, R, O, t^*, r^*, L^*, R^*\}$,
$\langle u \rangle = \varnothing$,
$\langle n \rangle = \varnothing$,
$\mathrm{Prec\,II.I}(s) = \{a, b, U, B, O, b^*, B^*\}$,
$\mathrm{Prec\,II.I}(t) = \{s, t, U, L, O, t^*, L^*\}$,
$\mathrm{Prec\,II.I}(a) = \{B, O, B^*\}$,
$\mathrm{Prec\,II.I}(b) = \{r, L, O, r^*, L^*\}$,
$\mathrm{Prec\,II.I}(r) = \{U, R, R^*\}$,
$\mathrm{Prec\,II.I}(U) = \{B, O, B^*\}$,
$\mathrm{Prec\,II.I}(L) = \{U, R, R^*\}$,
$\mathrm{Prec\,II.I}(R) = \{L, O, L^*\}$,
$\mathrm{Prec\,II.I}(B) = \{O, O^*\}$,
$\mathrm{Prec\,II.I}(O) = \{O, O^*\}$,
$\mathrm{Prec\,II.I}(Y^*) = \{H, s, t, a, b, U, L, R, O, H^*, t^*, b^*, L^*, R^*\}$,
$\mathrm{Prec\,II.I}(H^*) = \{H, U, R, H^*, R^*\}$,
$\mathrm{Prec\,II.I}(f^*) = \{H, s, t, a, b, r, U, L, R, B, O, H^*, t^*, b^*, L^*, R^*\}$,
$\mathrm{Prec\,II.I}(e^*) = \{H, e, s, t, a, b, r, U, L, R, B, O, H^*, t^*, b^*, r^*, L^*, R^*\}$,
$\langle u^* \rangle = \varnothing$,
$\mathrm{Prec\,II.I}(s^*) = \{s, t, a, b, r, U, L, R, B, O, t^*, b^*, r^*, L^*, R^*, B^*\}$,

$\text{Prec II.I}(t^*) = \{s, t, a, b, r, U, L, B, O, t^*, b^*, L^*, B^*\}$,
$\text{Prec II.I}(b^*) = \{a, r, U, L, B, O, r^*, L^*, B^*\}$,
$\text{Prec II.I}(r^*) = \{U, R, O, R^*\}$,
$\text{Prec II.I}(L^*) = \{U, R, O, R^*\}$,
$\text{Prec II.I}(R^*) = \{U, L, B, O, L^*, B^*\}$,
$\text{Prec II.I}(B^*) = \{O, O^*\}$,
$\text{Prec II.I}(O^*) = \{O, O^*\}$.

Proof. Similar to that of Lemma 9. □

Lemmas 9 and 10 lead us to Lemmas 11 and 12, which imply that every edge is traversed at most constant times. Note that in one execution of a reduction, only the edges on P_{sa}- and P_b'-cycles are traversed some constant times.

LEMMA 11. *Every edge in G is involved in Type I reductions at most four times during one execution of HPATH.*

Proof. Since a Type I reduction decomposes a graph into small graphs having no vertical separation pair, the reduction does not occur successively. Thus we can divide the sequence of reductions into pairs, each consisting of two consecutive reductions of Type I and Type II. Note the fact that the Type I reduction in a pair may be "spurious"; that is, possibly there is no vertical separation pair and the Type I reduction does nothing. This fact does not violate Lemma 9, since $X \in \text{Prec I}(X)$ and $X^* \in \text{Prec I}(X^*)$ for every X and X^* except B and B^*. Note that B and B^* do not exist just before the Type I reduction. Call the execution of the kth pair stage k. Then we claim the following:

At the beginning of stage k the edges in each class have been involved in Type I reductions during the preceding stages at most the following number of times:

$\langle Y\rangle:2$,	$\langle H\rangle:2$,	$\langle f\rangle:2$,	$\langle e\rangle:1$,
$\langle u\rangle:2$,	$\langle n\rangle:1$,	$\langle s\rangle:1$,	$\langle t\rangle:2$,
$\langle a\rangle:0$,	$\langle b\rangle:1$,	$\langle r\rangle:1$,	$\langle U\rangle:0$,
$\langle L\rangle:1$,	$\langle R\rangle:1$,	$\langle B\rangle:\varnothing$,	$\langle O\rangle:0$,
$\langle Y^*\rangle:2$,	$\langle H^*\rangle:1$,	$\langle f^*\rangle:2$,	$\langle e^*\rangle:3$,
$\langle u^*\rangle:1$,		$\langle s^*\rangle:1$,	$\langle t^*\rangle:2$,
	$\langle b^*\rangle:1$,	$\langle r^*\rangle:1$,	
$\langle L^*\rangle:1$,	$\langle R^*\rangle:1$,	$\langle B^*\rangle:\varnothing$,	$\langle O^*\rangle:0$.

(Here "$\varnothing$" means that the class is empty.)

Clearly the claim above implies this lemma. We prove the claim by induction on k.

Obviously the claim is true when $k = 1$. Assume that the claim is true for stage k. Since only the P_{sa}-cycles are involved in the Type I reduction at stage k, the edges in classes $\langle Y \rangle$, $\langle f \rangle$, $\langle e \rangle$, $\langle u \rangle$, $\langle n \rangle$, $\langle s \rangle$, $\langle a \rangle$, $\langle U \rangle$, $\langle Y^* \rangle$, $\langle f^* \rangle$, $\langle e^* \rangle$, $\langle u^* \rangle$, and $\langle s^* \rangle$ are involved in a Type I reduction once more. The stage k causes the transition of classes which are described in Lemma 9. For example, $\mathrm{Prec\,I.II}(Y) = \{H, n, U, H^*\}$. Therefore at the beginning of stage $k + 1$ the edges in class $\langle Y \rangle$ have been involved in Type I reductions at most $\max\{2, 1 + 1, 0 + 1, 2\} = 2$ times. Thus we can verify our claim holds at stage $k + 1$. □

LEMMA 12. *Every edge in G is involved in Type II reductions at most six times during one execution of HPATH.*

Proof. We divide the sequence of reductions into pairs, each consisting of two consecutive reductions of Type II and Type I. As in the proof of Lemma 11, we call the execution of the kth pair stage k. Then we prove the following stronger claim by induction on k:

At the beginning of stage k the edges in each class have been involved in Type II reductions in the preceding stages at most the following number of times:

$\langle Y \rangle : \varnothing$,	$\langle H \rangle : 2$,	$\langle f \rangle : 4$,	$\langle e \rangle : 4$,
$\langle u \rangle : \varnothing$,	$\langle n \rangle : \varnothing$,	$\langle s \rangle : 3$,	$\langle t \rangle : 4$,
$\langle a \rangle : 1$,	$\langle b \rangle : 2$,	$\langle r \rangle : 2$,	$\langle U \rangle : 1$,
$\langle L \rangle : 2$,	$\langle R \rangle : 2$,	$\langle B \rangle : 0$,	$\langle O \rangle : 0$,
$\langle Y^* \rangle : 4$,	$\langle H^* \rangle : 2$,	$\langle f^* \rangle : 4$,	$\langle e^* \rangle : 5$,
$\langle u^* \rangle : \varnothing$,		$\langle s^* \rangle : 4$,	$\langle t^* \rangle : 4$,
	$\langle b^* \rangle : 2$,	$\langle r^* \rangle : 2$,	
$\langle L^* \rangle : 2$,	$\langle R^* \rangle : 2$,	$\langle B^* \rangle : 0$,	$\langle O^* \rangle : 0$.

Obviously the claim is true when $k = 1$. Assume that the claim is true for the kth stage. Since the P_{sa}- and P_b'-cycles are involved in the Type II reduction at stage k, the edges in classes $\langle f \rangle$, $\langle e \rangle$, $\langle s \rangle$, $\langle a \rangle$, $\langle b \rangle$, $\langle U \rangle$, $\langle B \rangle$, $\langle f^* \rangle$, $\langle e^* \rangle$, $\langle s^* \rangle$, $\langle b^* \rangle$, $\langle B^* \rangle$ are traversed once more in stage k. Although there are edges on P_b'-cycles which are contained in $\langle H \rangle$, $\langle t \rangle$, $\langle r \rangle$, $\langle L \rangle$, $\langle R \rangle$, $\langle H^* \rangle$, $\langle t^* \rangle$, $\langle r^* \rangle$, $\langle L^* \rangle$, $\langle R^* \rangle$, they are traversed once more in stage k only if they have not been traversed so far. Thus it is not necessary to charge the edges in classes $\langle H \rangle$, $\langle t \rangle$, $\langle r \rangle$, $\langle L \rangle$, $\langle R \rangle$, $\langle H^* \rangle$, $\langle t^* \rangle$, $\langle r^* \rangle$, $\langle L^* \rangle$, and $\langle R^* \rangle$. The kth stage causes the transition of classes as in Lemma 10. For example $\mathrm{Prec\,II.I}(H) = \{H, U, H^*\}$. Therefore at the beginning of stage $k + 1$ the edges in class $\langle H \rangle$ have been involved in Type II reductions at most $\max\{2, 1 + 1, 2\} = 2$ times. Thus we can show that our claim holds at stage $k + 1$. □

The following theorem is an immediate consequence of Lemmas 5, 6, 11, and 12.

THEOREM 1. *The algorithm HPATH runs in linear time.*

Proof. If the input plane graph G has n vertices, then G initially has $O(n)$ edges and $O(n)$ virtual edges are introduced. All these edges are involved in at most 10 reductions by Lemmas 11 and 12. Therefore HPATH runs in $O(n)$ time. □

Algorithm HPATH uses a usual data structure to represent a plane embedding of a graph [1]. Therefore HPATH uses linear space.

ACKNOWLEDGMENTS

We wish to thank Professor Nobuji Saito of Tohoku University for stimulating discussion on the subject, and the referees for many very helpful comments and suggestions.

REFERENCES

1. T. ASANO, S. KIKUCHI, AND N. SAITO, A linear algorithm for finding Hamiltonian cycles in 4-connected maximal planar graphs, *Discrete Appl. Math.* **7** (1984), 1–15.
2. N. CHIBA AND T. NISHIZEKI, A theorem on paths in planar graphs, *J. Graph Theory* **10** (1986), 449–450.
3. M. R. GAREY, D. S. JOHNSON, AND R. E. TARJAN, The planar Hamiltonian circuit problem is NP-complete, *SIAM J. Comput.* **5** (1976), 704–714.
4. D. GOUYOU-BEAUCHAMPS, The Hamiltonian circuit problem is polynomial for 4-connected planar graphs, *SIAM J. Comput.* **11** (1982), 529–539.
5. J. E. HOPCROFT AND R. E. TARJAN, Dividing a graph into triconnected components, *SIAM J. Comput.* **2**, No. 3 (1973), 135–158.
6. R. M. KARP, Reducibility among combinatorial problems, *in* "Complexity of Computer Computations" (R. E. Miller and J. W. Thatcher, Eds.), pp. 85–104, Plenum, New York, 1972.
7. O. ORE, "The Four Color Problem," Academic Press, New York, 1967.
8. C. THOMASSEN, A theorem on paths in planar graphs, *J. Graph Theory* **7** (1983), 169–176.
9. W. T. TUTTE, A theorem on planar graphs, *Trans. Amer. Math. Soc.* **82** (1956), 99–116.
10. W. T. TUTTE, Bridges and Hamiltonian circuits in planar graphs, *Aequations Math.* **15** (1977), 1–33.