反転禁止部分グラフを有する平面グラフ抽出法

高藤 大介*，姉ヶ山 伸一郎†，木下 敏行*，渡邉 敏正*

[概要] 本稿では，反転禁止部分グラフを含むグラフに対する全域平面最大部分グラフ抽出問題を扱う．この問題に対する発見的解法として，*PLAN-PWB*, *PLAN-MWW*, *PLAN-DIVIDE* が既に提案されている．まず，*PLAN-PWB* の修正版 *PLAN-PWB2* を提案し，次に，既存解法 *PLAN-MWW* の拡張版 *PLAN-MWW2*，既存解法 *PLAN-DIVIDE* の改良版 *PLAN-DIVIDE2* を提案する．さらに，これらを実装して計算機による既存解法との比較実験を行い，提案手法の性能を実験的に比較評価する．

Heuristic Algorithms for Extracting a Planar Graph with Subgraphs Forbidding their Turning Over

Daisuke Takafuji*, Shin'ichiro Anegayama†, Toshiyuki Kinoshita* and Toshimasa Watanabe*

[Abstract] The subject of this paper is the problem of extracting a maximum spanning planar subgraph, which must be embedded as specified. Heuristic algorithms *PLAN-PWB*, *PLAN-MWW* and *PLAN-DIVIDE* have been proposed so far for this problem. First, we propose *PLAN-PWB2* that is an improved version of *PLAN-PWB*. Then we propose two heuristic algorithms *PLAN-MWW2* and *PLAN-DIVIDE2*: the first one is extended from *PLAN-MWW*, and the second one is improved from *PLAN-DIVIDE* by using *PLAN-MWW2* instead of *PLAN-PWB*. Furthermore, experimental results are given to compare performance of these algorithms.

1 Introduction

[Problem] The problem of extracting a maximum spanning planar subgraph is defined as follows: "Given a graph $G = (V, E)$, find an edge set $E' \subseteq E$ with the maximum cardinality among all edge sets $E'' \subseteq E$ such that $G' = (V, E'')$ is a spanning planar subgraph of G".

We call an algorithm for extracting such a spanning planar subgraph $G' = (V, E')$ a *planarization algorithm*. Consider any planar graph $G_p = (V, E_p)$ with cycles $C_i \subseteq G_p (i = 1, \cdots, k)$ which must be embedded as specified (that is, each cycle C_i is forbidden to be turned over). Let $\widetilde{G_p}$ denote a plane embedding of G_p. If all C_i are embedded as specified in $\widetilde{G_p}$, $\widetilde{G_p}$ is called a plane embedding (of G_p) under "*forbiddance of turning over*". Given a graph $G = (V, E)$, a turn-forbidden planarization algorithm is an algorithm to extract a spanning planar subgraph $G_p = (V, E_p)$, with $E_p \subseteq E$, such that $\widetilde{G_p}$ is a plane embedding under forbiddance of turning over. In order to realize a TFP algorithm, we represent each specified cycle as a clockwise directed cycle, and any operation during the algorithm maintains clockwise directedness of these cycles.

[Motivation] For designing printed-wiring-boards or VLSI, we often represent a given circuit as a graph model: for example, a graph model in which a path or a directed cycle represents how pins of a given element are located, and a spanning tree does a connection requirement among pins. Generally speaking, most elements and some modules have a side to be faced to a board in actual mounting, and they cannot be placed upside down. We call such an element as *one-sided* element. Designing layout of each layer of single- or multi-layered boards requires extracting a spanning planar subgraph of a given graph model, where one-sided elements have to be handled.

If we represent each one-sided element as a clockwise directed cycle and apply a turn-forbidden planarization algorithm, then we can find planar layout in which all one-sided elements are placed as speci-

*広島大学大学院工学研究科／Graduate school of Engineering, Hiroshima University
†日立中国ソフトウェア／Hitachi Chugoku Software, Ltd.

fied. Turn-forbidden planarization algorithms have great importance practically.

[Known Results] The problem of extracting a maximum spanning planar subgraph problem is NP-hard [8] in general. It has been well investigated and many algorithms have ever been proposed [1, 3–6, 9, 10, 14, 15, 17, 18]. Unfortunately however, any algorithm in [3–6, 9, 17] is unlikely to be useful in such practical situations, while those in [1, 10, 14, 15, 18] can extract a spanning planar subgraph under the forbiddance of turning over. Turn-forbidden planarization algorithms are useful not only in the field of designing layout of printed-wiring-boards having one-sided elements but in extracting a spanning planar subgraph from a given graph that is too huge to be handled without reduction of its size. Algorithms for designing printed-wiring-boards or a VLSI have been proposed in [10, 13, 15, 18]. The one in [13] is based on a finding maximum-weight face: a linear time algorithm for finding a maximum-weight face of a given planar graph G_p has been proposed in [11] which also gives a linear time algorithm for finding a planar embedding G'_p of G_p such that the infinite face of G'_p is a maximum-weight face of G_p. An algorithm for finding a maximum-weight face is also proposed in [16].

[Purpose] First, in this paper, we propose *PLAN-PWB2* that is an improved version of the known algorithm *PLAN-PWB* [10]. Then we propose two other algorithms *PLAN-MWW2* and *PLAN-DIVIDE2*: the first one is extended from *PLAN-MWW* [13], and the second one is improved from *PLAN-DIVIDE* [1] by replacing *PLAN-PWB* with *PLAN-MWW2*. All of them are heuristic turn-forbidden planarization algorithms for extracting a spanning planar subgraph from a given large graph. We experimentally compare the proposed algorithms with the known algorithms. Experimental results for 180 randomly generated graphs $G = (V, E)$ with $2000 \le |V| \le 10000$ and $6000 \le |E| \le 100000$ show that *PLAN-DIVIDE* [1] can extract a spanning planar subgraph quickly while any other known algorithm cannot, and it is useful for extracting a spanning planar subgraph under forbiddance of turning over.

2 Basic Definitions

Because of space limitation, many definitions are omitted (see [2, 8] for example).

$G' = (V', E')$ is a *spanning planar subgraph of* $G = (V, E)$ if and only if $V' = V$, $E' \subseteq E$ and G' is planar. A spanning planar subgraph $G' = (V', E')$ of $G = (V, E)$ is *maximal* if and only if $G'' = (V, E' \cup \{e\})$ is nonplanar for any $e \in E - E'$. Suppose that G'_p is a planar embedding of G_p. Let $F(G'_p)$ denote the set of all faces in G'_p. For any face $f \in F(G'_p)$, the summation $w(f)$ of the weights of vertices and edges on the contour of the face f is called the *weight of* f. We call any face f' of G'_p with $w(f') = max\{w(f) \mid f \in F(G'_p)\}$ a *maximum-weight face of* G'_p and denote it as $f_{max}(G'_p)$. We call G''_p with $w(f_{max}(G''_p)) = max\{w(f_{max}(G'_p)) \mid G'_p$ is a planar embedding of $G_p\}$ a *planar embedding with maximum-weight face* of G_p, and $f_{max}(G''_p)$ is called a *maximum-weight face of* G_p. For a set $S \subset V$ of a graph $G = (V, E)$, let $G[S]$ denote the graph (S, E_S), where $E_S = \{e = (u, v) \in E \mid u, v \in S\}$. $G[S]$ is called *the subgraph induced by* S of G. V or E is sometimes represented as $V(G)$ or $E(G)$, respectively. For any two vertex sets $S_i \subseteq V$ $(i = 1, 2)$, $K(S_1, S_2; G) = \{(u_1, u_2) \in E \mid u_1 \in S_1$ and $u_2 \in S_2\}$.

3 Known Algorithms

3.1 PQR-trees [12]

A PQR-tree is a data structure for turn-forbidden planarity testing. A PQR-tree, introduced first in [12], is a directed ordered rooted tree consisting of four kinds of nodes: P-nodes, Q-nodes, R-nodes and leaves. Fig. 1 shows an example of a PQR-tree, where a circle, a rectangle without an arrow and a rectangle with an arrow denote a P-node, a Q-node, an R-node, respectively. All nodes except R-nodes are elements of well-known PQ-trees [2]. Two PQR-trees T and T' are *equivalent* (denoted by $T \equiv T'$) if and only if T' is obtained from T by repeating any one the following two transformations: (i) changing the order of children of a P-node arbitrarily, and (ii) reversing the order of children of a Q-node. Note that the order of children of any R-node cannot be changed. Let $F(T)$ denote the sequence defined by concatenating leaves of a given PQR-tree T from left to right. In Fig. 1, $F(T) = abcde$. $F(T)$ is called a *frontier* of T and represents a permutation. Let $con(T) = \{F(T') \mid T' \equiv T\}$. A set S consisting of only leaves of T is called a *leaf set* of T. Given a leaf set S of T, a *reduction* of T for S is a procedure to construct a PQR-tree T' such that

$con(T') = \{\pi \in con(T) \mid$ each element of S appear in π consecutively$\}$.

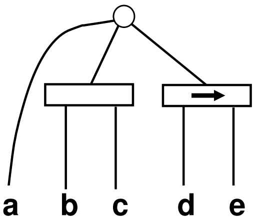

Figure 1: An example of a PQR-tree

If no reduction is possible then the current subgraph is nonplanar, and we search the present PQR-tree for a "minimum" set of edges whose deletion recover planarity. A reduction is done by one of template matchings. The details of template matchings are omitted (see [12]).

3.2 ***PLAN-PWB*** [10]

In this subsection, we describe a heuristic algorithm *PLAN-PWB* [10] for extracting a spanning planar subgraph using PQR-trees in Section 3.1. *PLAN-PWB* finds a spanning planar subgraph of G, such that every directed cycle is drawn clockwise, given a graph $G = (V, E)$ with the clockwise directed cycles. First, *PLAN-PWB* executes the following **(PWB1)**–**(PWB6)** to extract a spanning planar subgraph $G'[S] = (S, E'_S)$ of $G[S]$ for any biconnected component $S \subseteq V$ of G. Let E' be the union of E'_S for any biconnected component S of G. Then we obtain a spanning planar subgraph $G' = (V, E')$. *PLAN-PWB* uses PQR-trees for planarity testing.

(PWB1) $G[S] = (S, E_S)$, $n \leftarrow |S|$.

(PWB2) Calculate an st-number $r(v)$ for any $v \in S$. For simplicity, we consider the vertex $i \in V$ is the i-th st-numbered one. Construct a PQR-tree T consisting of only the vertex 1. $v \leftarrow 1$.

(PWB3) Add a vertex v into a PQR-tree as follows:

(1) Construct a PQR-tree T_v for the vertex v, where directed edges are handled carefully to avoid edges to be placed inside directed cycles. (Leaves of T_v correspond to vertices adjacent to v in G.)

(2) Delete all copies of the vertex v appearing as leaves of T, and add T_v into T by making the root of T_v as a child of the node to which those deleted copies were adjacent. Let T denote the resulting PQR-tree.

(3) $v \leftarrow v + 1$.

(PWB4) Let S be the set of those leaves of T corresponding to the vertex v of G. If a reduction of T for S is executable then go to **(PWB6)**

(PWB5) Find a minimum set of leaves of T such that, after edges that are incident upon those leaves are deleted from T, we can resume a reduction of the resulting PQR-tree for S.

(PWB6) In the PQR-tree obtained after reduction of **(PWB5)**, merge all elements of S, which appears consecutively, into one leaf (this leaf corresponds to the vertex v), and let T denote the resulting PQR-tree. If $v = n$ then halt else goto **(PWB3)**.

Definition 3.1 *[7] Let $G = (V, E)$ be a biconnected graph. An st-numbering is a bijection $r : V \rightarrow \{1, \ldots, |V|\}$ satisfying (i) and (ii). ($r(v)$ of each $v \in V$ is called an st-number.)*

(i) $(s, t) \in E, r(s) = 1$ *and* $r(t) = |V|$.

(ii) For any $v \in V - \{s, t\}$, there are two adjacent vertices $v' \in V$ and $v'' \in V$ satisfying $r(v') < r(v) < r(v'')$.

Theorem 3.1 *[7] Given a biconnected graph, an st-numbering is obtained in linear time.* □

PLAN-PWB gives a plane embedding under forbiddance of turning over by satisfying the following conditions (i)–(iii):

(i) Maintaining clockwise directedness of specified directed cycles.

(ii) Forbiddance of embedding any vertices or edges in the inside of directed cycles.

(iii) Forbiddance of deleting any edge in any specified directed cycles.

3.3 Hierarchical Planarization Algorithm ***PLAN-DIVIDE*** [1]

In this section, we describe a heuristic turn-forbidden planarization algorithm *PLAN-DIVIDE* [1] which extracts a spanning planar subgraph hierarchically. The purpose of *PLAN-DIVIDE* is to find a spanning planar subgraph of a given huge graph $G = (V, E)$ containing a family of directed cycles $\mathcal{K} = \{C_1, \ldots, C_k\}$ $(k \geq 1)$. Let max_edge be the maximum cardinality accordingly of an edge set that can be handled simultaneously by any existing planarization algorithm. First, *PLAN-DIVIDE* divides G with $|E| > max\_edge$ into some small graphs $G_i = (V_i, E_i)$ with $|E_i| \leq max\_edge$ for some $i \geq 1$. Then *PLAN-DIVIDE* extracts a spanning planar subgraph of each G_i, and finds planar edges from those edges connecting any pairs of subgraphs G_i and G_j $(i \neq j)$.

First, we extract subgraphs $G_i = (V_i, E_i)$ $(i = 1, \ldots, d)$ of G with $|E_i| \leq max\_edge$ if $|E| > max\_edge$, and set $d = 1$ otherwise, such that each vertex set $V(C_j)$ is a subset of some V_i. Let $\mathcal{K}_i$ be a family of directed cycles included in each G_i. Then $\mathcal{K}$ is partitioned as $\mathcal{K}_1 \cup \cdots \mathcal{K}_d$.

For any $i = 1, \ldots, d$, each spanning plane embedding $\widetilde{G_{f(i)}}$ of G_i in which every directed cycles are drawn clockwise, is obtained by *PLAN-PWB*. Let $E_C \subseteq E$ be a set of edges connecting any pair of graphs G_i and G_j with $i, j \in \{1, \ldots, d\}$ $(i \neq j)$. And then we extract the planar edges from E_C as follows. First represent the contour of the outer face of each $\widetilde{G_i}$ as a clockwise directed cycle C_i. Let $V_{red} = \bigcup_{i=1}^{d} V(C_i)$, $E_{red} = E_C \cup \left(\bigcup_{i=1}^{d} E(C_i)\right)$ and $G_{red} = (V_{red}, E_{red})$. If $|E_{red}| \leq max\_edge$, extract planar edges among E_C by applying *PLAN-PWB* to G_{red}. If $|E_{red}| > max\_edge$, put $G \leftarrow G_{red}$, and repeat above hierarchical planarization steps recursively. This is called a recursive step replacement with cycles. After some iteration, we can find G_{red} with $|E_{red}| \leq max\_edge$ and extract planar edges from E_C.

The description of *PLAN-DIVIDE* is as follows.

PLAN-DIVIDE

Input: A graph $G = (V, E)$ with $\mathcal{K} = \{C_1, \ldots, C_k\}$ $(k \geq 1)$, the maximum number of an edge set max_edge.

Output: An edge set $E' \subseteq E$ such that $G' = (V, E')$ is planar.

step 1. $\mathcal{K_V} \leftarrow \emptyset$, $E' \leftarrow \emptyset$, $H \leftarrow G$.

step 2. Repeat the follows until $|E(H)| \leq max\_edge$;

- **step 2-1.** Applying procedure *Find_Vertex_Set* to H, find a vertex set $S \subseteq V(H)$ with $|E_S| \leq max\_edge$, where $H[S] = (S, E_S)$. (Note that $V(C') \subseteq S$ or $V(C') \cap S = \emptyset$ for any $C' \in \mathcal{K}$.)
- **step 2-2.** For the vertex set $S \subseteq V(H)$, apply the follows (1)–(6).
  1. $H_S \leftarrow H[S]$.
  2. Extract a spanning planar subgraph $H'_S = (S, E'_S)$ of H_S by applying *PLAN-PWB* to H_S. $E' \leftarrow E' \cup E'_S$.
  3. Calculate a vertex weight $w(v)$ $(v \in S)$ defined by $w(v) = |K(\{v\}, V(H) - S; H)|$.
  4. Find a maximum weight face f_{max} of H'_S with the vertex weight and let $\widetilde{G_S}$ be a plane embedding such that f_{max} is an outer face.
  5. Exchange the outer face f_{max} of $\widetilde{G_S}$ for a cycle C'_S by applying *Replace_Cycle*.
  6. $E(H) \leftarrow (E(H) - E_S - K(S - V(C'_S), V(H) - S; H)) \cup E(C'_S)$, $V(H) \leftarrow (V(H) - S) \cup V(C'_S)$, $\mathcal{K} \leftarrow \mathcal{K} \cup \{C'_S\}$, $\mathcal{K}_V \leftarrow \mathcal{K}_V \cup \{C'_S\}$.

step 3. Extract a spanning planar subgraph H' of H by applying *PLAN-PWB* to H.

step 4. $E' \leftarrow E' \cup E(H') - \bigcup_{C' \in \mathcal{K}_V} E(C')$.

Procedure *Replace_Cycle* replaces each $\widetilde{G_i}$ with a directed cycle C'_s consisting of vertices in the contour of the outer face f_{max} of $G[S]''$. And procedure *Find_Vertex_Set* finds a vertex set $S \subseteq V$, satisfying the following (i) and (ii) for any given graph $G = (V, E)$ and a given family of directed cycles $\mathcal{K}$: (i) $|E_S| \leq max\_edge$ for $G[S] = (S, E_S)$; (ii) For any $C' \in \mathcal{K}$, $V(C') \subseteq S$ or $V(C') \cap S = \emptyset$. (The details of these procedure are omitted: see [1].)

Turn-forbidden planarization algorithms can be used in extracting a spanning planar subgraph with fixed embeddings of some specified subgraphs. Let us call any subgraph $H_j = (V_j, E_j)$ which must be embedded as specified as a fixed embedding subgraph. Given a graph $G = (V, E)$ with a family of fixed embedding subgraphs $\mathcal{K} = \{H_1, \ldots, H_d\}$, replace each H_j $(j = 1, \cdots, d)$ by a clockwise directed cycle C_j as follows. Let $N_j \in H_j$ be a set of vertices adjacent to some vertices $v \in V - V(H_j)$. For any H_j, construct directed cycle C_j with length $|N_j|$ by the follows: (i) deleting $E(H_j)$ from G, (ii) deleting all vertices $v \in V(H_j) - N_j$, and (iii) connecting vertices of N_j by directed edges so that a clockwise directed cycle C_j may be formed.

Clearly maintaining clockwise directedness corresponds handling fixed embedding subgraphs.

4 Improving or Extending Known Algorithms

In this section, we describe improvement and extension of some known turn-forbidden algorithms. In Section 4.1, we point that *PLAN-PWB* includes some logical error, and propose an improved version *PLAN-PWB2* by correcting it. In Section 4.2, we

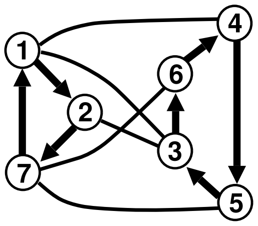

Figure 2: An example of an st-numbered graph G such that *PLAN-DIVIDE* can not planarize correctly

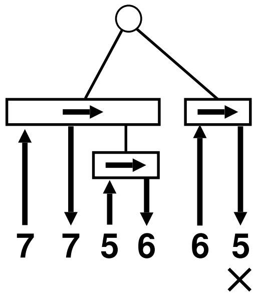

Figure 3: A PQR-tree T_4 cannot be reduced without deleting leaves corresponding to directed edges of a directed cycles such that their deletion is forbidden

propose a turn-forbidden planarization algorithm *PLAN-MWW2* by extending the algorithm *PLAN-MWW*.

4.1 Improving *PLAN-PWB*

In this subsection, we point that *PLAN-PWB* includes a logical error, and show how to correct it.

4.1.1 Conventional st-numbering in *PLAN-PWB*

Consider applying *PLAN-PWB* to the st-numbered graph shown in Fig. 2. Fig. 3 shows a PQR-tree T in reduction step for the vertex 5. In order to continue reduction of T, leaves corresponding to directed edges must be deleted, which is a contradiction. The reason why such deletion occurs is that two leaves corresponding to edges of the same directed cycle are placed separately in the PQR-tree. The existing st-numbering cannot avoid such a situation.

4.1.2 An inproved st-numbering

The proofs of the correctness of the modification is omitted due to shortage of space.

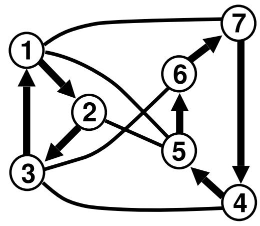

Figure 4: An example of an sc-st-numbered graph G such that turn-forbidden planarization can be executed

The improvement that we popose is to add the following condition to leaves included in a minimum set T found in **(PWB5)**: "leaves that do not correspond to edges of any directed cycle."

In oder to obtain desirable numbering of vertices, we propose the sc-st-numbering by modifying the conventional st-numbering as follows.

Definition 4.1 *Let $G = (V, E)$ be a biconnected graph with a family of directed cycles $\mathcal{K} = \{C_1, \ldots, C_k\}$ $(k \geq 1)$. An sc-st-numbering is the bijection $sc : V \to \{1, \ldots, |V|\}$. ($sc(v)$ of each $v \in V$ is called an sc-st-number.)*

(i) $(s,t) \in E$, $sc(s) = 1$ *and* $sc(t) = |V|$.

(ii) For any $v \in V - \{s,t\}$, there are two adjacent vertices $v' \in V$ and $v'' \in V$ satisfying $sc(v') < sc(v) < sc(v'')$.

(iii) Let $v_{sc_{max}}(C_j)$ be the largest sc-st-number of $V(C_j)$ and let $v_{sc_{min}}(C_j)$ be the smallest sc-st-number of $V(C_j)$ respectively. For each C_i with $1 \leq i \leq k$, the following condition is satisfied : For any $v \in V(C_i)$ such that $v \neq \{v_{sc_{max}}(C_i), v_{sc_{min}}(C_i)\}$, $sc(v') < sc(v) < sc(v'')$ or $sc(v'') < sc(v) < sc(v')$.

Theorem 4.1 *Given a biconnected graph, an sc-st-numbering is obtained in linear time.* □

Let *PLAN-PWB2* denote *PLAN-PWB* with an sc-st-numbering incorporated.

Theorem 4.2 *The PLAN-PWB2 is an $O(|V|^2)$ turn-forbidden planarization algorithm.* □

4.2 *PLAN-MWW2*

We propose a turn-forbidden planarization algorithm *PLAN-MWW2* by extending *PLAN-MWW*.

An algorithm *PLAN-MWW* for designing layout of printed wiring boards with few jumpes has been proposed in [13]. An input graph of *PLAN-MWW* is limited to only the special graph model of a given circuit. We propose a turn-forbidden planarization algorithm *PLAN-MWW2* by extending *PLAN-MWW* so that general graphs can be handled.

Any input graph of *PLAN-MWW2* is a general graph with a family of directed cycles $\mathcal{K} = \{C_1, \ldots, C_k\}$ $(k \geq 1)$.

PLAN-MWW2

Input: A graph $G = (V, E)$, with $\mathcal{K} = \{C_1, \ldots, C_k\}$ $(k \geq 1)$.

Output: A planar graph $G' = (V, E')$.

step 1. Set $E' \leftarrow E(\mathcal{K})$. Sort the $v \in V$ by the decrease order of the number of edges incident to v, let $\{v_1, v_2, \ldots, v_{|V|-1}, v_{|V|}\}$ be a such sorted vertex sequence. Then for any $i = 1, \ldots, |V|$, repeat the following step 2–4.

step 2: Let $Adj(v_i)$ be a set of vertices which are connected to v_i by $E - E'$. Give a weight $|V|$ to v_i, and give a weight 1 to every $v \in Adj(v_i)$. Then, give a weight 0 to every $v \in V - Adj(v_i) - \{v_i\}$.

step 3: Find a maximum-weight face of $G' = (V, E')$ (with the exception of an inner face constructed by directed closed path). Let V_F be a set of vertices constructing a maximum-weight face.

step 4: Let $Adj(v_i)'$ denote $Adj(v_i) \cap V_F$. If $|Adj(v_i)'| \geq 1$, find the edge set $E'' \subseteq E - E'$ between v_i and every $v' \in Adj(v_i)'$, add E'' to E'.

5 *PLAN-DIVIDE2*

In *PLAN-DIVIDE*, we use *PLAN-PWB* for extracting a spanning planar subgraph $G[S]'$ of each $G[S]$. Let *PLAN-DIVIDE2* denote *PLAN-DIVIDE* with *PLAN-PWB* replaced by *PLAN-MWW2* . We can expect that this replacement improves capability of the algorithm.

6 Experimental Results

[Implementation] We have implemented *PLAN-PWB2*, *PLAN-MWW2* and *PLAN-DIVIDE2* on a personal computer (CPU: Pentium IV/1.7GHz, OS: Free BSD 4.5-R) with the C programming code.

[Input data] Let $\mathcal{K}$ be a set of directed cycles $\{C_1, C_2, \ldots, C_n\}$, and $|V(\mathcal{K})|$ denotes $\bigcup_{i=1}^{n} |V(C_i)|$. Graphs $G = (V, E)$ with $\mathcal{K}$ satisfying the following are provided: $|V| \in \{2000, 5000, 10000\}$, $|E| = \{3|V|, 5|V|, 10|V|, 30|V|, 50|V|, 100|V|\}$ and $|V(\mathcal{K})| \leq \frac{|V|}{4}$, $3 \leq |V(C_i)| \leq \frac{|V(\mathcal{K})|}{2}$ $(1 \leq i \leq n)$ (generated by means of random numbers). The number of graphs is 10 for each pair $|V|$ and $|E|$: 180 input graphs in total.

[Comparison] We applied four turn-forbidden planarization algorithms *PLAN-DIVIDE* (DIV) [1], *PLAN-PWB2* (PWB2), *PLAN-MWW2* (MWW2) and *PLAN-DIVIDE2* (DIV2) for large graphs with $\mathcal{K}$, and compared the results. We has set $max\_edge = 2000$ for *PLAN-DIVIDE* and *PLAN-DIVIDE2*.

We show several results in Table 1 and Table 2. The terms "$|E_{np}|$" and "$CPU(s)$" show the number of nonplanar edges $|E - E'|$ and the CPU time (in second), respectively. And "—" in these tables shows that the algorithm could not extract a spanning planar subgraph of the data because of memory overflow. Also in the case that each algorithm could not extract a spanning planar subgraph within 24 hours, "—" is marked.

[Observation about experiment] Points of these results are summarized as follows.

(i) *PLAN-PWB* could not extract spanning planar subgraphs of graphs with over 200000 edges because of memory overflow.

(ii) *PLAN-DIVIDE* and *PLAN-DIVIDE2* could extract spanning planar subgraphs of graphs with over 200000 edges. *CPU* time of *PLAN-DIVIDE2* does not with the increase of $|E|$.

(iii) *PLAN-DIVIDE2* requires dividing an input graph into many small graphs in order to decrease computational time. And $|E_{np}|$ of *PLAN-DIVIDE2* increases as input graph is divided into more subgraphs. $|E_{np}|$ of *PLAN-DIVIDE2* is 97.06% of that by *PLAN-DIVIDE* in average. But *CPU* time of *PLAN-DIVIDE2* is much longer than *PLAN-DIVIDE.*

(iv) It is concluded from our experimental results that *PLAN-DIVIDE* is the most useful for extracting a spanning planar subgraphs of a large graph.

7 Concluding Remarks

In this paper, we have proposed two heuristic planarization algorithms *PLAN-DIVIDE2* and *PLAN-MWW2* under forbiddance of turning over. It is pointed out that many algorithms, including *PLAN-PWB*, assume that connectedness of st-numbering is kept after deletion of nonplanar edges:

Table 1: Comparison of average $|E_{np}|$

$\|V\|$	$\|E\|$	(DIV)	(MWW2)	(DIV2)	(PWB2)
2000	6000	3824.0	2925.0	3276.0	3824.0
	10000	7714.8	7699.0	7248.0	7714.8
	20000	17451.2	—	17283.0	17451.2
	60000	56897.6	—	56211.8	56772.8
	100000	96683.6	—	96079.8	96464.0
	200000	196144.0	—	195870.0	196017.0
5000	15000	9768.8	—	—	9768.8
	25000	19609.6	—	—	19578.8
	50000	44376.4	—	—	44233.2
	150000	143414.0	—	140693.0	142886.0
	250000	242797.0	—	240571.0	—
	500000	492029.0	—	490484.0	—
10000	30000	19715.5	—	—	19696.6
	50000	39567.5	—	—	39513.0
	100000	—	—	—	88941.8
	300000	288014.0	—	—	—
	500000	487364.0	—	—	—

Table 2: Comparison of CPU time (s)

$\|V\|$	$\|E\|$	(DIV)	(MWW2)	(DIV2)	(PWB2)
2000	6000	44.6266	31365.1	29600.7	43.3312
	10000	88.3891	19132.2	26648.8	81.6547
	20000	217.564	—	27164	197.867
	60000	242.495	—	21245.4	941.112
	100000	301.914	—	18934	1995.02
	200000	320.144	—	11987	6047.73
5000	15000	275.639	—	—	242.598
	25000	1333.11	—	—	567.895
	50000	696.833	—	—	1467
	150000	1015.36	—	109750	7272.91
	250000	741.52	—	72579	—
	500000	940.547	—	51927.9	—
10000	30000	8031.6	—	—	1116.07
	50000	3604.4	—	—	2746.01
	100000	—	—	—	7576.21
	300000	1201.94	—	—	—
	500000	1212.19	—	—	—

which is not always the case, causing some malfunction of algorithms. We have succeeded in correcting this error by proposing the sc-st-numbering. Moreover we have evaluated performance of two proposed algorithms and known algorithms experimentally. It is concluded that *PLAN-DIVIDE* [1] can quickly extract a spanning planar subgraph under forbiddance of turning over, showing usefulness in extracting a spanning planar subgraph from a given graph such that it is too large to be handled without reduction of its size.

Some problems left for future research are as follows: (i) proposing better graph partitioning algorithm such that the resulting planar subgraph has more edges. (ii) proposing better planarization algorithms under forbiddance of turning over.

References

[1] S. Anegayama, D. Takafuji, S. Taoka, and T. Watanabe, "Hierarchical extraction of a spanning planar subgraph under fixed embedding of specified subgraphs," IPSJ SIG Notes AL80–1, IPSJ, pp. 1–8, Sept. 2001 (in Japanese).

[2] K. S. Booth and G. S. Lueker, "Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms," *J. Comput. Sys. Sci.*, Vol. 13, No. 3, pp. 335–379, Dec. 1976.

[3] J. Cai, X. Han, and R. E. Tarjan, "An $O(m \log n)$-time algorithm for the maximal planar subgraph problem," *SIAM J. on Computing*, Vol. 22, No. 6, pp. 1142–1162, Dec. 1993.

[4] G. Călinescu, C. G. Fernandes, U. Finkler, and H. Karloff, "A better approximation algorithm for finding planar subgraphs," in *Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, New York/Philadelphia, pp. 16–25, ACM/SIAM, Jan. 28–30 1996.

[5] T. Chiba, I. Nishioka, and I. Shirakawa, "An algorythm of maximal planarization of graphs," in *Proc. IEEE Symp. on Circuits & Sys.*, pp. 649–652, 1979.

[6] H. Djidjev, "A linear algorithm for the maximal planar subgraph problem," in *Algorithms and Data Structures, 4th International Workshop*, S. G. Akl, F. K. H. A. Dehne, J.-R. Sack, and N. Santoro, Eds., Kingston, Ontario, Canada, Vol. 955 of *Lecture Notes in Computer Science*, pp. 369–380, Springer, 16–18 Aug. 1995.

[7] S. Even and R. E. Tarjan, "Computing an st-numbering," *Theoretical Computer Science*, Vol. 2, No. 3, pp. 339–344, Sept. 1976.

[8] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to NP-Completeness*, W.H. Freeman and Co., New York, 1979.

[9] O. Goldschmidt and A. Takvorian, "An efficient graph planarization two-phase heuristic," *Networks: An International Journal*, Vol. 24, pp. 69–73, 1994.

[10] K. Iwamoto, T. Watanabe, T. Araki, and K. Onaga, "Finding jumpers in printed wiring board design for analog circuits," in *Proc. of the 1991 IEEE International Symposium on Circuits and Systems*, pp. 2854–2857, 1991.

[11] K. Kotani, S. Masuda, and T. Kashiwabara, "An alogrithm for embedding a biconnected planar graph to to maximize the total sum of vertex-and edge-weights on the exterior window," *Trans. IEICE*, Vol. J74-A, No. 7, pp. 1041–1052, July 1991 (in Japanese).

[12] S. Masuda, T. Kashiwabara, and T. Fujisawa, "A wiring problem on single layer printed circuit board without mounting modules upside-down," *Trans. IEICE of Japan*, Vol. J66-A, No. 3, pp. 235–242, 1983 (in Japanese).

[13] A. Matsumoto, K. Yamaguchi, T. Kashiwabara, S. Masuda, and M. Taki, "A planarization algorithm in the layout design of single layer printed circuit board," Tech. Rep. COMP91-21, IEICE of Japan, pp. 79–88, 1991 (in Japanese).

[14] Y. Mizuguchi and T. Watanabe, "Application of the path-addition planarity testing algorithm to layout design of printed wiring boards," Tech. Rep. COMP94-87, IEICE of Japan, pp. 103–112, 1995 (in Japanese).

[15] K. Mizuno, T. Kobayashi, and T. Watanabe, "Extracting nonplanar connections in a terminal-vertex graph," in *Proc. of the 1999 IEEE International Symposium on Circuits and Systems*, pp. VI–121–VI124, 1999.

[16] T. Ozawa, "Efficient algorithms for planar embedding of graphs with constraints in placing specified vertices on face boundaries," in *Proc. of the 2002 IEEE International Symposium on Circuits and Systems*, pp. V–749–V–752, May 2002.

[17] T. Ozawa and H. Takahashi, "A graph-planarization algorithm and its application to random graphs," in *Proc. of the 17th Symposium of Research Institute of Electrical Communication on Graph Theory and Algorithms*, N. Saito and T. Nishizeki, Eds., Sendai, Japan, Vol. 108 of *LNCS*, pp. 95–107, Springer, Oct 1980.

[18] T. Yamaoki, S. Taoka, and T. Watanabe, "Extracting a planar spanning subgraph of a terminal-vertex graph by solving the independent set problem," in *Proc. of the 2001 IEEE International Symposium on Circuits and Systems*, pp. V–153–V–156, 2001.