Lempel, Even, and Cederbaum Planarity Method

Alexandre Noma[1]*, **Cristina G. Fernandes**[1]†

[1]Departamento de Ciência da Computação do IME-USP
Rua do Matão, 1010 — CEP 05508-090 São Paulo/SP

{noma,cris}@ime.usp.br

Abstract. *We present a simple pedagogical graph theoretical description of Lempel, Even, and Cederbaum (LEC) planarity method based on concepts due to Thomas. A linear-time implementation of LEC method using the PC-tree data structure of Shih and Hsu is provided and described in details. We report on an experimental study involving this implementation and other available linear-time implementations of planarity algorithms.*

1. Introduction

The first linear-time planarity testing algorithm is due to Hopcroft and Tarjan [8]. Their algorithm is an ingenious implementation of the method of Auslander and Parter [1] and Goldstein [7]. The second method of planarity testing proven to achieve linear time is due to Lempel, Even, and Cederbaum (LEC) [9]. This method was optimized to linear time thanks to the st-numbering algorithm of Even and Tarjan [6] and the PQ-tree data structure of Booth and Lueker (BL) [2].

All these algorithms are widely regarded as being quite complex. Recent research efforts have resulted in simpler linear-time algorithms proposed by Shih and Hsu (SH) [11] and by Boyer and Myrvold (BM) [5]. These algorithms implement LEC method and present similar and very interesting ideas. Each algorithm uses its own data structure to efficiently maintain relevant information on the (planar) already examined portion of the graph.

The description of SH algorithm made by Thomas [12] provided us with the key concepts to give a simple graph theoretical description of LEC method. This description increases the understanding of BL, SH, and BM algorithms, all based on LEC method. We implemented SH algorithm. Our implementation is available at `http://www.ime.usp.br/dcc/posgrad/teses/noma/` and is the unique available implementation of SH algorithm, even though the algorithm was proposed about 10 years ago. It is described in details by Noma in his dissertation [10] (see also [3, 4]), together with the results of an experimental study involving four planarity algorithms.

2. Frames, XY-paths, XY-obstructions and planarity

Let H be a planar graph. A subgraph F of H is a *frame of* H if F is induced by the edges incident to the external face of a planar embedding of H. If G is a connected graph, H is

* Supported by FAPESP 00/03969-2 (Brazil).
† Este trabalho foi coorientado por José Coelho de Pina.

a planar induced subgraph of G and F is a frame of H, then we say that F is a *frame of H in G* if it contains all vertices of H that have a neighbor in $V_G \setminus V_H$. Let F be a frame of H and P be a path in F. The *basis of* P is the subgraph of F formed by all blocks of F which contain at least one edge of P. Let $C_1, C_2, \ldots, C_k$ be the blocks in the basis of P. For $i = 1, 2, \ldots, k$, let $P_i := P \cap C_i$ and, if C_i is a cycle, let $\bar{P}_i := C_i \setminus P_i$, otherwise let $\bar{P}_i := P_i$. The *complement of* P *in* F is the path $\bar{P}_1 \cup \bar{P}_2 \cup \ldots \cup \bar{P}_k$, which is denoted by $\bar{P}$. If $E_P = \emptyset$ then $\bar{P} := P$.

Let W be a set of vertices in H and Z be a set of edges in H. A vertex v in H *sees W through Z* if there is a path in H from v to a vertex in W with all edges in Z. Let X and Y be sets of vertices of a frame F of H. A path P in F with basis S is an *XY-path* (Fig. 1) if

(p1) the endpoints of P are in X;
(p2) each vertex of S that sees X through $E_F \setminus E_S$ is in P;
(p3) each vertex of S that sees Y through $E_F \setminus E_S$ is in $\bar{P}$;
(p4) no component of $F - V_S$ contains vertices both in X and in Y.

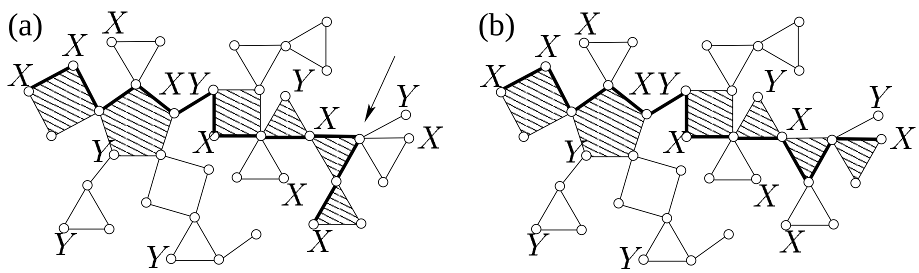

Figure 1: In (a) and (b), let P denote the thick path; its basis is shadowed. (a) P is not an XY-path since it violates (p3). (b) P is an XY-path.

There are three types of objects that obstruct an XY-path to exist. They are called *XY-obstructions* and are defined as

(o1) a 5-tuple (C, v_1, v_2, v_3, v_4) where C is a cycle of F and v_1, v_2, v_3, and v_4 are distinct vertices in C that appear in this order in C, such that v_1 and v_3 see X through $E_F \setminus E_C$ and v_2 and v_4 see Y through $E_F \setminus E_C$;
(o2) a 4-tuple (C, v_1, v_2, v_3) where C is a cycle of F and v_1, v_2, and v_3 are distinct vertices in C that see X and Y through $E_F \setminus E_C$;
(o3) a 4-tuple (v, K_1, K_2, K_3) where $v \in V_F$ and K_1, K_2, and K_3 are distinct components of $F - v$ such that K_i contains vertices in X and in Y.

The existence of an XY-obstruction is related to non-planarity as follows.

Lemma 2.1 (Thomas [12]) *Let H be a planar connected subgraph of a graph G and w be a vertex in $V_G \setminus V_H$ such that $G - V_H$ and $G - (V_H \cup \{w\})$ is connected. Let F be a frame of H in G, let X be the set of neighbors of w in V_F and let Y be the set of neighbors of $V_G \setminus (V_H \cup \{w\})$ in V_F. If F has an XY-obstruction then G has a subdivision of $K_{3,3}$ or K_5.*

3. Finding XY-paths or XY-obstructions

Let F be a connected frame and let X and Y be subsets of V_F. Let $\mathcal{B}$ be the set of blocks of a connected graph H and let T be the tree with vertex set $\mathcal{B} \cup V_H$ and edges of the form

Bv where $B \in \mathcal{B}$ and $v \in V_B$. We call T the *block tree*[1] of H (Fig. 2). Each node of T in $\mathcal{B}$ is said a *C-node* and each node of T in V_H is said a *P-node*.

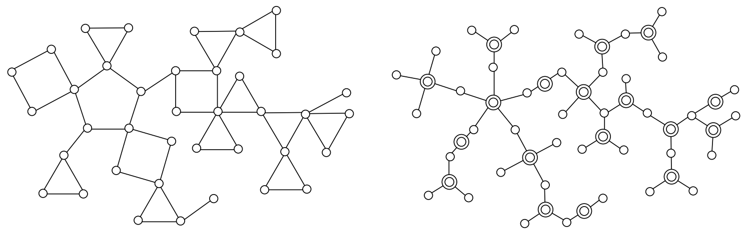

Figure 2: A graph and its block tree.

Apply the following process to T, starting with $X' := X$ and $Y' := Y$. If T has a leaf v which is neither in X' nor in Y', remove v from T. If T has a leaf v which is in X' but not in Y', include in X' the unique neighbor of v in T and remove v from X' and from T. If T has a leaf v which is in Y' but not in X', include in Y' the unique neighbor of v in T and remove v from Y' and from T. Repeat this until all leaves of T are both in X' and in Y'. If there are three or more leaves in (the remaining) T, then there is an XY-obstruction of type (o2) or (o3) in F. Indeed, if T has three or more leaves, it has a vertex of degree at least three whose removal leaves each of the three leaves in different components. If this vertex is in $\mathcal{B}$, one can get an XY-obstruction of type (o2) in F. Otherwise, one can get an XY-obstruction of type (o3). If T has less than three leaves, then T is a path. Test if some non-trivial block that appears in T plays the role of C in an XY-obstruction of type either (o1) or (o2). If no XY-obstruction is detected, T describes exactly the basis of an XY-path in F.

Theorem 3.1 (Thomas [12]) *If F is a frame of a connected graph and X and Y are subsets of V_F, then either there exists an XY-path or an XY-obstruction in F.* ■

4. LEC planarity testing method

One of the ingredients of LEC method is a certain ordering $v_1, v_2, \ldots, v_n$ of the vertices of the given graph G such that, for $i = 1, \ldots, n$, the induced subgraphs $G[\{v_1, \ldots, v_i\}]$ and $G[\{v_{i+1}, \ldots, v_n\}]$ are connected. Equivalently, G is connected and, for $i = 2, \ldots, n-1$, vertex v_i is adjacent to v_j and v_k for some j and k such that $1 \leq j < i < k \leq n$. A numbering of the vertices according to such an ordering is called a *LEC-numbering* of G. If the ordering is such that $v_1 v_n$ is an edge of the graph, the numbering is called an *st-numbering* [9]. One can show that every biconnected graph has a LEC-numbering.

LEC method examines the vertices of a given biconnected graph, one by one, according to a LEC-numbering. The algorithm works in iterations. Each iteration starts with an induced subgraph H of G and a frame F of H in G. At the beginning of the first iteration, H and F are empty. Roughly, each iteration tries to extend F by finding an XY-path in F. If it is not possible, the method finds an XY-obstruction and declares the graph is non-planar. The correctness of the method implies the following classical theorem.

[1]The leaves in V_H make the definition slightly different than the usual.

Theorem 4.1 (Kuratowski) *A graph is planar if and only if it has no subdivision of $K_{3,3}$ or K_5.* ■

Three of the algorithms mentioned in the introduction are very clever linear-time implementations of LEC method. BL use an st-numbering instead of an arbitrary LEC-numbering of the vertices and use a PQ-tree to store F. SH use a DFS-numbering and a PC-tree to store F. BM also use a DFS-numbering and use still another data structure to store F. One can use the previous description easily to design a quadratic implementation of LEC method.

5. Experimental study

The main purpose of this study was to confirm the linear-time behavior of our implementation and to acquire a deeper understanding of SH algorithm. Our experimental study extends the one presented in LEDA including our implementation of SH algorithm made on the LEDA platform and an implementation of BM algorithm developed in `C`. We performed all empirical tests used in LEDA to compare HT and BL implementations [10]. The experimental environment was a PC running GNU/Linux (RedHat 7.1) on a Celeron 700MHz with 256MB of RAM. The compiler was the `gcc 2.96` with options `-DLEDA_CHECKING_OFF -O`.

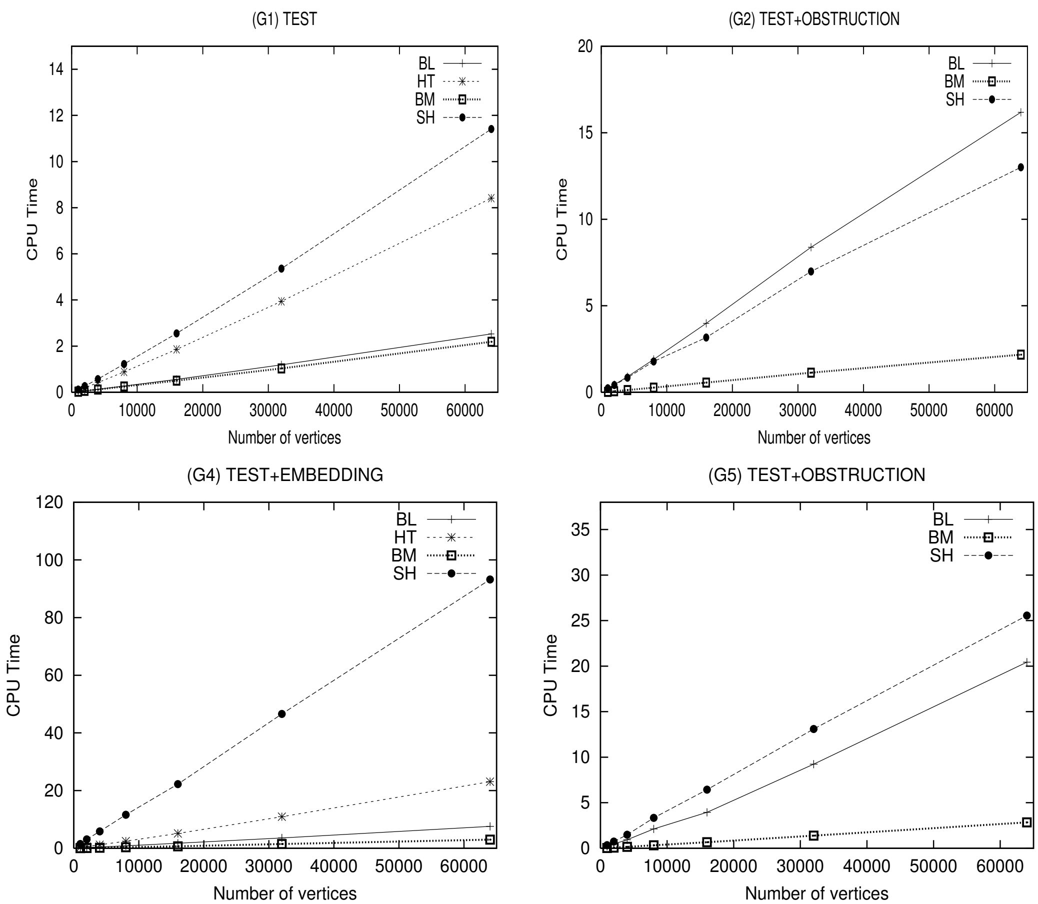

Figure 3: Empirical results comparing SH, HT, BL, and BM implementations.

Figure 3 shows the average CPU time of each implementation on (a) random planar graphs for only testing planarity, (b) random graphs with a $K_{3,3}$ for testing and finding an obstruction (HT is not included in this table, by the reason mentioned above), (c) random maximal planar graphs for testing and building an embedding, and (d) random maximal planar graphs plus a random edge connecting two non-adjacent vertices, for testing and finding an obstruction (again, HT excluded). We believe the results shown in the table are initial and still not conclusive because our implementation is yet a prototype. (Also, in our opinion, it is not fair to compare LEDA implementations with C implementations.)

Our current understanding of SH algorithm makes us believe that we can design a new implementation which would run considerably faster. Our belief comes, first, from the fact that our current code was developed to solve the planarity testing only, and was later on modified to also produce a certificate for its answer to the planarity test. Building an implementation from the start thinking about the test and the certificate would be the right way, we believe, to have a more efficient code. Second, during the adaptation to build the certificate (specially the embedding when the input is planar) made us notice several details in the way the implementation of the test was done that could be improved. Even though, we decide to go forward with the implementation of the complete algorithm (test plus certificate) so that we could understand it completely before rewriting it from scratch. It is our intention to reimplement SH algorithm from scratch.

References

[1] L. Auslander and S.V. Parter. On imbedding graphs in the plane. *J. of Mathematics and Mechanics*, 10:517–523, 1961.

[2] K.S. Booth and G.S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ–tree algorithms. *J. of Comp. and Syst. Sciences*, 13:335–379, 1976.

[3] J. Boyer, C. G. Fernandes, W. L. Hsu, A. Noma, and J. C. Pina. Correcting and implementing the PC-tree planarity algorithm. Submetido, 2003.

[4] J. Boyer, C. G. Fernandes, A. Noma, and J. C. Pina. Lempel, Even, and Cederbaum planarity method. In *Proc. of the III Workshop on Efficient and Experimental Algorithms*, 2004.

[5] J.M. Boyer and W. Myrvold. On the cutting edge: Simplified $O(n)$ planarity by edge addition. Preprint, 29pp.

[6] S. Even and R.E. Tarjan. Computing an st-numbering. *Theor. Comp. Science*, 2:339–344, 1976.

[7] A.J. Goldstein. An efficient and constructive algorithm for testing whether a graph can be embedded in a plane. In *Graph and Combinatorics Conf.*, 1963.

[8] J. Hopcroft and R. Tarjan. Efficient planarity testing. *J. of the ACM*, 21(4):549–568, 1974.

[9] A. Lempel, S. Even, and I. Cederbaum. An algorithm for planarity testing of graphs. In P. Rosenstiehl, editor, *Theory of Graphs*, pages 215–232, New York, 1967. Gordon and Breach.

[10] A. Noma. Análise experimental de algoritmos de planaridade (in Portuguese). Master's thesis, Universidade de São Paulo, 2003.

[11] W.K. Shih and W.L. Hsu. A simple test for planar graphs. In *Proc. of the International Workshop on Discrete Math. and Algorithms*, pages 110–122. University of Hong Kong, 1993.

[12] R. Thomas. Planarity in linear time. `http://www.math.gatech.edu/~thomas/`, 1997.