Theoretical Computer Science 575 (2015) 17–32

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Some results on point visibility graphs ☆

Subir Kumar Ghosh, Bodhayan Roy *

School of Technology & Computer Science, Tata Institute of Fundamental Research, Mumbai 400005, India

A R T I C L E   I N F O

Article history:
Received 7 April 2014
Received in revised form 25 September 2014
Accepted 28 October 2014
Available online 31 October 2014

Keywords:
Necessary conditions
Recognizing point visibility graphs
Characterizing point visibility graphs
Visibility embedding
Planar graph
Computational geometry
PSPACE

A B S T R A C T

In this paper, we present three necessary conditions for recognizing point visibility graphs. We show that this recognition problem lies in PSPACE. We state new properties of point visibility graphs along with some known properties that are important in understanding point visibility graphs. For planar point visibility graphs, we present a complete characterization which leads to a linear time recognition and reconstruction algorithm.

1. Introduction

The visibility graph is a fundamental structure studied in the field of computational geometry and geometric graph theory [5,9]. Some of the early applications of visibility graphs included computing Euclidean shortest paths in the presence of obstacles [14] and decomposing two-dimensional shapes into clusters [18]. Here, we consider problems from visibility graph theory.

Let $P = \{p_1, p_2, \dots, p_n\}$ be a set of points in the plane (see Fig. 1). We say that two points p_i and p_j of P are *mutually visible* if the line segment $p_i p_j$ does not contain or pass through any other point of P. In other words, p_i and p_j are visible if $P \cap p_i p_j = \{p_i, p_j\}$. If two vertices are not visible, they are called an *invisible pair*. For example, in Fig. 1(c), p_1 and p_5 form a visible pair whereas p_1 and p_3 form an invisible pair. If a point $p_k \in P$ lies on the segment $p_i p_j$ connecting two points p_i and p_j in P, we say that p_k blocks the visibility between p_i and p_j, and p_k is called a *blocker* in P. For example in Fig. 1(c), p_5 blocks the visibility between p_1 and p_3 as p_5 lies on the segment $p_1 p_3$. The *visibility graph* (also called the *point visibility graph* (PVG)) G of P is defined by associating a vertex v_i with each point p_i of P such that (v_i, v_j) is an undirected edge of G if and only if p_i and p_j are mutually visible (see Fig. 1(a)). Observe that if no three points of P are collinear, then G is a complete graph as each pair of points in P is visible since there is no blocker in P. Sometimes the visibility graph is drawn directly on the point set, as shown in Figs. 1(b) and 1(c), which is referred to as a *visibility embedding* of G.

☆ An extended abstract of this paper appeared in the Proceedings of the Eighth International Workshop on Algorithms and Computation, pp. 163–175, 2014 [11].

* Corresponding author.
E-mail addresses: ghosh@tifr.res.in (S.K. Ghosh), bodhayan@tifr.res.in (B. Roy).

http://dx.doi.org/10.1016/j.tcs.2014.10.042

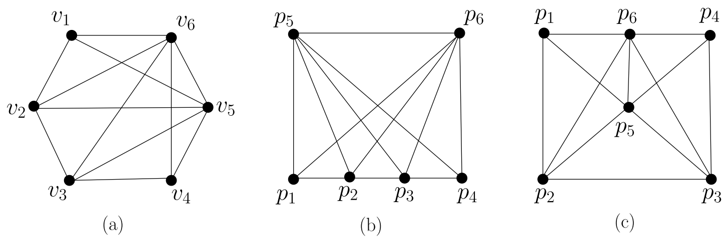

Fig. 1. (a) A point visibility graph with (v_1, v_2, v_3, v_4) as a CSP. (b) A visibility embedding of the point visibility graph where (p_1, p_2, p_3, p_4) is a GSP. (c) A visibility embedding of the point visibility graph where (p_1, p_2, p_3, p_4) is not a GSP.

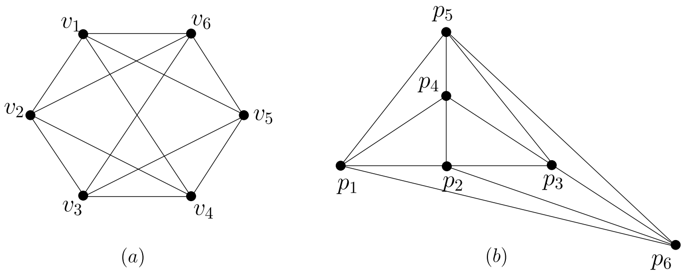

Fig. 2. (a) A planar graph G. (b) A planar visibility embedding of G.

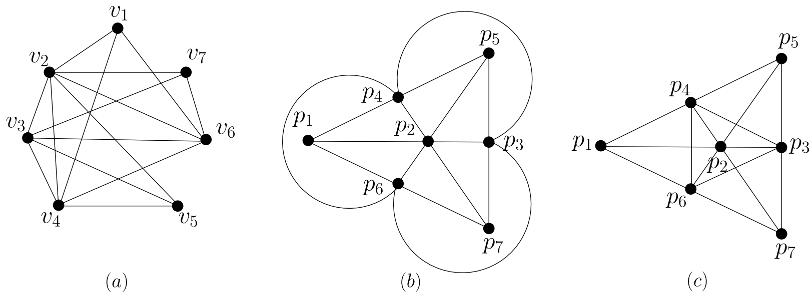

Fig. 3. (a) A planar graph G. (b) A planar embedding of G. (c) A non-planar visibility embedding of G.

Given a point set P, the visibility graph G of P can be computed as follows. For each point p_i of P, the points of P are sorted in angular order around p_i. If two points p_j and p_k are consecutive in the sorted order, check whether p_i, p_j and p_k are collinear points. By traversing the sorted order, all points of P, that are not visible from p_i, can be identified in $O(n \log n)$ time. Hence, G can be computed from P in $O(n^2 \log n)$ time. Using the result of Chazelle et al. [4] or Edelsbrunner et al. [7], the time complexity of the algorithm can be improved to $O(n^2)$ by computing sorted angular orders for all points together in $O(n^2)$ time.

Consider the opposite problem of determining if there is a set of points P whose visibility graph is the given graph G. This problem is called the visibility graph *recognition* problem: Given a graph G in adjacency matrix form, determine whether G is the visibility graph of a set of points P in the plane [10]. Identifying the set of properties satisfied by all visibility graphs is called the visibility graph *characterization* problem. The problem of actually drawing one such set of points P whose visibility graph is the given graph G, is called the visibility graph *reconstruction* problem.

In Section 2.1, we present three necessary conditions for the recognition problem. Though the first necessary condition can be tested in $O(n^3)$ time, it is not clear whether the second necessary and third conditions can be tested in polynomial time. In Section 2.2, we present a linear time algorithm for identifying a Hamiltonian Cycle of a point visibility graph from a given visibility embedding. In Section 2.3, we establish a property on the connectivity of point visibility graphs.

We address some complexity aspects of point visibility graphs. In Section 3.1, we show that the recognition problem lies in PSPACE. In Section 3.2, we show that the problems of Vertex Cover, Independent Set and Maximum Clique remain NP-hard on point visibility graphs.

If a given graph G is planar, there can be three cases: (i) G has a planar visibility embedding (Fig. 2), (ii) G admits a visibility embedding, but no visibility embedding of G is planar (Fig. 3), and (iii) G does not have any visibility embedding (Fig. 4). Case (i) has been characterized by Eppstein [6] by presenting four infinite families of G and one particular graph. In order to characterize graphs in Case (i) and Case (ii), we show that two infinite families and five particular graphs are required in addition to graphs for Case (i). Using this characterization, we present an $O(n)$ algorithm for recognizing and