Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 17, no. 2, pp. 81–102 (2013)
DOI: 10.7155/jgaa.00286

Triangle-Free Outerplanar 3-Graphs are Pairwise Compatibility Graphs

Sammi Abida Salma[1] *Md. Saidur Rahman*[1] *Md. Iqbal Hossain*[1]

[1]Graph Drawing and Information Visualization Laboratory,
Department of Computer Science and Engineering,
Bangladesh University of Engineering and Technology(BUET),
Dhaka-1000, Bangladesh.

Abstract

A graph $G = (V, E)$ is called a *pairwise compatibility graph* (PCG) if there exists an edge-weighted tree T and two non-negative real numbers d_{min} and d_{max} such that each vertex $u' \in V$ corresponds to a leaf u of T and there is an edge $(u', v') \in E$ if and only if $d_{min} \leq d_T(u, v) \leq d_{max}$ in T. Here, $d_T(u, v)$ denotes the distance between u and v in T, which is the sum of the weights of the edges on the path from u to v. It is known that not all graphs are PCGs. Thus it is interesting to know which classes of graphs are PCGs. In this paper we show that triangle-free outerplanar graphs with the maximum degree 3 are PCGs.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
April 2012	July 2012	December 2012	December 2012	December 2012

Published: January 2013

Article type: Regular paper
Communicated by: S.-i. Nakano

E-mail addresses: sammiq@gmail.com (Sammi Abida Salma) saidurrahman@cse.buet.ac.bd (Md. Saidur Rahman) mdiqbalhossain@cse.buet.ac.bd (Md. Iqbal Hossain)

1 Introduction

Let T be an edge-weighted tree and let d_{min} and d_{max} be two non-negative real numbers such that $d_{min} \leq d_{max}$. A *pairwise compatibility graph(PCG) of T for d_{min} and d_{max}* is a graph $G = (V, E)$, where each vertex $u' \in V$ represents a leaf u of T and there is an edge $(u', v') \in E$ if and only if the distance between u and v in T lies within the range from d_{min} to d_{max}. T is called the *pairwise compatibility tree* of G. We denote a pairwise compatibility graph of T for d_{min} and d_{max} by $PCG(T, d_{min}, d_{max})$. Figure 1(b) depicts an edge-weighted tree T and Fig. 1(a) depicts a pairwise compatibility graph G of T for $d_{min} = 4$ and $d_{max} = 7$; there is an edge between a' and b' in G since in T the distance between a and b is six, which is within the range; but G does not contain the edge (a', c') since in T the distance between a and c is eight, which is larger than seven; also there is an edge between b' and c' in G since in T the distance between b and c is four, but G does not contain the edge (b', d') since in T the distance between b and d is three, which is smaller than four. It is quite apparent

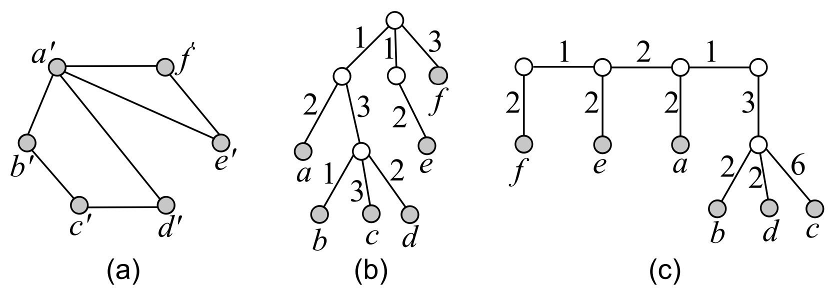

Figure 1: (a) A pairwise compatibility graph G, (b) an edge-weighted tree T and (c) an edge-weighted tree T_2.

that a single edge-weighted tree may have many pairwise compatibility graphs for different values of d_{min} and d_{max}. Likewise, a single pairwise compatibility graph may have many trees of different topologies as its pairwise compatibility trees. For example, the graph in Fig. 1(a) is a PCG of the tree in Fig. 1(b) for $d_{min} = 4$ and $d_{max} = 7$, and it is also a PCG of the tree in Fig. 1(c) for $d_{min} = 5$ and $d_{max} = 8$. Thus, the pairwise compatibility concept concerns two bounds d_{min} and d_{max}. The special case where $d_{min} = 0$ reduces PCGs to graph classes, namely "leaf power graphs", investigated by Kolen [15] and later by Brandstadt *et al.* [2, 3]. This interesting case where $d_{min} = 0$ leads to graphs that are a subclass of strongly chordal graphs [8] which have many practical applications. A graph $G(V, E)$ is a *leaf power graph* (LPG) iff there exists a tree T and a nonnegetive number d_{max} such that for an edge (u', v') in E and their corresponding leaves u, v in T we have $d_T(u, v) \leq d_{max}$ [17]. A lot of works has been done on this class (LPG) of graphs by Kennedy *et al.* at [14], again by Brandstadt *et al.* at [5, 4, 1], also by Fellows *et al.* at [9]. However the complete description of leaf power graphs is still unknown. This class LPG is a subclass of PCG. In [7] the authors have introduced another subclass of PCG, namely "mLPG", by concerning only the minimum distance constraint and showed the relations between PCG, LPG and mLPG. In *mLPG* (we set $d_{max} = +\infty$), there is an edge in E if and only if the corresponding leaves are at a distance greater than d_{min} in the tree.

There are two fundamental problems in the realm of pairwise compatibility graphs. One is the tree construction problem and the other is the pairwise compatibility graph recognition problem. Given a PCG G, the tree construction problem asks to construct an edge-weighted tree T, such that G is a pairwise compatibility graph of T for suitable d_{min} and d_{max}. The second problem, pairwise compatibility graph recognition problem, seeks the answer whether or not a given graph is a PCG.

Pairwise compatibility graphs have their applications in reconstructing evolutionary relationships among organisms from biological data (also called phylogeny) [12, p.196-200] [16, p.189-199]. The phylogeny reconstruction problem is known to be NP-hard [10, 11]. Phylogenetic relationships are usually represented as trees known as the phylogenetic trees. Dealing with the problem of collecting leaf samples from large phylogenetic trees, Kearney *et al.* introduced the concept of pairwise compatibility graphs [13]. Furthermore, in that paper, the proponents of PCGs have shown that "the clique problem", a well known NP-complete problem, is polynomially solvable for pairwise compatibility graphs if the pairwise compatibility tree construction problem can be solved in polynomial time.

Since their inception, several interesting problems have been raised in pairwise compatibility graphs concept , and hitherto most of these problems have remained unsolved. Among the others, identifying different graph classes as pairwise compatibility graphs is an important concern. Seeing the exponentially increasing number of possible tree topologies for large graphs, the proponents of PCGs conjectured that all undirected graphs are PCGs [13]. Yanhaona *et al.* refute the conjecture by showing that not all graphs are PCGs [22]. Phillips has shown that every graph of five vertices or less is a PCG [18] and also very recently it is proved that every graph of seven vertices or less is a PCG [6]. It has also been shown that all cycles, single chord cycles, cactus graph, tree power graphs, Steiner k-power and phylogenetic k-power graphs, some particular subclasses of bipartite graphs, some particular subclasses of split matrogenetic graphs are PCGs [22, 23, 7]. In this paper we show that trees, ladder graphs, outer subdivision of ladder graphs and triangle-free outerplanar 3-graphs are PCGs. We also provide algorithms for constructing pairwise compatibility trees for graphs of these classes.

The rest of the paper is organized as follows. Section 2 gives some of the definitions along with some trivial results on trees and ladder graphs. Section 3 deals with outer subdivisions of ladder graphs which is a subclass of triangle free outerplanar 3-graphs. In Section 4 we show that triangle free outerplanar 3-graphs are pairwise compatibility graphs. Finally, Section 5 concludes our paper with discussions. A preliminary version of this paper has been presented at [20].

2 Preliminaries

In this section we define some terms that we have used in this paper.
Let $G = (V, E)$ be a simple connected graph with vertex set V and edge set E. The sets of vertices and edges of G are denoted by $V(G)$ and $E(G)$, respectively. An edge between two vertices u and v of G is denoted by (u, v). Two vertices u and v are *adjacent* and called *neighbors* if $(u, v) \in E$; the edge (u, v) is then said to be *incident* to vertices u and v. The *degree* of a vertex v in G is the number of edges incident to it. The maximum degree of a graph G is the maximum degree of its vertices. We call a graph k-graph if the maximum degree of that graph is k. A *path* $P_{uv} = w_0, w_1, \cdots, w_n$ is a sequence of distinct vertices in V such that $u = w_0, v = w_n$ and $(w_{i-1}, w_i) \in E$ for every $1 \leq i \leq n$. A *subgraph* of a graph $G = (V, E)$ is a graph $G' = (V', E')$ such that $V' \subseteq V$ and $E' \subseteq E$; we then write $G' \subseteq G$. The *connectivity* $\kappa(G)$ of a graph G is the minimum number of vertices whose removal results in a disconnected graph or a single vertex graph. G is called a *k-connected* graph if $\kappa(G) \geq k$. We call a vertex of G a *cut vertex* if its removal results in a disconnected or single-vertex graph. A biconnected graph of n vertices is a *ladder* if it consists of two distinct paths of the same length $(u_1, u_2, \cdots, u_{\frac{n}{2}})$ and $(v_1, v_2, \cdots, v_{\frac{n}{2}})$ plus the edges (u_i, v_i) $(i = 1, 2, \cdots, \frac{n}{2})$.
A *cycle* of G is a sequence of distinct vertices starting and ending at the same vertex such that two vertices are adjacent if they appear consecutively in the sequence. A *tree* T is a connected graph with no cycle. Vertices of degree one in T are called *leaves* and the others are *internal nodes*. A tree T is *weighted* if each edge is assigned a number as the weight of the edge. The weight of an edge (u, v) is denoted by $W(u, v)$. The *distance between two vertices u and v* in T, denoted by $d_T(u, v)$, is the sum of the weights of the edges on P_{uv}. A *caterpillar* graph is a tree such that if all leaves and their incident edges are removed, the remainder of the graph forms a single path. The path is called the *spine* of the caterpillar.

A graph is *planar* if it can be embedded in the plane so that no two edges intersect geometrically except at a vertex to which both of them are incident. A *plane graph* is a planar graph with a fixed embedding. A plane graph divides the plane into connected regions called *faces*. The unbounded region is called the *unbounded* or *outer face* of G. The other faces are called *bounded* or *internal faces*. Any face with exactly 3 edges is called a *triangulated face*. A *dual graph* of a plane graph G is a graph which has a vertex for each face of G, and an edge for each edge in G joining two neighboring faces. A *weak dual* of a plane graph G is the subgraph of the dual graph of G whose vertices correspond to the bounded faces of G. A graph is *outerplanar* if it has a planar embedding where all vertices are on the outer face. Throughout the paper by an outerplanar graph we mean an outerplanar embedding of an outerplanar graph, that is, an outerplane graph. A weak dual of a biconnected outerplanar graph is a tree. A *triangle-free outerplanar graph* is an outerplanar graph containing no triangulated faces. *Subdividing an edge (u,v)* of a graph G is the operation of deleting the edge (u, v) and adding a path $u(= w_0), w_1, w_2, \cdots, w_k, v(w_{k+1})$

through new vertices $w_1, w_2, \cdots, w_k, k \geq 1$, of degree two. A graph G' is said to be a *subdivision* of a graph G if G' is obtained from G by subdividing some of the edges of G. An *outer subdivision* of a graph G is a subdivision of G obtained by subdividing some edges on the outer face of G.

We are now going to present some elementary results on pairwise compatibility graphs. The following theorem gives a trivial result on trees.

Theorem 1 *Every tree is a pairwise compatibility graph.*

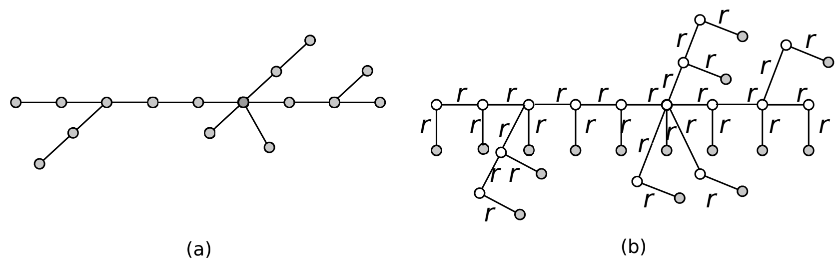

Figure 2: (a) A tree T and (b) a pairwise compatibility tree T'' of T.

Proof: We give a constructive proof. Let T be a tree as illustrated in Fig. 2(a). We take a replica T' of T. We construct a tree T'' from T' by introducing a new (pendant) vertex v'' for each $v' \in T'$ such that $(v', v'') \in E(T'')$. Then we assign weight r to all the edges of T'' as illustrated in Fig. 2(b). The leaves of T'' represent the vertices of T. Clearly, T'' is a pairwise compatibility tree of T with $d_{min} = d_{max} = 3r$. □

The construction in the proof of Theorem 1 can be done in linear time. We now present our result on ladder graphs as in the following theorem.

Theorem 2 *Every ladder graph is a pairwise compatibility graph.* [1]

Proof: Let $G = (V, E)$ be a ladder graph with $2n$ vertices where $V = \{v'_1, v'_2, v'_3, \cdots, v'_n, u'_1, u'_2, u'_3, \cdots, u'_n\}$ such that $\{v'_1, v'_2, v'_3, \cdots, v'_n\}$ and $\{u'_1, u'_2, u'_3, \cdots, u'_n\}$ are two distinct paths and for $1 \leq i \leq n$, $(u'_i, v'_i) \in E$, as illustrated in Fig. 3(a). We construct a caterpillar T where the leaves $v_1, v_2, v_3, \cdots, v_n$, $u_1, u_2, u_3, \cdots, u_n$ represent the vertices $v'_1, v'_2, v'_3, \cdots, v'_n$, $u'_1, u'_2, u'_3, \cdots, u'_n$ of G, respectively, as illustrated in Fig. 3(b). Let p_i be the vertex adjacent to v_i and q_i be the vertex adjacent to u_i in the caterpillar for $1 \leq i \leq n$. We now assign weights for the edges of T as follows. For $1 \leq i \leq n$, we assign $W(v_i, p_i) = r + 1, W(p_i, q_i) = r - 1, W(q_i, u_i) = r$ and for $1 \leq i < n$, we assign $W(q_i, p_{i+1}) = r - 2$. We next show $G = PCG(T, d_{min}, d_{max})$ with $d_{min} = 3r$, $d_{max} = 4r - 1$ and $r \geq 3$.

[1]This result was presented at [19].

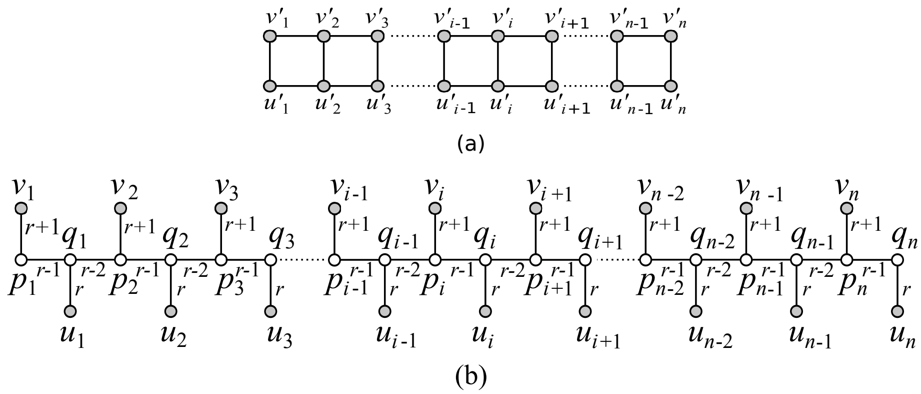

Figure 3: (a) A ladder graph G and (b) a pairwise compatibility tree T of G.

We first consider the case where $2 \leq i < n$. Here $d_T(v_i, u_i) = 3r = d_{min}$. Since this distance is within the range, v'_i and u'_i are adjacent in G. Similarly, since $d_T(v_{i-1}, v_i) = d_T(v_i, v_{i+1}) = (r+1) + (r-2) + (r-1) + (r+1) = 4r - 1 = d_{max}$, the vertices v'_{i-1} and v'_i are adjacent in G. Since $d_T(u_{i-1}, u_i) = d_T(u_i, u_{i+1}) = r + (r-2) + (r-1) + r = 4r - 3 = (4r-1) - 2 = d_{max} - 2 = 3r + (r-3) = d_{min} + (r-3)$, for $r \geq 3, d_{min} \leq d_T(u_{i-1}, u_i), d_T(u_i, u_{i+1}) < d_{max}$ and hence u'_{i-1} and u'_i are adjacent in G. Since $d_T(v_i, u_{i-1}) = (r+1) + (r-2) + r = 3r - 1 < d_{min}$, v'_i and u'_{i-1} are not adjacent in G. The vertices v'_i and u'_{i+1} are not adjacent because $d_T(v_i, u_{i+1}) = (r+1) + (r-1) + (r-2) + (r-1) + r = 5r - 3 = (4r-1) + (r-2) = d_{max} + (r-2) > d_{max}$. Similarly, the vertices v'_{i-1} and u'_i are not adjacent because $d_T(v_{i-1}, u_i) = (r+1) + (r-1) + (r-2) + (r-1) + r = 5r - 3 = (4r-1) + (r-2) = d_{max} + (r-2) > d_{max}$. Again v'_{i+1} and u'_i are not adjacent, since $d_T(v_{i+1}, u_i) = (r+1) + (r-2) + r = 3r - 1 < d_{min}$.

One can easily verify the other two cases, where $i = 1$ and $i = n$.

Hence, by definition, $G = PCG(T, d_{min}, d_{max})$. □

Based on the proof of Theorem 2, one can obtain an $O(n)$ time algorithm for constructing a pairwise compatibility tree of a ladder graph of n vertices.

3 Outer subdivisions of ladder graphs and PCGs

In an outer subdivision of a ladder graph every vertex lies on the boundary of the outer face of the graph. In this section, graphs of this class will be shown to be a PCG by constructing the pairwise compatibility trees for these graphs. The main idea is to decompose a graph of this class into cycles, then construct pairwise compatibility trees for these cycles as caterpillars and finally merge the caterpillars to get the desired pairwise compatibility tree. Theorem 3 states the main result of this section.

Theorem 3 *Outer subdivisions of ladder graphs are pairwise compatibility graphs.*

To prove the claim of Theorem 3 we need the following lemmas. Our first lemma finds a pairwise compatibility tree of a cycle. Although the authors of [23] gave two algorithms for finding pairwise compatibility trees of given cycle by considering odd cycle and even cycle separately, we give a generalized construction in the proof of Lemma 1 for all cycles.

Lemma 1 *Every cycle is a PCG which has a pairwise compatibility tree as a caterpillar.*

Proof: Let C be a cycle with n vertices $v'_1, v'_2, v'_3, \cdots, v'_n$ where (v'_i, v'_j) are adjacent for $i < n, j \leq n, j = i+1$ and $(v'_1\ , v'_n)$ are also adjacent. We make a caterpillar T for this cycle C such that the leaves $v_1, v_2, v_3, \cdots, v_{n-1}$ represents the vertices $v'_1, v'_2, v'_3, \cdots, v'_{n-1}$ of C where path $P_{v_i v_j}$ in T for $j = i+1$ contains three edges including an edge on the spine as illustrated in Fig. 4. We assign weight d to each edge on the spine and weight w to each of the other edges. Then the distance between any two leaves in the caterpillar is $d_T(v_i, v_j) = w + (j-i)d + w = 2w + (j-i)d$. Let $u_1, u_2, u_3, \cdots, u_{n-1}$ be the vertices on the spine of the caterpillar that are adjacent to the leaves $v_1, v_2, v_3, \cdots, v_{n-1}$, respectively. If n is odd, we put a vertex u_n in the middle of the path $P_{u_1 u_{n-1}}$ on the spine and place the last vertex v_n as a leaf which is adjacent to the vertex u_n with an edge of weight w_n, as illustrated in Fig. 4. If n is even, we use the vertex $u_{\frac{n}{2}}$ as u_n which is in the middle of the path $P_{u_1 u_{n-1}}$ on the spine and place the last vertex v_n as a leaf which is adjacent to the vertex $u_{\frac{n}{2}}$ with an edge of weight w_n, as illustrated in Fig. 5.

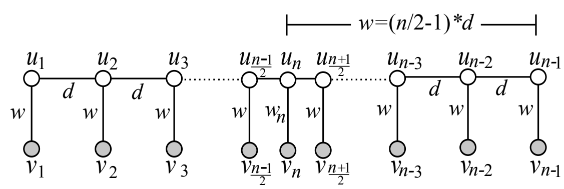

Figure 4: A caterpillar which is a pairwise compatibility tree of a cycle C of odd number of vertices.

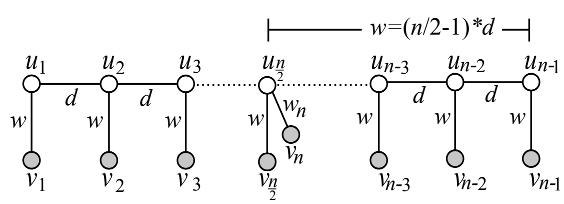

Figure 5: A caterpillar which is a pairwise compatibility tree of a cycle C of even number of vertices.

Now let us keep the relations between the weights w and d such that w be the distance between vertices u_{n-1} and u_n, i.e. $w = (n/2 - 1) * d$ (see Figures 4 and 5). Let us assign $w_n = 2d - r$ where $r \ll d$, $d_{min} = 2w + d$ and $d_{max} = 2w + 2d - r = d_{min} + d - r$. One can easily verify that the distance between the last vertex and any other vertex that is not adjacent to the last vertex is less than d_{min}. On the other hand, the distance between any other pair of non-adjacent vertices is greater than d_{max} and $C = PCG(T, d_{min}, d_{max})$. $\square$

The outer subdivisions of the ladder graph G can be decomposed into several cycles and we can merge corresponding compatibility tree to get PCG of G. We cannot merge the caterpillars of different cycles unless they have the same d_{min} and d_{max}. However, the following lemma focuses on constructing caterpillars of different cycles with the same d_{min} and d_{max}.

Lemma 2 *Let C be a cycle of n vertices. Assume that C has a pairwise compatibility tree T for some d_{min} and d_{max}. Let C' be a cycle of $n' \leq n$ vertices. Then C' has a pairwise compatibility tree T' for the same d_{min} and d_{max}.*

Proof: According to Lemma 1, Cycle C has a pairwise compatibility tree as a caterpillar T where every edge on the spine has weight d, the edge with the last vertex has weight $w_n = 2d - r$ and each of the other edges has weight w where $w = d(n/2 - 1)$, $d_{min} = 2w + d$ and $d_{max} = 2w + 2d - r$.

Let T be the pairwise compatibility tree of cycle C of n vertices. Let d be the assigned weight to the each edge of spine in T. So according to Lemma 1 $w = d(n/2 - 1)$ and the weight of the edge with the last vertex is $w_n = 2d - r$ where $d_{min} = 2w + d$ and $d_{max} = 2w + 2d - r$. The distance $d_T(v_n, v_1) = d_T(v_n, v_{n-1}) = d_{max} = 2w + 2d - r$.

Let T' be the pairwise compatibility tree of C'. We set weight d to each edge of spine and $w = d(n/2 - 1)$ to each of the other edges in T'. One can easily observe that, the distance between any two vertices of $v_1, v_2, \ldots, v_{n'-1}$ in T' hold the relation for being $C' = PCG(T', d_{min}, d_{max})$. Since the distance of $d_{T'}(v_{n'}, v_1)$ or $d_{T'}(v_{n'}, v_{n'-1})$ must be equal to d_{max}, we set $w_{n'} = d_{max} - d_{T'}(v_1, u_{n'})$. Since $d_{max} = 2w + 2d - r$, $d_{T'}(u_1, u_{n'}) = d(n'/2 - 1)$ and $w = d(n/2 - 1)$, $w_{n'} = d(n - n')/2 + 2d - r$.

The weight chosen above satisfies the condition that $d_{T'}(v_{n'}, v_2) < d_{min}$ and $d_{T'}(v_{n'}, v_{n'-2}) < d_{min}$. This implies that the distance between $v_{n'}$ and any vertex which is not adjacent to $v_{n'}$ is less than d_{min}.

$\square$

Once we have the caterpillars of different cycles with the same $d_{min} = 2w+d$ and $d_{max} = 2w + 2d - r$, we have to proceed to merge the caterpillars. Before merging we adjust some weights of all the caterpillars; we set $w'_n = w_n - (d-r)$; and for $1 \leq i \leq (n-1)$, $d_T(u_i, v_i) = w + b$, where $b \ll r \ll d$ and $b \ll (d-r)$, as illustrated in Fig. 6. Then $d_T(v_n, v_1) = d_T(v_n, v_{n-1}) = 2w+d+b = (w+b)+d+w$ and $d_T(v_n, u_2) = d_T(v_n, u_{n-2}) = w$. One can easily observe that, the distance between any two vertices of $v_1, v_2, \ldots, v_{n-1}$ in T hold the relation for being $C = PCG(T, d_{min}, d_{max})$ when we set $d_T(u_i, v_i) = w + b$. The distance $d_T(v_n, v_1) =$

$d_T(v_n, v_{n-1}) = d_{max} - (d-r) + b = (2w+2d-r) - (d-r) + b = 2w+d+b$ which is greater than d_{min} and less than d_{max} and again $d_T(v_n, v_2) = d_T(v_n, v_{n-2}) = d_T(v_n, v_1) - (w+b) - d + (w+b) = (2w+d+b) - d = 2w+b < d_{min}$. So the relations between the leaves of the caterpillars remain unchanged after adjusting the weights. Now we observe that $d_T(v_n, u_2) = d_T(v_n, u_{n-2}) = w$.

The following lemma focuses on merging the caterpillars.

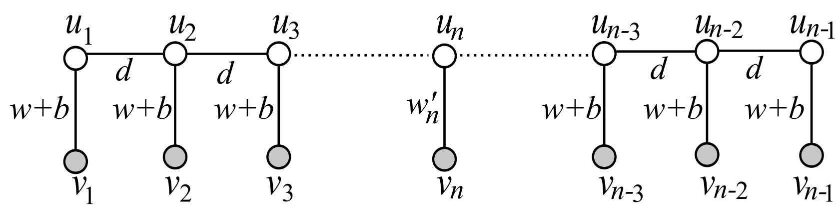

Figure 6: Caterpillar of a cycle after adjusting weights.

Lemma 3 *Let G be a graph such that $G = C_1 \cup C_2$ where C_1 and C_2 are two cycles having exactly one common edge. Let T_1 and T_2 be two caterpillars such that T_1 and T_2 are pairwise compatibility trees of C_1 and C_2, respectively, with the same d_{min} and d_{max}. Then, a pairwise compatibility tree T for G can be obtained for the same d_{min} and d_{max} by merging T_1 and T_2.*

Proof: Let G be a graph such that $G = C_1 \cup C_2$ where C_1 and C_2 are two cycles having exactly one common edge. Hence C_1 and C_2 have exactly two common vertices. Let cycles C_1 and C_2 contain s and t vertices respectively. Let T_1 and T_2 be two caterpillars such that T_1 and T_2 are pairwise compatibility trees of C_1 and C_2, respectively, with the same d_{min} and d_{max}, as illustrated in Fig. 7. We have to create pairwise compatibility tree T of G by using T_1 and T_2 for the same d_{min} and d_{max}.

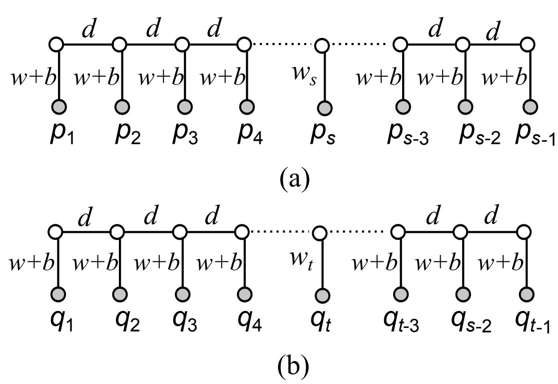

Figure 7: (a) A pairwise compatibility tree T_1 of cycle C_1 and (b) a pairwise compatibility tree T_2 of cycle C_2.

Without loss of generality we can consider that (q'_1, q'_2) of C_2 is the common edge (p'_x, p'_y) of C_1 where $p'_y = p'_{x+1}$. Since C_1 is a cycle, there are two paths between p'_1 and p'_y. We call the path from p'_1 to p'_y the *reference path* on which neither p'_2 nor p'_x lies. The reference paths are shown by thick lines in Figs. 8(a), 9(a) and 10(a). Depending on the number of vertices on the reference path, we have three cases to consider.

Case 1: The reference path contains exactly three vertices. (See Fig. 8(a).) In this case, in C_1, p'_x is p'_{s-2} and p'_y is p'_{s-1}. We label the vertices of C_2 as counter clock wise as shown in the Fig. 8(a).

Let T_1 be the pairwise compatibility tree of C_1. We set $d_{T_1}(p_{s-1}, u_{s-1}) = w + 2b$ and $d_{T_1}(u_{s-1}, u_{s-2}) = d - b$. Hence the resultant $d_{T_1}(p_{s-1}, u_{s-2}) = w + b + d$ (see the Fig. 8(b)). Since, the vertex u_{s-1} is a vertex of degree 2, this weight adjustment does not affect to other relation.

Let T_2 be the pairwise compatibility tree of C_2. In T_2 we set $d_{T_2}(v_1, v_2)) = d - b$ and $d_{T_2}(v_2, q_2)) = w + 2b$. We are changing the weight at most b and $b \ll r \ll d$, so the relations of the distances ($distance > d_{max}, distance < d_{min}, d_{min} \leq distance \leq d_{max}$) between the leaves of the caterpillars remain unchanged after adjusting the weights as shown in the Fig. 8(c).

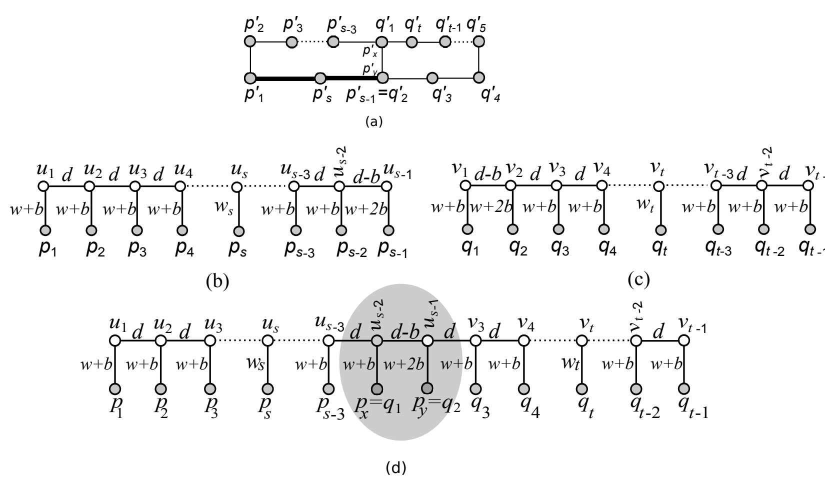

Figure 8: Pairwise compatibility tree construction for Case 1; (a) a graph G with cycles C_1 and C_2, (b) the pairwise compatibility tree T_1 of C_1, (c) the pairwise compatibility tree T_2 of C_2, and (d) the pairwise compatibility tree T of G.

Pairwise compatibility tree T for G can be obtained by merging T_1 and T_2 such that the vertices q_1,v_1, v_2, and q_2 of T_2 lie on the vertices p_{s-2}, u_{s-2}, u_{s-1} and p_{s-1} of T_1, receptively as illustrated in Fig. 8(d).

Now we show that T is a pairwise compatibility tree of G. The distance from p_s to q_1 is $w + (w + b)$ which is smaller than d_{min}, the distance from p_s to q_2 is equal to $2w + d + b$ which is greater than d_{min} and less than d_{max} and the distances from p_s to each of the leaves $q_3, q_4, ..., q_{t-1}$ are greater than d_{max}.

Again, the distance from q_t to p_y is smaller than d_{min}, the distance from q_t to p_x is equal to $2w+d$ which is equal to d_{min} and the distances from q_t to each of the leaves $p_1, p_2, p_3, ..., p_{s-3}$ are greater than d_{max}. Hence all the leaves except p_s and q_t hold the relations for being $G = PCG(T, d_{min}, d_{max})$.
Here p'_s and q'_t are not adjacent in G and
$d_T(p_s, q_t) = d_T(p_s, u_{s-1}) + d_T(u_{s-1}, q_t)$
$= d_T(p_s, p_y) - (w + 2b) + w$
$= (2w + d + b) - (w + 2b) + w$
$= 2w + d - b < d_{min}$. i.e. $d_T(p_s, q_t) < d_{min}$. Thus $G = PCG(T, d_{min}, d_{max})$.

Case 2: The reference path contains more than three vertices. (See Fig. 9(a).) In this case, in C_1, neither p'_x nor p'_y is p'_s or p'_{s-1}. Let us assume p_x, among the consecutive leaves p_x and p_y, is nearer to p_s in T_1. Let the leaf p_z be at an equal distance from p_x as p_y from p_x.

We insert a vertex u_l between u_z and u_x in T_1 such that $d_{T_1}(u_z, u_l) = d - l$ and $d_{T_1}(u_l, u_x) = l$ as shown in the Fig. 9(b). One can easily observe that adding such a vertex preserves $C_1 = PCG(T_1, d_{min}, d_{max})$.

We insert a vertex v_l between v_2 and v_3 in T_2 such that $d_{T_2}(v_3, v_l) = d$ and $d_{T_2}(v_l, v_2) = l$ where $(r \gg l \gg b)$ as shown in the Fig. 9(c). Now $d_{T_2}(v_2, v_3) = d + l$. We can easily verify that adding such a small weight on the spine does not break the relation for being $C_2 = PCG(T_2, d_{min}, d_{max})$.

Note that $d_{T_2}(v_l, v_2) = d_{T_1}(u_l, u_x) = l$, $d_{T_2}(v_2, q_2) = d_{T_1}(u_x, p_x) = w + b$, $d_{T_2}(v_1, v_2) = d_{T_1}(u_x, u_y) = d$, and $d_{T_2}(v_1, q_1) = d_{T_1}(u_y, p_y) = w + b$.

So we can easily merge T_1 and T_2 such that the vertices v_l, v_2, q_2, v_1 and q_1 of T_2 lie on the vertices on u_l, u_x, p_x, u_y and p_y of T_1, respectively, as illustrated in Fig. 9(d).

Now the distances from p_s to q_1 and from p_s to q_2 are smaller than d_{min} and the distances from p_s to each of the leaves $q_3, q_4, ..., q_{t-1}$ are either less or equal to $w - l + d + w + b$ which is less than d_{min} or greater or equal to $w - l + 2d + w + b$ which is greater than d_{max}. Again, the distance from q_t to $p_x = q_2$ is $d_T(q_t, u_l) + l + w + b = (w) + l + w + b = 2w + l + b < d_{min}$, the distance from q_t to $p_y = q_1$ is $d_T(q_t, u_l) + l + d + w + b = (w) + l + d + w + b = 2w + d + l + b$ which is within the range from d_{min} to d_{max}, the distance from q_t to p_z is $d_T(q_t, u_l) + d - l + w + b = 2w + d - (l - b) < d_{min}$ and the distances from q_t to each of the leaves $p_1, p_2, p_3, ..., p_{z-1}$ and $p_{y+1}, ..., p_{s-1}$ are greater than d_{max}. Hence all the leaves except p_s and q_t hold the relations for being
$G = PCG(T, d_{min}, d_{max})$. Again p'_s and q'_t are not adjacent in G, and $d_T(p_s, q_t) = d_T(p_s, u_l) + d_T(u_l, q_t) < (w - d - l) + (w) = 2w - (d + l) < d_{min}$. Thus $G = PCG(T, d_{min}, d_{max})$.

Case 3: The reference path contains only two vertices. (See Fig. 10(a).) In this case, p'_y is p'_s and p'_x is p'_{s-1} in C_1.

We set $d_{T_1}(u_{s-2}, u_{s-1}) = d+b$ and $d_{T_1}(u_{s-1}, p_{s-1}) = w$ in T_1 so the resultant distance is $d_{T_1}(u_{s-2}, p_{s-1}) = w + d + b$. Since u_{s-1} is a vertex of degree 2 this weight adjustment preserves $C_1 = PCG(T_1, d_{min}, d_{max})$ (see in Fig. 10(b)).

In T_2 we set $d_{T_2}(q_1, v_1) = w$, $d_{T_2}(v_1, v_2) = d + b$, $d_{T_2}(v_2, q_2) = w$. We insert a new vertex v_l on the spine between v_2 and v_3 and set $d_{T_2}(v_2, v_l) = d - r - b$ and $d_{T_2}(v_l, v_3) = d$, as illustrated in the Fig. 10(c). Now we prove that such

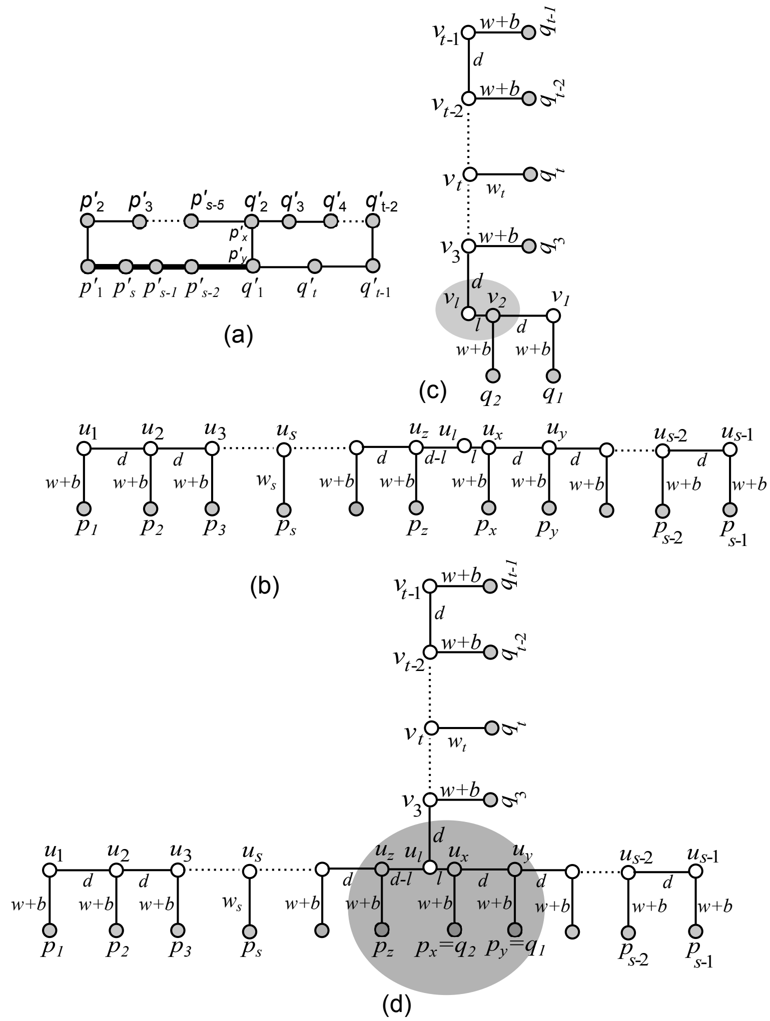

Figure 9: Pairwise compatibility tree construction for Case 2; (a) a graph G with cycles C_1 and C_2, (b) the pairwise compatibility tree T_1 of C_1, (c) the pairwise compatibility tree T_2 of C_2, and (d) the pairwise compatibility tree T of G.

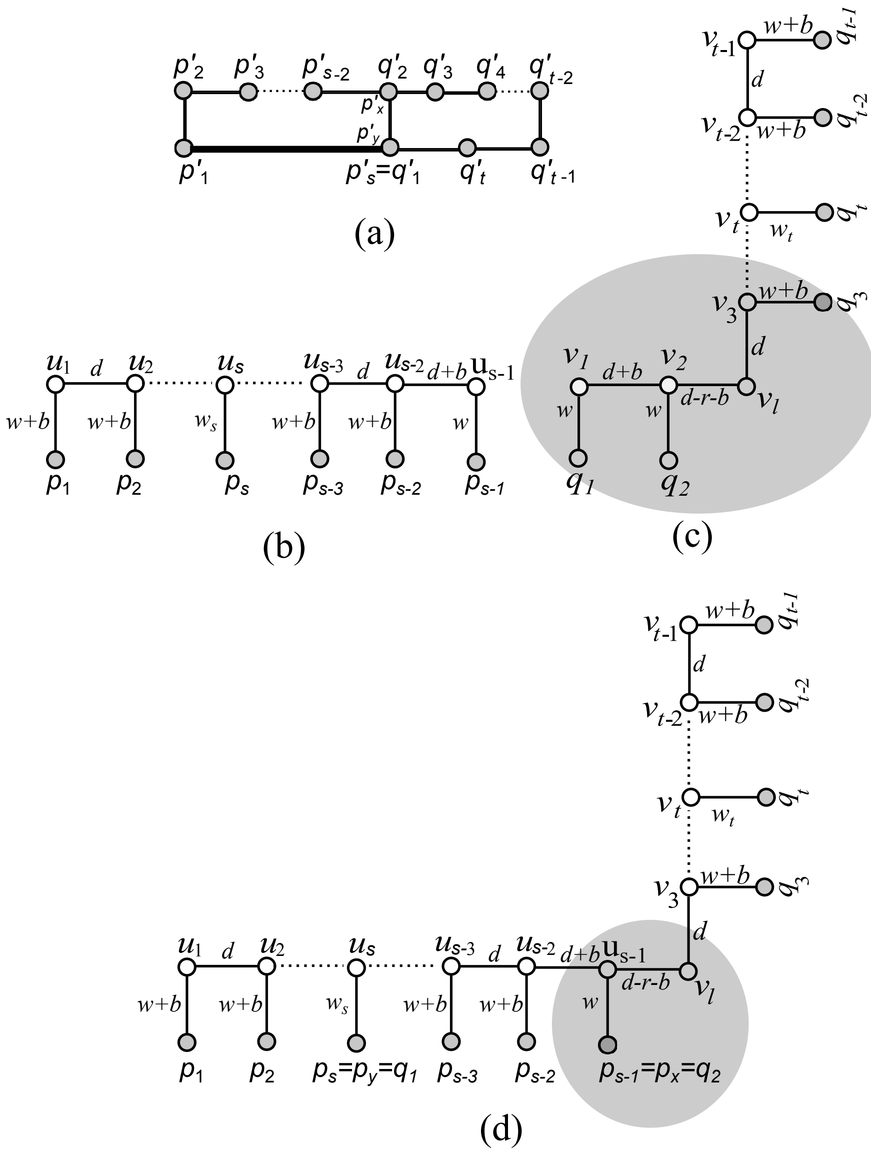

Figure 10: Pairwise compatibility tree construction for Case 3; (a) a graph G with cycles C_1 and C_2, (b) the pairwise compatibility tree T_1 of C_1, (c) the pairwise compatibility tree T_2 of C_2, and (d) the pairwise compatibility tree T of G.

changes preserve the relation for being $C_2 = PCG(T_2, d_{min}, d_{max})$.
$d_{T_2}(q_1, q_2) = w + d + b + w = 2w + b + d$, so $d_{min} < d_{T_2}(q_1, q_2) < d_{max}$.
$d_{T_2}(q_t, v_2) = w + (d - r - b)$.

$d_{T_2}(q_t, q_1) = w + (d - r - b) + d + b + w = 2w + 2d - r$, so $d_{T_2}(q_t, q_1) = d_{max}$.
$d_{T_2}(q_1, q_3) = w + d + b + d - r - b + d + w + b = 2w + 3d + b - r > d_{max}$ so $d_{T_2}(q_1, q_k) > d_{max}, 3 \leq k \leq t-1$.
$d_{T_2}(q_2, q_3) = w + d - r - b + d + w + b = 2w + 2d - r$, so $d_{T_2}(q_2, q_3) = d_{max}$.
$d_{T_2}(q_2, q_4) = w + d - r - b + d + d + w + b = 2w + 3d - r > d_{max}$, so $d_{T_2}(q_2, q_k) > d_{max}, 4 \leq k \leq t-1$.
$d_{T_2}(q_t, q_2) = d_{T_2}(q_t, v_2) + w = w + (d - r - b) + w = 2w + d - r - b < d_{min}$.

Note that $d_{T_1}(p_s, u_{s-2}) = w$.

We merge T_1 and T_2 such that the vertices q_1, v_1, v_2 and q_2 of T_2 lie on the vertices p_s, u_{s-2}, u_{s-1} and p_{s-1} of T_1, respectively, as illustrated in the Fig. 10(d).

Now p_s is q_1, the distance from p_s to q_2 is equal to $w+d+b+w$ which is within the range from d_{min} to d_{max} and the distances from p_s to each of the leaves $q_3, q_4, ..., q_{t-1}$ are greater than d_{max}. Again, the distance from q_t to $p_{s-1} = q_2$ is less than d_{min}, $d_T(q_t, p_{s-2}) = d_T(q_t, v_l) + (d - r - b) + (d + b) + (w + b) = w + (d - r - b) + (d + b) + (w + b) = 2w + 2d - r + b > d_{max}$ and the distances from q_t to each of the leaves $p_1, p_2, p_3, ..., p_{s-3}$ are greater than d_{max}. Hence all the leaves except p_s and q_t hold the relations for being $G = PCG(T, d_{min}, d_{max})$. Here p'_s and q'_t are adjacent in G and $d_T(p_s, q_t) = d_{T_2}(q_1, q_t) = d_{max}$.Thus $G = PCG(T, d_{min}, d_{max})$. □

Note that when we merge two trees the weights of the edges changed locally, as indicated by the shaded areas in Figs. 8, 9 and 10. The weights of the edges outside shaded areas remain unchanged. Based on this observation we obtain a generalization of Lemma 3 as in the following lemma.

Lemma 4 *Let G be a triangle-free biconnected outerplanar 3-graph such that $G = C_1 \cup C_2 \cup C_3 \cdots \cup C_l$ where $C_1, C_2, C_3, \cdots C_l$ are cycles of the graph. In G a pair of adjacent cycles have exactly one common edge. Let $G' = C_1 \cup C_2 \cup C_3 \cdots \cup C_{l-1}$. Let cycle C_l has a common edge on the outer face of G' and the common edge is a part of cycle C_j of G'. Let T' and T_l are pairwise compatibility trees of G' and C_l, respectively, with the same d_{min} and d_{max}. Then, a pairwise compatibility tree T for G can be obtained for the same d_{min} and d_{max} by merging T' and T_l.*

Proof: Let G be a triangle-free biconnected outerplanar 3-graph such that $G = C_1 \cup C_2 \cup C_3 \cdots \cup C_l$ where $C_1, C_2, C_3, \cdots, C_l$ are cycles of the graph. In G a pair of adjacent cycles have exactly one common edge, i.e., exactly two common vertices. Let $G' = C_1 \cup C_2 \cup C_3 \cup \cdots \cup C_{l-1}$. Let cycle C_l has a common edge on the outer face of G' and the common edge is a part of cycle C_j of G'. Let T' and T_l be pairwise compatibility trees of G' and C_l respectively. Since G is a graph with the maximum degree 3, no 3 cycles of G have a common vertex. Let V_j be the corresponding vertices of the cycle C_j in T'. Let T_j be the minimal subtree of T' containing the vertices in V_j. The first two vertices of T_l are the common leaves with T'.

Let C_x be a cycle of G' such that the common vertices of C_x and C_j are not first and second vertices of C_j (see Fig. 11.) Let T_x be the minimal subtree of T' containing the corresponding vertices of C_x. Let C_y be a cycle of G' such that the common vertices of C_y and C_j are first and second vertices of C_j(see Fig. 11.) Let T_y be the minimal subtree of T' containing the corresponding vertices of C_y.

We merge T' and T_l and get T using the technique similar to that in the proof of Lemma 3. Every pair of leaves in T corresponding to non-adjacent vertices except the last vertex of the cycles are at a distance more than d_{max}. Since no three cycles have common vertices, the last vertices of T_x and T_l are at a distance greater than d_{max}. Now we have to observe the distances of the last vertices of T_y and T_l. We have to consider nine cases, since T_y and T_j have been merged according to any one case of the three cases and also T_j and T_l have been merged according to any one case of the three cases of the proof of Lemma 3. For the cases where T_y and T_j have been merged according to either Case 1 or Case 3 and T_j and T_l have been merged according to any one case of the three cases, the last vertices of T_y and T_l are at a distance greater than d_{max}. Now let T_y and T_j have been merged according to Case 2. If T_j and T_l have been merged according to Case 1, the distance between the last vertices of T_y and T_l is either less or equal to $w - l + d + w$ which is less than d_{min} or greater or equal to $w - l + 2d + w$ which is greater than d_{max}. If T_j and T_l have been merged according to Case 2, the distance between the last vertices of T_y and T_l is either less or equal to $w - 2l + d + w$ which is less than d_{min} or greater or equal to $w - 2l + 2d + w$ which is greater than d_{max}. If T_j and T_l have been merged according to Case 3, the distance between the last vertices of T_y and T_l is greater than $w - l + 3d - r + w$ which is greater than d_{max}. Thus T is a pairwise compatibility tree of G.

□

We are now ready to prove Theorem 3.

Proof of Theorem 3: Let G be an outer subdivision of a ladder graph. We first decompose G into cycles $C_1, C_2, \cdots, C_l$ where C_i and C_{i+1}, for $1 \leq i < l$, have exactly one common edge and C_i and C_{i+2}, for $1 \leq i < l-1$, has no common vertices. We then create individual caterpillars T_i as pairwise compatibility trees for the cycles C_i, for $1 \leq i \leq l$, for the same d_{min} and d_{max} according to Lemma 2. We have to merge the caterpillars $T_1, T_2, \cdots, T_l$ of cycles $C_1, C_2, \cdots, C_l$.
Let $G_i = C_1 \cup C_2 \cup \cdots \cup C_i$. We merge the caterpillars T_1 and T_2 and get the resulting tree T'_2 as a pairwise compatibility tree of G_2 according to Lemma 3. Assume that we have merged $T_1, T_2, \cdots, T_j$ and obtained a pairwise compatibility tree T'_j for the graph $G_j = C_1 \cup C_2 \cup \cdots \cup C_j$, for $2 \leq j < l$. We merge T'_j and T_{j+1} and get a resulting tree T'_{j+1} as pairwise compatibility tree of the graph G_{j+1} according to Lemma 4. □

Based on the proof of Theorem 3, one can obtain a linear-time algorithm to construct a pairwise compatibility tree of an outer subdivision of a ladder graph.

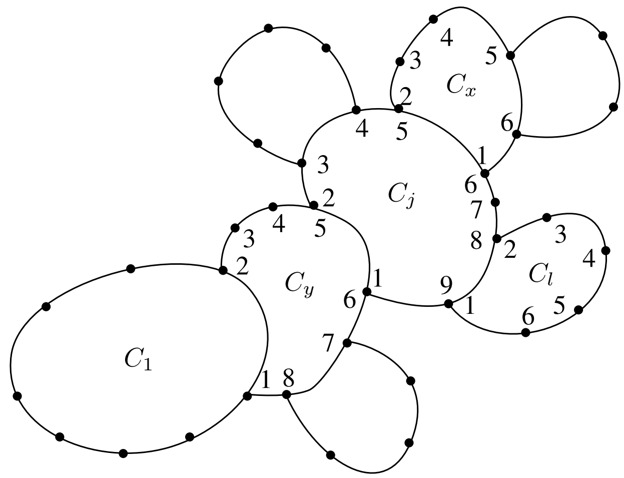

Figure 11: Illustration for C_y, C_j, C_x and C_l.

4 Triangle-free outerplanar 3-graphs are PCGs

In this section we present our main result that triangle-free outerplanar 3-graphs are pairwise compatibility graphs. In Section 4.1 we deal with biconnected triangle-free outerplanar 3-graphs and in Section 4.2 we deal with the general case.

4.1 Triangle-free biconnected outerplanar 3-graphs

In this section we show that every triangle-free biconnected outerplanar 3-graph is a PCG by giving a linear-time construction. We have the following lemma.

Lemma 5 *Every triangle-free biconnected outerplanar 3-graph is a pairwise compatibility graph.*

Proof: Let G be a biconnected outerplanar 3-graph. Let G' be the weak dual graph of G. Then G' is a tree. Every node of G' corresponds to a face, which is a cycle, of G. We create pairwise compatibility trees for these cycles for the same d_{min} and d_{max} according to Lemma 2. We incrementally construct a pairwise compatibility tree T of G as follows. Initially we have an empty tree and begin a depth first search (DFS) on G'. The first time we encounter a vertex in our depth first search, we add the pairwise compatibility tree of the cycle in G corresponding to this node as a branch of T. For the next vertex in the DFS of G', we merge the pairwise compatibility tree of the cycle in G corresponding to this node with already constructed part of T by Lemma 4. The construction of T is completed with the completion of DFS on G'. The process is illustrated in Fig. 12. □

The algorithm described in the proof of Lemma 5 is called **Biconnected-Construction**. Algorithm **Biconnected-Construction** runs in linear time.

4.2 Triangle-free outerplanar 3-graphs

We call a subgraph B of a connected graph G, a *biconnected component* of G if B is a maximal biconnected subgraph of G. We call an edge (u, v) a *bridge* of G if the deletion of (u, v) results in a disconnected graph. Any connected graph can be decomposed to biconnected components and bridges. A *block* of a connected graph G is either a biconnected component or a bridge of G. The graph in Fig. 13(a) has the blocks B_1, B_2, B_3 depicted in Fig. 13(b). The blocks and cut vertices in G can be represented by a tree T, called the *BC-tree* of G. In T each block is represented by a *B-node* and each cut vertex of G is represented by a *C-node*. We can have a pairwise compatibility tree of a biconnected component by Algorithm **Biconnected-Construction**. For a bridge (u', v'), we can construct pairwise compatibility tree as a caterpillar T where u and v are at a distance equal to d_{min} in T. We now have the following theorem.

Theorem 4 *Every triangle-free outerplanar 3-graph is a pairwise compatibility graph.*

Proof: Let G be a connected triangle-free outerplanar 3-graph. Let G' be the block-cutpoint graph of G. Since G is connected, G' is a tree [21]. We consider G' as a rooted tree with root B_1. Let $B_1, B_2, \cdots, B_b$ be the ordering of the blocks following a depth-first search order starting from B_1. We assume that we have obtained a pairwise compatibility tree T_i by merging the pairwise compatibility trees of the blocks $B_1, B_2, \cdots, B_i$ and that we are now going to obtain a pairwise compatibility tree T_{i+1} by merging T_i with pairwise compatibility tree of the block B_{i+1}. Let v_t be the cut vertex corresponding to the C-node which is the parent of B_{i+1} in T. Let B_x be the parent of v_t in T . Then both B_x and B_{i+1} contain v_t, and T_i contains the drawing of B_x. Since G is a 3-graph, both B_x and B_{i+1} cannot be biconnected components. Then either both B_x and B_{i+1} are bridges or one of B_x and B_{i+1} is a biconnected component and the other is a bridge. If B_{i+1} is a bridge (x', y'), we construct a pairwise compatibility tree of (x', y') as a caterpillar T as illustrated in Fig. 13(c), where x and y are at a distant equal to d_{min} in T. However, if B_{i+1} is a biconnected component, we have pairwise compatibility tree T_{i+1} by Algorithm **Biconnected-Construction**. For both the cases, we can merge T_i and T_{i+1} by overlapping the edges containing the vertex v_t. The construction is illustrated in Fig. 13. □

Clearly, the algorithm described in the proof of Theorem 4 runs in linear time.

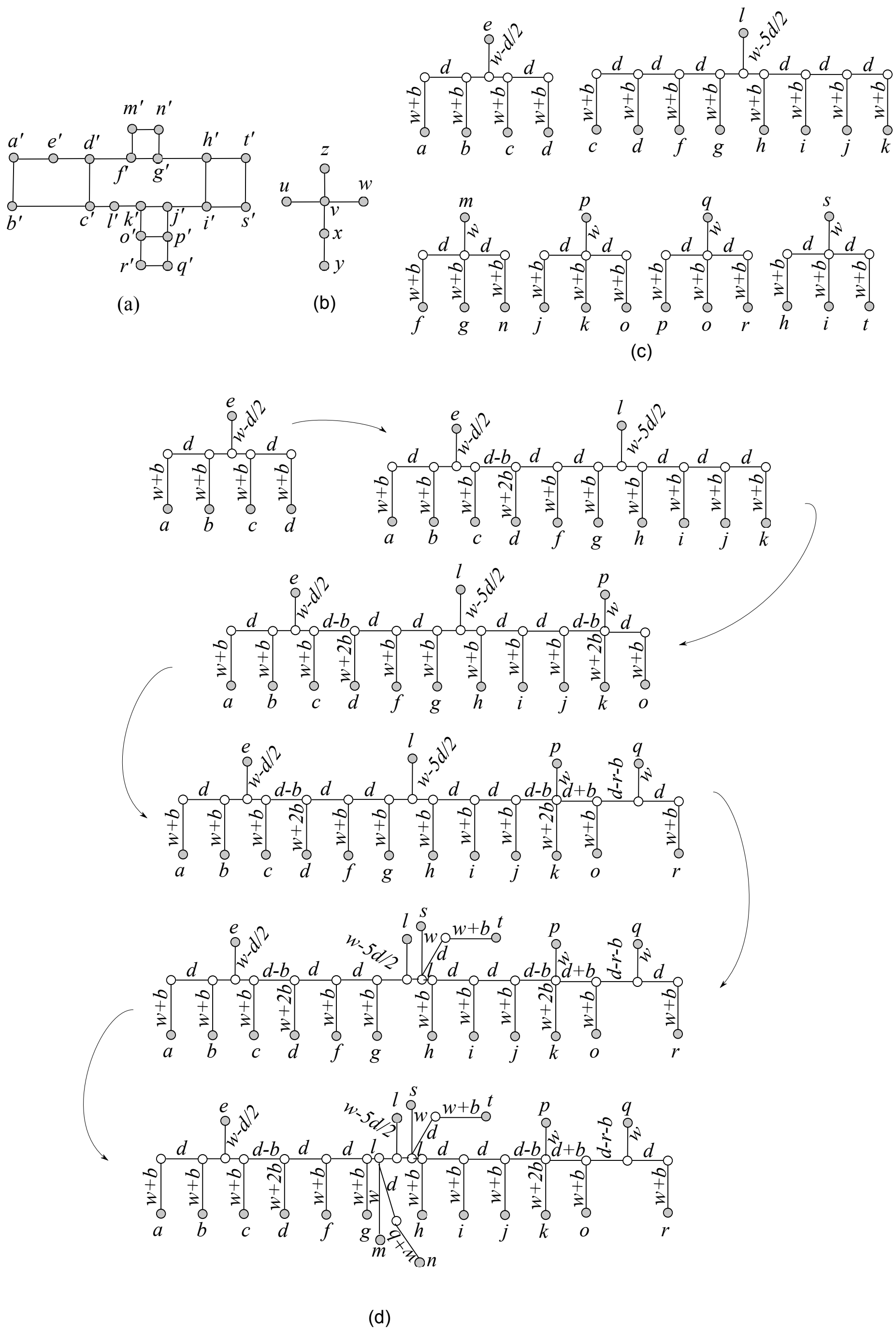

Figure 12: (a) A triangle-free biconnected outerplanar graph G, (b) weak dual graph G' of G, (c) pairwise compatibility trees for the internal faces of G and (d) incremental construction of a tree T.

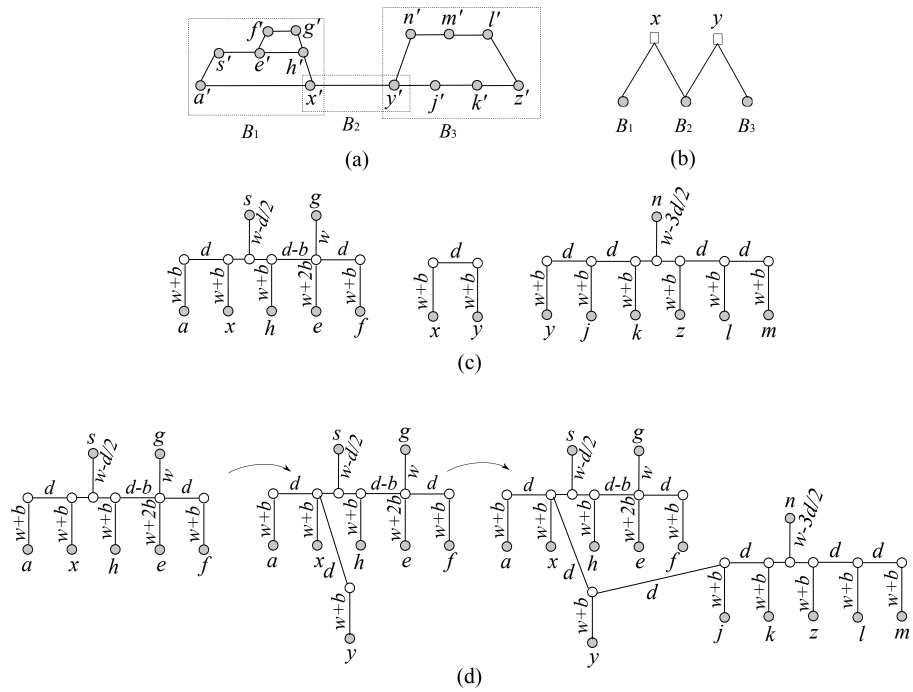

Figure 13: (a) An outerplanar graph G, (b) block-cutpoint graph G' of G, (c) pairwise compatibility trees for the biconnected blocks and (d) incremental construction of a tree T.

5 Conclusion

In this paper we have proved that trees and ladder graphs are PCGs by giving linear-time algorithms for constructing pairwise compatibility trees for graphs of these classes. Additionally, we have proved that outer subdivisions of ladder graphs are also PCGs. Finally we have proved that triangle free outerplanar 3-graphs are PCGs. However, a complete characterization of PCGs is not known. The characterization of PCGs would be helpful to characterize leaf power graphs and chordal graphs. All graphs of at least seven vertices are PCGs [6] and the only graph of 15 vertices is proved not to be PCG [22]. Thus interested researcher can work on finding the smallest graph that is not a PCG. The clique problem can be solved in polynomial time for the class of compatibility graphs if we are able to construct a weighted tree in polynomial time for those graphs. Hence it is very interesting to identify the pairwise compatibility graph classes.

Acknowledgements

This work is done in Graph Drawing & Information Visualization Laboratory of the Department of CSE, BUET, established under the project "Facility Upgradation for Sustainable Research on Graph Drawing & Information Visualization" supported by the Ministry of Science and Information & Communication Technology, Government of Bangladesh.

References

[1] A. Brandstädt. On leaf powers. Technical report, University of Rostock, 2010.

[2] A. Brandstädt, V. Chepoi, and F. F. Dragan. Distance approximating trees for chordal and dually chordal graphs (extended abstract). In *Proceedings of the 5th Annual European Symposium on Algorithms*, ESA '97, pages 78–91, London, UK, UK, 1997. Springer-Verlag. URL: `http://dl.acm.org/citation.cfm?id=647907.739970`.

[3] A. Brandstädt, V. Chepoi, and F. Draganc. Distance approximating trees for chordal and dually chordal graphs. *Journal of Algorithms*, 30:166–184, 1999. `doi:10.1006/jagm.1998.0962`.

[4] A. Brandstädt, C. Hundt, F. Mancini, and P. Wagner. Rooted directed path graphs are leaf powers. *Discrete Mathematics*, 310:897–910, 2010. `doi:10.1016/j.disc.2009.10.006`.

[5] A. Brandstädt, V. B. Le, and R. Sritharan. Structure and linear-time recognition of 4-leaf powers. *ACM Transactions on Algorithms*, 5:11:1–11:22, 2008. `doi:10.1145/1435375.1435386`.

[6] T. Calamoneri, D. Frascaria, and B. Sinaimeri. All graphs with at most seven vertices are pairwise compatibility graphs. *CoRR*, abs/1202.4631, 2012.

[7] T. Calamoneri, R. Petreschi, and B. Sinaimeri. On relaxing the constraints in pairwise compatibility graphs. In *Proceedings of the 6th International Workshop on Algorithms and Computation*, volume 7157 of *Lecture Notes in Computer Science*, pages 124–135. Springer, 2012. `doi:10.1007/978-3-642-28076-4_14`.

[8] M. Farber. Characterizations of strongly chordal graphs. *Discrete Mathematics*, 43:173–189, 1983. `doi:10.1016/0012-365X(83)90154-1`.

[9] M. R. Fellows, D. Meister, F. A. Rosamond, R. Sritharan, and J. A. Telle. Leaf powers and their properties: Using the trees. In *ISAAC'08*, pages 402–413, 2008.

[10] L. Foulds and R. Graham. The Steiner problem in phylogeny is NP-complete. *Advances in Applied Mathematics*, 3:43–49, 1982. `doi:10.1016/S0196-8858(82)80004-3`.

[11] T. Jiang, P. Kearney, and M. Li. A polynomial time approximation scheme for inferring evolutionary trees from quartet topologies and its application. *Siam Journal on Computing*, 30:1942–1961, 2001. `doi:10.1137/S0097539799361683`.

[12] N. C. Jones and P. A. Pevzner. *An Introduction to Bioinformatics Algorithms.* The MIT Press, Cambridge, 2004.

[13] P. Kearney, I. Munro, and D. Phillips. Efficient generation of uniform samples from phylogenetic trees. In *Algorithms and Bioinformatics: 3rd International Workshop (WABI)*, volume 2812 of *Lecture Notes in Bioinformatics*, pages 177–189. Springer, September 2003. `doi:10.1007/978-3-540-39763-2_14`.

[14] W. Kennedy, G. Lin, and G. Yan. Strictly chordal graphs are leaf powers. *J. of Discrete Algorithms*, 4:511–525, 2006. `doi:10.1016/j.jda.2005.06.005`.

[15] A. Kolen. *A polynomial-time algorithm for solving the set covering problem on a totally-balanced matrix.* Technical report BW 147, Mathematisch Centrum, Ams-terdam, 1981.

[16] A. M. Lesk. *Introduction to Bioinformatics.* Oxford University Press, Oxford, 2002.

[17] N. Nishimura, P. Radge, and D. Thilikos. On graph powers for leaf-labeled trees. *Journal of Algorithms*, 42:69–108, 2002. `doi:10.1006/jagm.2001.1195`.

[18] D. Phillips. Uniform sampling from phylogenetics trees. Master's thesis, University of Waterloo, August 2002.

[19] S. A. Salma and M. S. Rahman. Ladder graphs are pairwise compatibility graphs. In *Asian Association for Algorithms and Computation (AAAC)*, 2011.

[20] S. A. Salma and M. S. Rahman. Triangle-free outerplanar 3-graphs are pairwise compatibility graphs. In *Proceedings of the 6th International Workshop on Algorithms and Computation*, volume 7157 of *Lecture Notes in Computer Science*, pages 112–123. Springer, 2012. `doi:10.1007/978-3-642-28076-4_14`.

[21] D. West. *Introduction to Graph Theory.* Prentice Hall, New York, 2003.

[22] M. N. Yanhaona, M. S. Bayzid, and M. S. Rahman. Discovering Pairwise Compatibility Graphs. *Discrete Mathematics, Algorithms and Applications*, 2:607–623, 2010. `doi:10.1142/S1793830910000917`.

[23] M. N. Yanhaona, K. S. M. T. Hossain, and M. S. Rahman. Pairwise compatibility graphs. *Journal of Applied Mathematics and Computing*, 30:479–503, 2009. `doi:10.1007/s12190-008-0204-7`.