Proof regarding the NP-completeness of the unweighted complex-triangle elimination (CTE) problem for general adjacency graphs

S.Roy, S.Bandyopadhyay and U.Maulik

Abstract: The elimination of all complex triangles (CT) is an essential step in the rectangular dualisation approach of floor-planning. It is known that the weighted complex triangle elimination problem, i.e. the version of the problem where the input to the problem is a weighted adjacency graph, is NP-complete. Also, for adjacency graphs with 0-level containment the unweighted problem is optimally solvable in polynomial time. However, the complexity of the unweighted CTE problem for general graphs with multiple levels of containment was unknown though it was conjectured that this problem is also NP-complete. The authors present a claim that the unweighted complex triangle elimination problem for general graphs with multiple levels of containment is, indeed, NP-complete, and present a proof supporting the claim.

1 Introduction

Floor-planning is an early step in VLSI chip design in which the relative location of functional entities in a chip is decided. A floor-plan can be represented by a partition of a rectangular chip area into modules (usually rectangular polygons) where each module stands for a functional entity. Such a partition can be expressed by a partition graph in which each region of the graph represents a module, edges stand for sides of the modules, and vertices are junctions. The adjacency relationship between the individual modules is generally expressed with the help of an adjacency graph $G_{ad}(V_{ad}, E_{ad})$, where each node represents a functional module and an edge between two nodes express the adjacency requirement between the corresponding modules in space. In the case of the rectangular dualisation approach to floor planning [1], each module is realised as a rectangular area. In this approach an edge *e* of the adjacency graph is mapped to a side that is shared by two adjacent rectangular areas corresponding to the nodes incident on *e*.

A complex triangle (CT) is a cycle of three edges that contains a node inside the enclosed area. The dual of a rectangular floor plan (i.e. the adjacency graph corresponding to the floor-plan) does not contain any CT [2]. In [3] it has been proved that a rectangular, also called 0-concave rectilinear module (0-CRM), floor-plan does not exist if and only if G_{ad} contains a CT. This complicates the construction of a floor-plan because all CTs must be eliminated before the floor-plan phase.

Kozminsky and Kinen [4] developed a technique to transform one floor-plan to another where adjacency requirements are preserved. Bhaskar and Sahni [5] reported a linear time algorithm for constructing a rectangular floor-plan from a planar triangulated graph (PTG) devoid of any CT. In [6] the generation of sliceable floor-plans was considered. In this paper an algorithm with $O(n \times \log(n) + h \times n)$ time to generate a sliceable floor-plan for an *n*-vertex adjacency graph is presented, where *h* is the height of the sliced floor-plan tree.

In the rectangular dualisation approach to floor-planning, all CTs have to be eliminated before the floor-planning phase. The expected number of CTs for a randomly generated PTG is approximately 16% of the number of vertices [7]. A method was proposed in [8] for the CT elimination problem for graphs without any nested CT. In [9] a technique that introduces new vertices and edges in the original PTG, producing empty spaces in the floor-plan, was proposed to eliminate all CTs. Yeap and Sarrafzadeh [2] have shown that if 2-bend modules (i.e. T-shaped and Z-shaped) are allowed, then every biconnected PTG admits a floor-plan. They also developed a linear-time algorithm for constructing a 2-CRM (concave rectilinear module) floor-plan of an arbitrary PTG.

Complex triangle (CT) elimination in weighted adjacency graphs has been shown to be NP-complete [8]. In this work the more general CT elimination problem for unweighted adjacency graphs with multiple levels of nesting of CTs is addressed. We prove that the CT elimination problem for general unweighted adjacency graphs is NP-complete. The proof consists of two parts. In the first part we show that the unweighted CT elimination problem is in NP. Then, in the second part, we present a method to reduce a well known NP-complete problem [10] to the unweighted CT elimination problem in polynomial time.

Paper first received 26th March and in revised form 7th September 2001

S. Roy and U. Maulik are with the Department of Computer Science & Technology, Kalyani Govt. Engineering College, P.O.-Kalyani, Nadia, W.B., Pin 741235, India

S. Bandyopadhyay is with the Machine Intelligence Unit, Indian Statistical Institute, 203 B. T. Road, Calcutta 700035, India

2 Preliminaries

In this Section the basic concepts and definitions employed in this paper are first explained. This is followed by a presentation of the technique to eliminate a CT. The Section ends with the formulation of the unweighted complex triangle elimination (CTE) problem and an outline of the proof.

2.1 Basic concepts and definitions

An adjacency graph is a simple connected graph $G_{ad}(V_{ad}, E_{ad})$, where each node $v \in V_{ad}$ represents a rectangular module and an edge $(v_i, v_j) \in E_{ad}$ between two nodes v_i and v_j implies that there are interconnections between the modules corresponding to v_i and v_j. When an adjacency graph is finally converted to a floor-plan through rectangular dualisation, each edge $(v_i, v_j) \in E_{ad}$ will map to a shared side of the two rectangular areas corresponding to v_i and v_j. Two simple adjacency graphs and their rectangular duals are shown in Fig. 1*a* and Fig. 1*b*.

A PTG is a plane embedding of a simple planar graph where every face except the exterior is a triangle. For example, the adjacency graph shown in Fig. 1*a* is not a PTG, whereas the graph shown in Fig. 1*b* is. Fig. 2 shows another PTG.

Given a PTG, a CT in it is defined to be a cycle of length 3 which is not a face, i.e. which encloses an area containing one or more vertices. For example, in Fig. 2, the cycle *abca* is a CT because the node *g* is enclosed within it. A CT is said to be isolated if it does not share any edge with any other CT. There exists no rectangular dual of a CT because three rectangles adjacent to each other cannot enclose another rectangle (Fig. 3). The technique of eliminating a CT is explained later.

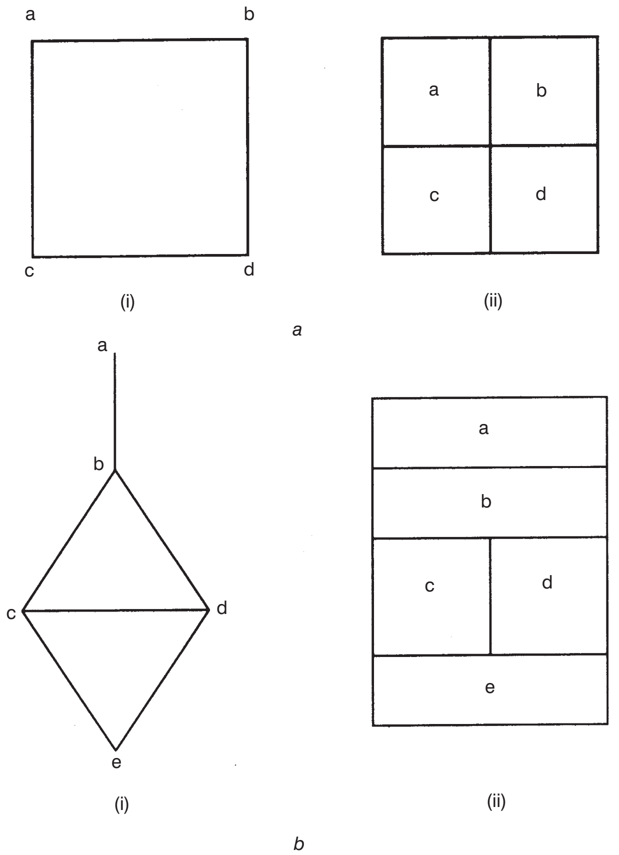

Fig. 1 *Two simple adjacency graphs and their rectangular dual*
a Example 1
(i) Adjacency graph
(ii) Rectangular dual
b Example 2
(i) Adjacency graph
(ii) Rectangular dual

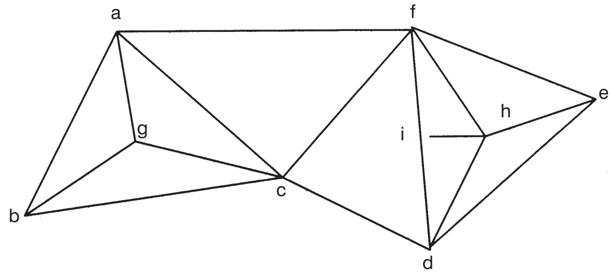

Fig. 2 *Planar triangulated graph (PTG)*

A properly triangulated planar (PTP) graph is a simple connected planar graph that satisfies the following properties.

P1: Every face, except the exterior, is a triangle.
P2: All internal vertices have degree greater than or equal to 4.
P3: All cycles that are not faces have length greater than 3.

For example, the adjacency graph shown in Fig. 1*b* is a PTP graph whereas that shown in Fig. 2 is not, because the cycle *abca* does not satisfy property P3 and the node *g* does not satisfy property P2. A PTP can be mapped to its rectangular dual in $O(n)$ time [5], but a PTG may not be converted to a rectangular dual because it may contain a complex triangle. Therefore, a PTG must first be mapped to a PTP by eliminating the complex triangles, after which the rectangular dual may be obtained.

For a graph G(V, E), a set $S \subseteq V$ is a node cover iff all edges in E are incident to at least one vertex in S. For example, in Fig. 4*a*, the set $\{d, e\}$ or $\{d, c\}$ is a node cover, but $\{a, b, c\}$ is not a node cover because none of the nodes incident on edge (d, e) is present in $\{a, b, c\}$. Similarly, in Fig. 4*b*, $\{a, c, f, m, i, j\}$ is a node cover but $\{c, g, m, h, k\}$ is not because the edge (a, b) is not incident on any of the nodes it contains.

Let G(V, E) be a graph of n vertices $V = \{v_1, v_2, \ldots, v_n\}$ and m edges $E = \{e_1, e_2, \ldots, e_m\}$, without any isolated vertex, such that there is a label attached to each edge. A particular edge-label may be associated with more than one edge. An edge-label cover is a set $E_{elc} = \{e_1, e_2, \ldots, e_p\}$ of

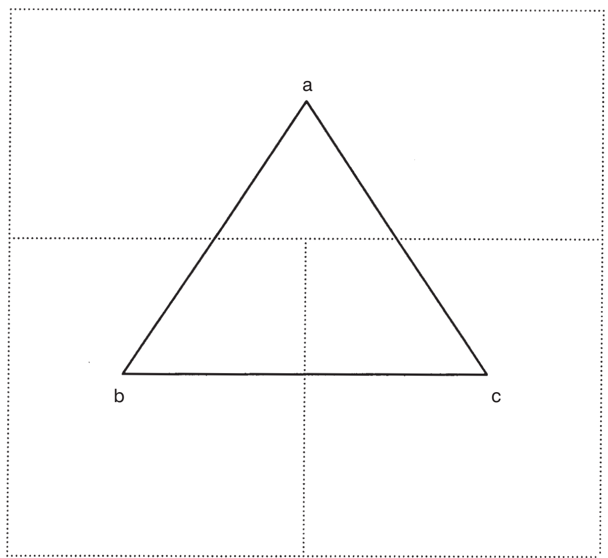

Fig. 3 *Three adjacent rectangles cannot enclose another rectangle*

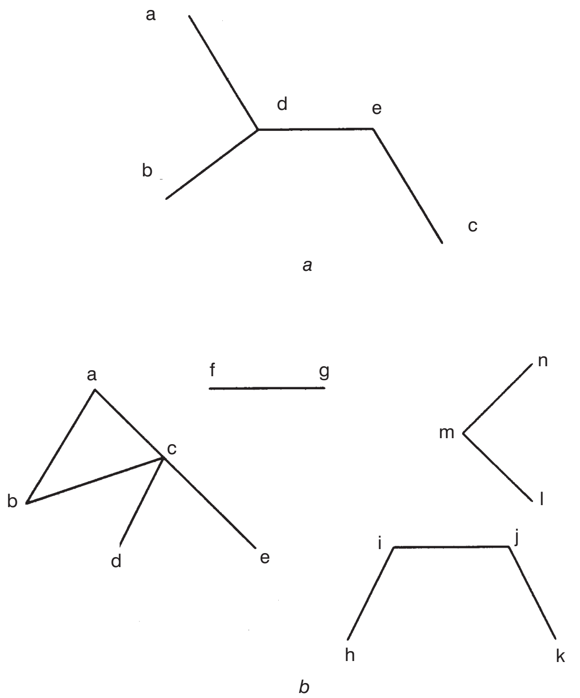

Fig. 4 *Two graphs and their node covers*

a Node cover $\{d, e\}$ or $\{d, c\}$
b Node cover $\{a, c, f, m, i, j\}$

edge-labels such that each vertex of G is covered by at least one $e_i \in E_{elc}$. For example, in Fig. 5*a*(ii), the set $\{bc, ac\}$ is an edge-label cover but $\{bc\}$ is not. In Fig. 5*b*(ii) the set $\{jk, ik, em, fh, gn\}$ is an edge-label cover. Note that we have not considered the isolated vertex 4, which, as we shall see later, does not have an impact on our problem. If any edge-label from this set is removed, the resulting set will no longer cover the entire set of vertices.

A complex triangle X is said to *contain* another complex triangle Y if the region bounded by the edges of Y is inside the region bounded by the edges of X. If a complex triangle X contains another complex triangle Y and there is no complex triangle Z such that X contains Z and Z contains Y then X is said to immediately contain Y, or Y is said to be immediately contained by X. In Fig. 5*a*(i), the complex triangle *abc* contains the complex triangle *bcf* but is not contained by any other CT. There are four complex triangles in the adjacency graph of Fig. 5*a*(i). The remaining two CTs, *ace* and *bcd*, neither contain nor are contained by any other CT.

Let G(V, E) be a PTG. Let us add a large CT containing all CTs of G, to obtain G′. A complex triangle tree T(G) is a tree where vertices correspond to the complex triangles of G′. There is an edge $(X, Y) \in T$ if and only if Y immediately contains X. G is said to have *k*-level containment if the length of the longest path in T from leaf to root is $k+1$.

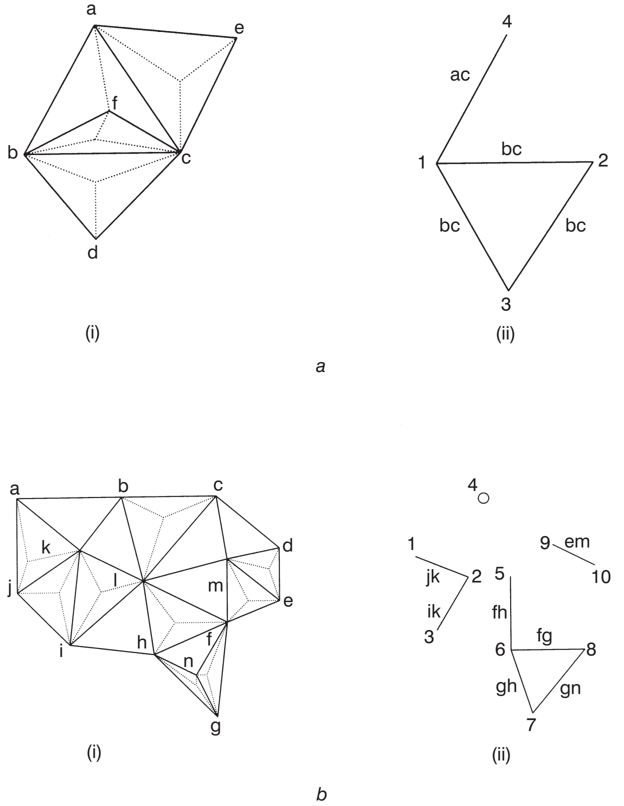

Fig. 5 *Conversion of adjacency graph G_{ad} to its CT adjacency graph G_{ct} (Examples 1 and 2)*

a Example 1
(i) Adjacency graph G_{ad}
(ii) Corresponding CT adjacency graph G_{ct}
b Example 2
(i) Adjacency graph G_{ad}
(ii) Corresponding CT adjacency graph G_{ct}

2.2 Elimination of complex triangles

There exists no rectangular dual of a CT because three rectangles, adjacent to each other, cannot enclose another rectangle (Fig. 3). The only way to convert a CT to a rectangular dual is to select an edge and insert a vertex on to it. This converts a CT into a complex 4-cycle that leads to a rectangular region in the floor-plan which is not a module of the original circuit but contains only routing wires. The technique is illustrated in Fig. 6. The CT of Fig. 6*a* is eliminated by introducing an extra node *e* on the side *bc* and connecting the node *d* to *e*. The resulting graph is shown in Fig. 6*b*. The extra node *e* maps to the region marked *e* in Fig. 6*c*. The region *e* will contain only routing wires connecting modules *b* and *c*.

Therefore, given a PTG G_{ad} with an arbitrary level of containment, all CTs in G_{ad} can be eliminated by selecting an outer edge from each CT, putting an extra vertex on that edge and introducing the new edges connecting the new vertex to other vertices as necessary. However, whether the resulting adjacency graph (after elimination of all CTs) can still be converted to a floor-plan is not obvious. In the following lemma we prove that it is always possible to obtain an adjacency graph G'_{ad} from G_{ad} in the above-mentioned procedure that can readily be converted to a rectangular floor-plan.

Lemma 1: Given a PTG G_{ad} with arbitrary level of containment of CTs, it is possible to obtain an adjacency graph G'_{ad} from G_{ad} by selecting an outer edge from each CT of G_{ad} and putting an extra node on that edge, such that there exists a rectangular dual of G'_{ad}.

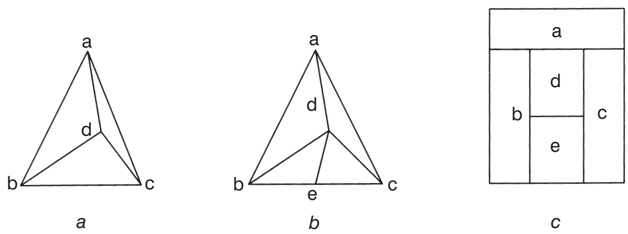

Fig. 6 *Elimination of complex triangle (CT)*

a Complex triangle (CT)
b CT converted to complex 4-cycle
c Floor-plan after removal of CT

Proof: It is known that a PTP graph can be mapped to a rectangular dual in linear time [5]. Therefore all we have to show is that it is always possible to convert a given G_{ad} (with an arbitrary level of containment) to G'_{ad} using the procedure mentioned above such that G'_{ad} is a PTP graph.

Suppose *abc* is a CT in G_{ad}. Without loss of generality, suppose that the edge *bc* is chosen for eliminating this CT. Accordingly, a new vertex *e* is inserted onto it so that *bc* is now broken into two edges, *be* and *ec*. Since *abc* is a complex triangle, there exists at least one node, say *d*, within the region enclosed by the cycle *abca* which is connected to the vertices *b* and *c*. This is guaranteed by the fact that G_{ad} is triangulated. To eliminate the CT *abc* a new edge *de* connecting the vertices *d* and *e* has to be inserted. Now two cases may arise.

Case (i): The edge *bc* is on the boundary of the region enclosed by G_{ad} (Fig. 7). In this case, the insertion of the new vertex *e* and new edge *de* neither introduces a face that is not a triangle, nor produces any internal vertex with degree less than 4, nor creates a cycle that is not a face and has a length equal to 3 (i.e. a CT). Therefore none of the conditions of a PTP graph are violated.

Case (ii): The edge *bc* is inside the region enclosed by G_{ad} (Fig. 8*a*). Since *bc* is inside the region enclosed by G_{ad} and G_{ad} is triangulated, there must exist a triangle outside the region of *abc* of which *bc* is an edge (Fig. 8*a*). Let that triangle be *bcf*. We introduce the edge *ef* (Fig. 8*b*). It is easy to see that by doing so none of the three conditions of a PTP graph are violated.

If we go on eliminating all the CTs of G_{ad} in this way, i.e. by introducing one new vertex for each CT and one or two edges, whichever is appropriate, for each newly introduced vertex, the resulting graph G'_{ad} is a PTP graph. □

Thus, given a planar triangulated adjacency graph G_{ad}, all CTs of G_{ad} can be eliminated by selecting an edge from each CT and putting an extra node on that edge. However, with each extra node, an additional rectangular area is introduced in the floor-plan that does not contain any circuitry. This is undesirable because it wastes valuable floor area and complicates routing. Therefore the number of such additional areas should be minimised. Since more than one CT may share an edge, it is possible to eliminate all such CTs in one go by selecting such a shared edge and putting an extra node on it. The complex triangle elimination (CTE) problem for general adjacency graphs with an arbitrary level of containment may, thus, be defined as the problem of finding the minimal set of edges so that each CT may have at least one edge in that set.

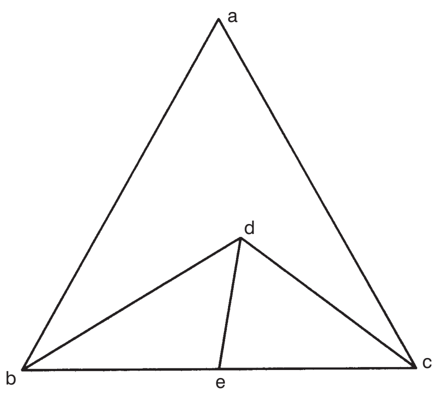

Fig. 7 *Case (i): The extra node e is on the border of the region enclosed by G_{ad}*

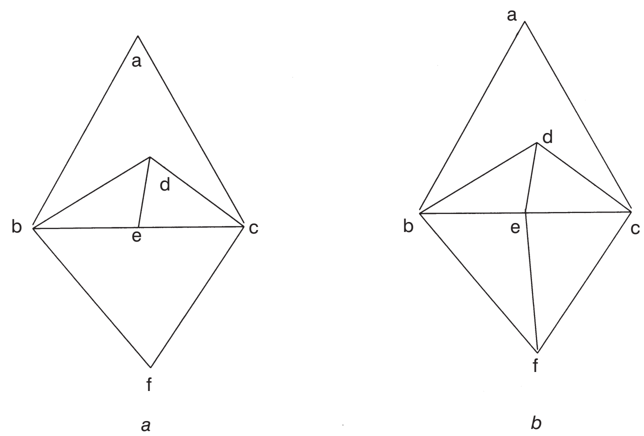

Fig. 8 *Case (ii): The extra node e is inside the region enclosed by G_{ad}*
a A 4-cycle *becf* appears that is a face
b Condition for PTP is restored by introducing the edge *ef*

In [8] Sun and Yeap have shown that for adjacency graphs with 0-level containment the unweighted problem is optimally solvable in polynomial time, although the complexity of the unweighted CTE problem for general graphs with multiple levels of containment was unknown. They conjectured that this problem is also NP-complete. In this paper, we prove that the unweighted CTE problem for adjacency graphs with multiple level of containment is also NP-complete even though for 0-level containment it is optimally solvable in polynomial time. This can be intuitively explained by the fact that in the case of 0-level containment, i.e. when no CT of G_{ad} contains (or is contained by) any other CT, an edge *e* of G_{ad} can be shared by at most two CTs. This is because, given a straight line-segment *pq*, at most two non-overlapping triangles can be drawn on it. This greatly simplifies the optimisation problem and renders it solvable in polynomial time. In the case of adjacency graphs with multiple levels of containment, an edge may be shared by an arbitrary number of CTs. Moreover, while eliminating a CT in an optimal way, it can be approached from outside, i.e. from CTs containing it, or from inside, i.e. from CTs contained by it. This complicates the problems as the number of choices multiply.

The outline of the proof supporting the claim that the unweighted CTE problem with multiple levels of containment is NP-complete is now described. It is first shown that the decision problem corresponding to the said optimisation problem is in NP. Then a graph called the complex-triangle-adjacency graph G_{ct} is constructed from G_{ad} in polynomial time and the CTE decision problem is formulated in terms of this graph G_{ct}. In particular, we show that the CTE decision problem is equivalent to the edge-label cover decision problem for such G_{ct} obtained from G_{ad}. The concept of an edge-label as well as the edge-label cover decision problem has been introduced in this work. Finally, it has been proved that the edge-label cover decision problem is NP-complete by reducing the node cover decision problem, a well-known NP-complete problem [10] to the edge-label cover decision problem, in polynomial time.

3 Complete proof

In this Section we formally state the CTE problem for unweighted adjacency graphs with an arbitrary level of containment and then present the complete proof regarding its NP-completeness.

The CTE problem can be stated as follows: given a planar triangulated adjacency graph $G_{ad}(V_{ad}, E_{ad})$ with an arbitrary level of containment of CTs, the problem is to find a set $E' \subseteq E_{ad}$ of minimal cardinality such that E' contains at least one edge from each complex triangle of G_{ad}. The CT elimination problem is an optimisation problem. We may state the corresponding decision problem in the following way.

CT elimination decision problem: Given a PTG $G_{ad}(V_{ad}, E_{ad})$ with an arbitrary level of containment, is there a set $E' \subseteq E_{ad}$, $|E'| \leq k$, such that E' contains at least one edge from each CT of G_{ad}?

As already mentioned, in the weighted version of the CT elimination problem there is a positive weight attached to each edge of the PTG. This positive weight is proportional to the importance of interconnection, e.g. the number of nets connecting the modules. In the weighted CT elimination problem, the total weight of E' (i.e. $\Sigma_{u,v} w(u, v)$, $(u, v) \in E'$) should be minimal. The weighted CT elimination problem is NP-complete. Also, for an adjacency graph with 0-level of containment, the unweighted complex triangle elimination problem is optimally solvable in polynomial time. In this paper, we consider the more generalised problem of k-level containment. We establish that the generalised unweighted CT elimination problem is NP-complete.

Theorem 1: The unweighted CT elimination decision problem is in NP.

Proof: Let $G_{ad}(V_{ad}, E_{ad})$ be a PTG containing n edges, say $e_1, e_2, \ldots, e_n$ and p number of CTs, say $CT_1, CT_2, \ldots, CT_p$. Let $\{e_{i1}, e_{i2}, e_{i3}\}$ be the outer edges of CT_i, $\forall i = 1, 2, \ldots, p$. A non-deterministic algorithm to obtain a set of k edges such that there is at least one edge from each CT, if there exists such a set, is given below in **Algorithm** CT-COVER-k in a pseudo-language. The algorithm uses an array S of size k. For each element $S[i]$ of the array, an edge of $E_{ad} = \{e_1, e_2, \ldots, e_n\}$ is selected non-deterministically. Note that the set of edges included in S may contain less than k edges because an edge may be selected more than once. Once S is filled up in this way, the algorithm checks whether each CT of G_{ad} contains at least one edge in it. The paradigm of non-determinism ensures that S will contain a set of k or less number of edges that covers all CTs of G_{ad} provided there exists such a set.

Algorithm CT-COVER-k

/* To find a set of k edges that covers all CTs of a given PTG $G_{ad}(V_{ad}, E_{ad})$. */
/* S is an array of k elements, each element of S contains an edge of G_{ad} */

1. Begin
2. For $i \leftarrow 1$ To k Do
3. $j \leftarrow$ choice $(1 : n)$
/*choose an integer between 1 to n non-deterministically*/
4. $S[i] \leftarrow e_j$ /* include e_j, the jth edge, in the set */
5. Endfor
6. For $i \leftarrow 1$ To p Do /* has the ith CT been covered? */
7. covered $\leftarrow$ FALSE
/* start assuming that it is not covered */
8. For $j \leftarrow 1$ To k Do
/* see if the jth edge in S covers CT_i*/
9. If $S[j] \in \{e_{i1}, e_{i2}, e_{i3}\}$ Then
10. covered $\leftarrow$ TRUE, break
/* go for the next CT if this CT is covered */
11. Endif;
12. Endfor
13. If (covered = FALSE) Then **failure** Endif
/* CT_i is not covered */
14. Endfor
15. **Success** /* all CTs are covered */
16. End-CT-COVER-k.

Analysis of **Algorithm** CT-COVER-k

The For-loop of lines 2–5 will execute k times. The For-loop of lines 6–14 runs, in the worst case i.e. in case the set really covers all CTs, p times and it contains an inner loop on lines 8–12 that runs, in the worst case, k times. Therefore the time complexity of the algorithm is $O(k + k \times p)$. Since $k \leq p$, the complexity of the algorithm is $O(p^2)$.

Hence it is proved that the unweighted CT elimination decision problem is in NP. □

We now proceed to prove that not only the unweighted CT elimination decision problem is in NP but the problem is also NP-complete. In order to do so, we first transform the given instance of a CT elimination decision problem (presented in the form of an adjacency graph $G_{ad}(V_{ad}, E_{ad})$ containing a set of CTs along with the edge set of each CT), to another graph, called the complex triangle adjacency (CT adjacency) graph $G_{ct}(V_{ct}, E_{ct})$. The CT elimination decision problem is then restated in terms of this CT adjacency graph G_{ct}. It is then shown that the node cover decision problem, a well-known NP-complete problem, can be reduced to the unweighted CT elimination decision problem. The steps for constructing the CT adjacency graph $G_{ct}(V_{ct}, E_{ct})$ from the given adjacency graph $G_{ad}(V_{ad}, E_{ad})$ are as follows.

(i) For each CT_i of G_{ad}, include a vertex v_i in V_{ct}. In other words, each CT of G_{ad} is represented in G_{ct} by a vertex.
(ii) For each pair of vertices (v_i, v_j) of G_{ct}, add an edge (v_i, v_j), labelled with e_{ij}, if and only if e_{ij} is an edge shared by CT_i and CT_j, where CT_i and CT_j are the CTs correspond to v_i and v_j, respectively.

The CT adjacency graph shown in Fig. 5*a*(i) is shown in Fig. 5*a*(ii). The list of CTs and the associated edges of Fig. 5*a*(i) is given in Table 1. The CT adjacency graph and the list of CTs and the associated edges of Fig. 5*b*(i) is shown in Fig. 5*b*(ii) and Table 2, respectively. The complexity of the aforesaid algorithm can be determined by making use of the following theorem.

Theorem 2: A planar triangulated graph (PTG) with n vertices ($n \geq 4$) can have at most $(n - 3)$ CTs.

Table 1: List of CTs in the adjacency graph of Fig. 5*a*(i)

CT	CT	Exterior edge list
1	abca	ab, bc, ac
2	bcfb	bc, bf, cf
3	bcdb	bc, bd, cd
4	acea	ac, ae, ce

Table 2: List of CTs in the adjacency graph of Fig. 5*b*(i)

CT	CT	Exterior edge list
1	ajka	ak, aj, jk
2	ijki	ij, ik, jk
3	ikli	ik, il, kl
4	bclb	bc, bl, cl
5	fhlf	fh, fl, hl
6	fghf	fg, fh, gh
7	ghng	gh, gn, hn
8	fgnf	fg, fh, gn
9	efne	ef, en, fn
10	demd	de, dm, em

Proof: We prove the theorem by the method of contradiction. Let G(V, E) be a PTG with n vertices and more than k CTs. Specifically, let G have k CTs such that $k > n - 3$. Among all the CTs, we select an innermost CT, i.e. a CT that does not contain any other CT within it. Suppose v_i, v_j and v_k are the outer vertices of the selected CT and v_l is the vertex inside the triangle defined by v_i, v_j, v_k. If we remove v_l from G, exactly one CT will be removed from G. The resulting PTG, G$'$, will consist of $n' = n - 1$ vertices and $k' = k - 1$ CTs. By applying the process repeatedly, always choosing an innermost CT at each step, we can remove all CTs from G and at each step of this procedure exactly one vertex will be removed from G. Eventually we obtain a PTG, say G_0, without any CT at all. If n_0 is the number of vertices in G_0, then $n_0 = n - k < n - (n - 3) = 3$, or $n_0 < 3$. This is a contradiction because a PTG cannot be formed with less than three vertices.

Since there are at most $(n - 3)$ CTs in the adjacency graph, step (i) of the algorithm runs $(n - 3)$ times in the worst case. Step (ii) runs $(n - 3) \times (n - 4)/2$ times in the worst case. Therefore the worst case complexity of the algorithm is $O(n^2)$. □

Two important observations regarding the CT adjacency graphs are:

(a) Each isolated CT of the adjacency graph maps to an isolated vertex of the CT adjacency graph.
(b) If there exists an edge, e, shared by l number of CTs, say $CT_1, CT_2, \ldots, CT_l$, then there exists a clique of size l in the CT adjacency graph consisting of the corresponding l number of vertices. Each edge of this clique will be labelled e.

In a CT adjacency graph $G_{ct}(V_{ct}, E_{ct})$ an edge-label e is said to cover a vertex $v \in V_{ct}$ if there exists an edge in G_{ct} that is incident on v and is labelled e. For example, in Fig. 5*a*(ii), the edge-label *ac* covers the vertex 4. Vertex 1 is covered by both *ac* and *bc*.

In the context of the CT elimination optimisation problem, the isolated CTs do not have any significance. This is because, if $E' \subseteq E$ is a set of edges containing at least one edge of each CT, then it must contain exactly one edge from each of the isolated CTs. Therefore whatever optimisation we wish to achieve, we have to obtain it using the remaining CTs. If we ignore the isolated CTs, the corresponding isolated vertices may be removed from the CT adjacency graph.

The CT elimination problem can now be restated in terms of the CT adjacency graph. As for the construction of $G_{ct}(V_{ct}, E_{ct})$ from $G_{ad}(V_{ad}, E_{ad})$, a set of edges of G_{ad} which contains at least one edge from each CT of G_{ad} corresponds to an edge-label cover of G_{ct}. For example, consider the adjacency graph of Fig. 5*a*(i) and the corresponding CT adjacency graph, shown in Fig. 5*a*(ii). The CTs of Fig. 5*a*(i) and the outer edges of each CT are listed in Table 1. The minimal set of edges of Fig. 5*a*(i) such that each CT has at least one edge in it is $\{bc, ac\}$. This is the minimal edge-label cover of the CT adjacency graph shown in Fig. 5*a*(ii).

Therefore, the CT elimination problem now reduces to the problem of finding a minimal edge-label cover of G_{ct}. The corresponding decision problem is given below.

Edge-label cover decision problem

Given a CT adjacency graph $G_{ct}(V_{ct}, E_{ct})$ and an integer k, is there an edge-label cover E_{etc} such that $|E_{elc}| \le k$?

Theorem 3: The edge-label cover decision problem is NP-complete.

Proof: The theorem can be proved by reducing the *node cover decision* (NDC) problem to the edge-label cover problem with the help of a deterministic algorithm that has a polynomial time complexity.

Construction: Given a graph G = (V, E), let us construct a dual graph G$'$(V$'$, E$'$) in the following way:

i) For each edge $e \in E$, include a vertex $v(e) \in V'$.
ii) For each pair of vertices $v_1(e_1)$ and $v_2(e_2)$ of V$'$ corresponding to the edges $e_1, e_2 \in E$ perform step (iii).
iii) If there exists a vertex $x \in V$ such that both the edges e_1 and e_2 are incident on x, then include an edge between $v_1(e_1)$ and $v_2(e_2)$ of V$'$ and label that edge by x.

For example, the G$'$(V$'$, E$'$) graphs for the graphs of Fig. 4*a* and Fig. 4*b* are shown in Fig. 9*a* and Fig. 9*b*.

Complexity analysis: If n_e is the number of edges in G then step (i) will be executed n_e times. Step (ii) and Step (iii) will be executed $n_e \times (n_e - 1)/2$ times. Therefore the worst case complexity of the algorithm is $O(n_e^2)$.

Theorem 4 presented below establishes that the node cover decision problem can be reduced to the edge label cover decision problem in polynomial time.

Theorem 4: A graph G(V, E) has a node cover of size k iff there exists an edge-label cover of size $(k - i)$ in the corresponding transformed graph G$'$(V$'$, E$'$), where i is the number of isolated vertices in G$'$.

Proof:
(i) *Only-if part:* We assume that G(V, E) has a node cover of size k and G$'$(V$'$, E$'$) is the corresponding transformed graph with i number of isolated vertices. Each of these isolated vertices of G$'$ corresponds to an isolated edge of G. Therefore, G(V, E) must have i number of isolated edges. Obviously, any node cover of G must contain i number of nodes corresponding to these isolated edges, and the rest of

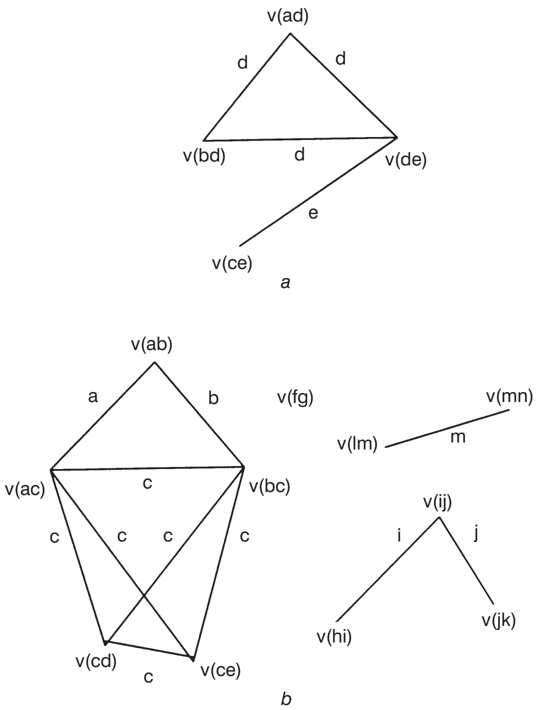

Fig. 9 *Dual graphs of Fig. 4a and Fig. 4b (v(xy) is the vertex corresponding to the edge (x, y) of G(V, E))*

a Dual of Fig. 4*a*
b Dual of Fig. 4*b*

the $(k - i)$ nodes should come from the rest of the graph. Without loss of generality, we may remove the isolated edges from G and the corresponding isolated vertices from G′. Now, we have to show that if G(V, E) (devoid of isolated edges and nodes) has a node cover of size k', then G′(V′, E′) (devoid of isolated nodes) must have an edge-label cover of size k'.

Since there is no isolated edge in G, each edge in G is adjacent to at least one other edge. This implies that corresponding to each edge e in G there is a node v adjacent to it such that $d(v) \geq 2$, where $d(v)$ is the degree of the vertex v. In other words, if there exists a node cover, $S \subseteq V$ of size k, then there is a node cover $S' \subseteq V$ such that $|S'| \leq k$. This is obtained simply by replacing each $v \in S$ with $d(v) = 1$ by the vertex v', at the other end of the edge. That v' has a degree of at least 2 is guaranteed by the fact that the edge (v, v') is not isolated. If v' is already in S, then drop v from S. We are justified in doing so because $S - \{v\}$ still covers G. Since each node $v \in S$ now has $d(v) \geq 2$, i.e. v is shared by at least two edges of G, each v must appear in G′(V′, E′) as edge-labels. This is similar to the construction of G′ from G.

Let $S' = \{v_1, v_2, \ldots, v_p\}$, $p \leq k$ be the node cover of G obtained in this manner. Let $v_i \in S'$ cover l edges $e_1, e_2, \ldots, e_l \in E$. Since v_i is a common vertex between any pair e_i, $e_j \in \{e_1, e_2, \ldots, e_l\}$, the edges will map to a clique of size l with vertices $v_1(e_1)$, $v_2(e_2), \ldots, v_l(e_l)$ in G′. Each of the $l(l+1)/2$ edges of this clique will be labelled v_i. Therefore, in G′, the edge-label v_i will cover all vertices $v_1(e_1)$, $v_2(e_2), \ldots, v_l(e_l)$. The above analysis is valid for any $v_i \in S$. Since all edges of G are covered by S, the set of edges of G′ labelled by the same set of vertices of S will cover precisely all vertices of G′. For example, in Fig. 4*a*, $\{d, e\}$ is a node cover of size 2. The corresponding dual graph shown in Fig. 9*a* has an edge-label cover $\{d, e\}$ of size 2. In Fig. 4*b*, ignoring the isolated edge fg, we have a node cover $\{a, c, m, i, j\}$ of size 5. The dual graph of Fig. 4*b*, shown in Fig. 9*b*, also has an edge-label cover $\{a, c, m, i, j\}$.

(ii) *If part*: Suppose that $S' = \{e_1, e_2, \ldots, e_k\}$ is an edge-label cover of G′. Let $C(e_l) = \{v_{l1}, v_{l2}, \ldots, v_{li}\}, \ldots, C(e_k) = \{v_{k1}, v_{k2}, \ldots, v_{kj}\}$ be the vertices of G′ covered by the edge-labels $e_1, e_2, \ldots, e_k$ respectively. Now $\{v_{l1}, v_{l2}, \ldots, v_{li}\} \cup \cdots \cup \{v_{k1}, v_{k2}, \ldots, v_{kj}\} = V'$. Since each vertex $v \in V'$ corresponds to an edge in E and each edge-label in G′ corresponds to a node shared by more than one edge in G, the set $\{v(e_1), v(e_2), \ldots, v(e_k)\} \subseteq V$ is a node cover of G. □

Hence it has been proved that the node cover decision problem can be reduced to the edge-label cover decision problem in polynomial time. This establishes the fact that the edge-label cover decision problem is NP-complete which, in turn, proves that the unweighted complex-triangle (CT) elimination problem for adjacency graphs with an arbitrary level of containment is NP-complete.

4 Conclusion

In this paper we have advanced the claim that the unweighted complex-triangle (CT) elimination problem for adjacency graphs with an arbitrary level of containment is NP-complete and have presented a proof in support of our claim. The elimination of all complex triangles is an indispensable step towards the floor-planning of a circuit presented in the form of an adjacency graph using the rectangular graph dualisation approach. Therefore the proof regarding the NP-completeness of the problem has far-reaching significance. Future work on this problem will focus on finding efficient heuristics to generate an optimal or near-optimal solution to the CT elimination problem.

5 Acknowledgment

The authors wish to thank the anonymous referees for their insightful comments which helped to improve the paper.

6 References

1 LENGAUER, T.: 'Combinational algorithms for integrated circuit layout' (John Wiley and Sons, New York, 1990)
2 YEAP, K.H., and SARRAFZADEH, M.: 'Floor planning by graph dualization: 2-concave rectilinear modules', *SIAM J. Comput.*, 1993, **22**, (3), pp. 500–526
3 KOZMINSKI, K., and KINNEN, E.: 'Rectangular dual of planner graphs', *Networks J.*, 1985, **15**, pp. 145–157
4 KOZMINSKI, K., and KINNEN, E.: 'Rectangular dualization and rectangular dissection', *IEEE Trans. Circuits Syst.*, 1988, **35**, pp. 1401–1416
5 VHASKAR, J., and SAHANI, S.: 'A linear algorithm to find a rectangular dual of a planer triangulated graph', *Algorithmica*, 1988, **3**, pp. 247–278
6 YEAP, GARY, K.H., and SARRAFZADEH, M.: 'Sliceable floor-planning by graph dualization', *SIAM J. Disc. Math.*, 1995, **8**, (2), pp. 258–280
7 YEAP, K.-H., and SARRAFZADEH, M.: 'A theorem of sliceability'. Second Great Lakes Computer Science Conference, 1991, Western Michigan University, Kalamazoo, MI
8 SUN, Y., and YEAP, K.H.: 'Edge covering of complex triangle in rectangular dual floorplanning', *J. Circuits, Syst. Comput.*, November 1993
9 TSUKIYAMA, S., KOIKE, K., and SHIRAKAWA, I.: 'An algorithm to eliminate all complex triangles in a maximal planner graph for use in VLSI floor-plan'. IEEE International Symposium on *Circuits and systems*, 1986, pp. 321–324
10 GAREY, M.R., and JOHNSON, G.J.: 'Computers and intractability' (W. H. Freeman and Co., 1979)