# A Unified Approach to Topology Generation and Area Optimization of General Floorplans

Partha S. Dasgupta
Computer Center
Indian Institute of Management
Calcutta 700 027, India
partha@iimcal.ernet.in

Susmita Sur-Kolay
Dept. of CSE
Jadavpur University
Calcutta 700 032, India
dgd@jadav.ernet.in

Bhargab B. Bhattacharya
Electronics Unit
Indian Statistical Institute
Calcutta 700 035, India
bhargab@isical.ernet.in

## Abstract

*In this paper, it is shown that for any rectangularly dualizable graph, a feasible topology can be obtained by using only either straight or Z-cutlines recursively within a bounding rectangle. Given an adjacency graph, a potential topology, which may be nonslicible and is likely to yield an optimally sized floorplan, is produced first in a top-down fashion using heuristic search in AND-OR graphs. The advantage of this technique is four-fold: (i) accelerates top-down search phase, (ii) generates a floorplan with minimal number of nonslice cores, (iii) ensures safe routing order without addition of pseudo-modules, and (iv) solves the bottom-up algorithm efficiently for optimal sizing of general floorplans in the second phase.*

## 1 Introduction

Floorplanning is important not only for layout synthesis but even more during the early design stages for deciding which design alternatives will probably yield optimal quality. Finding a suitable topology and determining the dimensions and aspect ratios of the modules in order to satisfy certain optimization criteria such as area, perimeter of bounding rectangle, total wire length, etc., is the *floorplan optimization problem.*

Constructive floorplanning approaches based on graph dualization, [3, 4] generate *only a feasible topology* for a given adjacency graph in polynomial time without considering any optimization criteria. The number of topologies for a given interconnection pattern may be exponential [9]. The iterative approaches *assume a given topology* and try to find an optimum implementation, given a set of aspect ratios; module adjacencies may be altered. The problem of minimizing area or perimeter for a given topology is known as the *sizing problem.* Time complexity of this problem is polynomial for slicing floorplans and NP-complete for general ones [4]. Encouraging results have been obtained for sizing by branch-and-bound method, simulated annealing and generalization of Stockmeyer's algorithm for general floorplans [4, 5], and for hierarchical nonslicible floorplans of order 5 in [6, 10].

Techniques reported in [3, 12] provide *an integrated approach* to obtaining an area-optimal slicible floorplan from a given adjacency graph. Several slicing trees are enumerated in [12] and the minimum area for each slicing tree is evaluated. A better integration is presented in [3] where sizing features are utilized during slicing topology generation. For inherently nonslicible (INS) [8] and some slicible rectangular graphs, the method in [12] forcibly produces slicings by introducing pseudo-modules, whereas that in [3] outputs slicings always except for INS graphs where it aborts.

In this paper, a unified approach is attempted for general floorplans by (i) generation of a potential floorplan topology that is likely to have both a minimum area implementation and a minimal number of nonslice cores; (ii) finding the minimum area implementation for the topology generated. Since the solution space of potential topologies is huge, a well-known AND-OR heuristic search algorithm [1] is adapted with novel heuristic estimates to reduce the search effort. The decomposition is based on our result that for any rectangularly dualizable adjacency graph, a floorplan can always be found by considering only slices or Z-cuts recursively. Moreover, such a floorplan has minimal number of nonslice cores and a channel definition with safe routing order. All channels are monotone [7], i.e., either straight or Z-shaped, and number of Z channels is minimal.

In the second phase, the method reported in [10] is extended to determine an optimal sizing efficiently. The proposed method outperforms existing approaches both in time and in area.

## 2 Topology using Slices and Z-cuts

**Preliminaries:** A *floorplan* is a rectangular dissection of a bounding rectangle by isothetic line segments called cuts. The four sides of the bounding rectangle are denoted by North, East, South and West. A floorplan is said to be *slicible* if it is obtained recursively by using through-cuts or *slices* only, otherwise it is nonslicible (Fig. 1a). Formally, a floorplan is a plane graph $G_d$ with each edge either horizontal or vertical. The four degree-2 vertices on the outermost cycle are denoted by NW, NE, SE, SW; all other vertices are of degree-3. In the *rectangular dualization* method [4, 9], interconnection requirements in a floorplan are represented by an adjacency graph $G = (V, E)$, where the vertex set $V$ corresponds to the modules, and an edge $(u, v) \in E$ implies adjacency of the modules $u$ and $v$. An adjacency graph is called *rectangularly dualizable (rectangular)* if there exists a floorplan corresponding to it. A *complex n-cycle* of a plane graph is a cycle of length $n$ such that there exists at least one vertex in the finite region bounded by the cycle; this is denoted by $C_{n,m}$ where $n$ = length of the cycle and $m$ = number of vertices interior to the cycle.

A rectangular graph may have more than one *equivalent* floorplan realizations which may or may not be slicible. Rectangular graphs which do not have any slicible realization are known as *Inherently Nonslicible (INS)* [9]. A *canonical embedding* [9] of a floorplan is an equivalent floorplan with the minimum number of nonslice cores.

A *cut* in the floorplan is a path in $G_d$ between either an exterior vertex of North (West) side and one of South (East) side. The corresponding cut-set of edges in $G$ decomposes it into exactly two non-empty subgraphs $G_l$ and $G_r$. The two sequences of vertices on the two sides of these cut-edges of $G$, called the two boundary paths $P_l$ and $P_r$ of the cut, are essentially the new boundaries of the sub-floorplans on the two sides of the cut, corresponding to

$G_l$ and $G_r$. A *chord free path* (CFP) [11] in $G$ is a path $P = < v_1, v_2, \ldots, v_n >$ where for all $i \neq j$ : a) $v_i \neq v_j$ and, b) if $(v_i, v_j) \in G$, then $|i - j| = 1$. If $P$ is not *CFP*, then it has two vertices $v_i$ and $v_j$ such that $|i - j| > 1$ and the edge $(v_i, v_j)$ is in $G$; such an edge is called a *chord* of $P$. A chord $(v_i, v_j), i < j$ is called a *maximal chord* of $P$, if there is no other chord $(v_k, v_l)$ where $k \leq i < j \leq l$. If both of its boundary paths of a cut are chord free, then it is called a *slice* in a floorplan and both $G_l$ and $G_r$ are rectangular. A *vertical (horizontal) cut* is between North and South (East and West) sides. The cut-set in $G$ corresponding to a slice in a floorplan is referred to as a slice in $G$. Hence, a floorplan of a given $G$ can be obtained in a top-down fashion by recursively finding slices among several possible cuts in the floorplan. But for nonslicible floorplans, slices may not exist at some levels of recursion.

## 2.1 Z Cuts and their properties

Although complete characterization of slicibility of a rectangular graph is unknown, many sufficient conditions for slicibility and properties of slicible floorplans appear in earlier literature [8, 11]. Some new properties related to nonslicible floorplans are presented below. Let $F$ be a floorplan and $G$ the corresponding rectangular graph.

*Definition 1:* A generalized k-cut in $F$ is a cut for which the number of maximal chords on either of its boundary paths is at most $k$.

*Definition 2:* A Z-cut in $F$ is a generalized k-cut with $k = 1$.

A cut-set in $G$ for a generalized k-cut in $F$ is referred to as a k-cut of $G$. A slice is a generalized k-cut with $k = 0$. Any generalized k-cut in $F$ has a rectilinear embedding with $2k$ bends or intermediate corners, and decomposes $F$ into two sub-floorplans each having a rectilinear polygonal boundary with $2k + 4$ corners. It is easy to see that any generalized k-cut can always be rectilinearly embedded with $2k'$ bends for any $k' > k$ . Floorplans with more than 4 corners are called *rectilinear floorplans.* Both the sub-floorplans for a slice and a Z-cut (Fig. 1a) have rectangular and L-shaped boundaries respectively. An L-module has one concave and five convex corners. Four of these convex corners are denoted by NW, NE, SE and SW in a clockwise manner, where the NW corner is conventionally the leftmost one with maximum height; the remaining two corners are labeled as MW (mid-west) and ME (mid-east). Note that indivisible modules in $F$ are rectangular, not L-shaped or 2-concave rectilinear.

For the sake of efficiency in optimal sizing and existence of safe channel routing order, it is desirable to have a minimum number of nonslice cores in nonslicible floorplans. Moreover, it was shown in [9] that a canonical embedding has either 0 or 1 nonslice core at each level of its unique maximal rectangular hierarchy (MRH). One nonslice core within a maximal rectangle implies the existence of a Z-cut. Thus any canonical embedding can be obtained by recursively subdividing a rectangle with slices or Z-cuts depending on whether there are 0 or 1 nonslice cores at that level of its MRH. However, the non-uniqueness of canonical embeddings leads to the observation that for a Z-cut in $F$, the corresponding cut-set in $G$ may be a slice of $G$. Hence obtaining a particular nonslicible canonical embedding from a given $G$ in a top-down fashion is non-trivial. On the other hand, the following pertinent question can be raised:*for any G, does there exist any rectangular dual which can be recursively decomposed by using slices or Z-cuts only?* The affirmative answer follows from the lemmata below.

*Lemma 1:* For any rectilinear floorplan, there exists an equivalent rectilinear $F_{rect}$ having a slice.

*Lemma 2:* Let $G$ be a rectangular graph, $C_v$ be a k-cut in $G$ which decomposes it into $G_l$ and $G_r$, and $P_l(C_v)$ and $P_r(C_v)$ be the corresponding boundary paths in $G$ on left and right of $C_v$. If exactly one of $P_l(C_v), P_r(C_v)$ has a chord in it, then both $G_l$ and $G_r$ admit rectilinear floorplans.

Lemma 1 is proved by applying a theorem in [11] stating that if a $C_{4,n}$ exists in $G$, a slice may not exist. Generalized k-cuts may be necessary to rectangularly dualize it. Fig. 1b shows a $C_{4,1}$; recursive bipartitioning of the rectangular graph yields the floorplan of Fig. 1a with slices and Z-cuts only. Therefore:

**Observation 1.** In any rectangular floorplan realization of a $C_{4,1}$ with four distinct corner vertices, generalized k-cuts with $k \leq 1$ are sufficient to decompose it recursively. There may be more than one possible Z-cut in $C_{4,1}$.

*Lemma 3:* In a rectangular graph, the rectangular dualization problem for the subgraph interior to a $C_{4,k}$ in $G$, can be considered independent of that of the rest of $G$.

*Lemma 4:* In a complex 4-cycle $C_{4,n}(n \geq 1)$, if all four of its exterior vertices are chosen as corners, a Z-cut can always be obtained.

Hence the following important theorem:

*Theorem 1:* For obtaining a rectangular floorplan from a rectangular graph using recursive bipartitioning, it is sufficient to use generalized k-cuts, where $k \leq 1$, (i.e., slices and Z-cuts only).

The above theorem suggests that a rectangular dual for a given $G$ can be obtained by a hierarchical bipartitioning scheme using slices and Z-cuts; subgraphs interior to complex 4-cycles in $G$ can be independently dualized. Only Z-cuts are used for complex 4-cycles. Moreover, from Lemma 4 these Z-cuts do not partition the subgraph interior to the complex 4-cycle. Such special Z-cuts, called *sZ-cuts*, are used in the proposed method in order to prune the search space to a moderate size even for the general floorplans. In practice, the sZ-cuts may be used only when the four corners chosen for a complex 4-cycle are distinct. For rectangular graphs without any complex 4-cycle [11], our method always produces a slicible floorplan. For graphs with complex 4-cycles, it may or may not produce slicible floorplan, depending on the sizing constraints. **Observation 2:** The number of nonslice cores, i.e., the number of sZ-cuts in the topology generated is minimal.

The rectangular dual produced by this scheme can be represented by a *binary floorplan tree*, $T_F$ where the root represents the bounding rectangle and a cut in it, an interior vertex is either a rectangle with a slice or a Z-cut in it or an L-shaped module with a slice in it, and the leaves are the indivisible rectangular modules. Furthermore, the Z-cuts and slices at each level simultaneously yield a unique channel definition for the topology generated. Nonslicible topologies with straight channels do not have a safe routing order, since the channel dependency digraph has directed cycles. Channel definition based on slices and Z-cuts only lead to an acyclic channel dependency graph, as shown below.

Let $CG_F$ be the channel dependency graph defined by sZ-cuts and slices obtained by applying Theorem 1. Let $F$ be the floorplan topology produced by our top-down scheme.

*Lemma 5:* (a) For any sZ-cut in $T_F$, there is a unique composite rectangle in $F$ which is partitioned by the sZ-cut.

(b) In $CG_F$, there is no common directed cycle for the vertices corresponding to two sZ-cuts on different levels of $T_F$.

(c) In $CG_F$ of the unique composite rectangle for any sZ-cut, there are no directed cycles.

The theorem below follows.

*Theorem 2:* $CG_F$ has a safe routing order.

## 2.2 AND-OR graph search

In our two-phase method for floorplan optimization, the first phase of topology generation is formulated as an AND-OR graph search. The entire set of possibilities explored in order to achieve the solution to a problem, is either a network or a tree and is called the *search graph.* A cost function is associated with each arc or each node of the search graph which contributes to the cost of the solution(s). In the *decomposition* scheme, a given problem is recursively divided into several subproblems until no further decomposition is possible. At any stage there can be many possible decompositions. The root problem can then be reconstructed in several ways by considering the possible combinations of the subproblems. Such an approach is generally represented by an *AND-OR graph*, where each node is either an AND node representing an actual decomposition or an OR node representing a set of possible decomposition strategies. A solution is a subgraph of the AND-OR graph, and called a solution graph. In a combinatorial optimization problem, the objective is to find the solution subgraph with optimum cost for the root. Heuristic search methods exist [1] to tackle an NP-complete problem in this framework.

An optimal topology is determined by the top-down search algorithm *AO-FP*$^*$ based on *AO*$^*$, similar to [3]. In the AND-OR search graph, the root node $s$ represents the original adjacency graph with a chosen set of corners, each AND node corresponds to a bipartition of an adjacency graph or its subgraph, and each OR node is the set of possible bipartitions. The AND and OR nodes occur at alternate levels, having associated costs based on sizing constraints. Fig. 2 shows a portion of the AND-OR partitioning of an adjacency graph. A solution graph where each OR node has one successor, gives a binary floorplan tree $T_F$ and thus a topology (Fig. 3).

The work reported here being a generalization of [3], the salient developments over the previous work are outlined below. The major addition in the first phase is *determination of Z-cuts and the associated rectilinear corner assignments in the two subgraphs of the cut.* From Lemma 4, it follows that within a complex 4-cycle with four distinct corner vertices, Z-cuts are to be looked for, whereas in other cases, both slices and Z-cuts need to be searched. The criterion for maximal chord is tested incrementally during path-finding. Bipartition of composite modules is followed by a proper assignment of corners [2] to the submodules. There can be four different orientations for the L-modules due to rotation and reflection symmetry.

The definition of score of a cut, method of cost computation for any node in the AND-OR graph and Algorithm *AO-FP*$^*$ for the first phase described in [3], hold good. In order to improve the quality of solution, the partial cost computation method mentioned in [3] may also be employed here. It should be noted that the second component in the score of a cut, derived from the estimated range of dimensions along the cut to minimize dead (wasted) area in parent composite module, is considered for three independent segments in the case of a Z-cut.

# 3 Optimal Sizing

The second phase of our method achieves optimal sizing in a bottom-up fashion. Using the given set of possible implementations of the leaf modules of $T_F$ produced by the first phase, the implementations of its floorplan can be obtained by bottom-up post-order traversal of $T_F$ and combination of the implementation lists of children of an internal vertex to form that of their parent. An optimal implementation of the floorplan can then be obtained by selecting one with minimum area from the irredundant set of implementations for the root. Since area is a non-decreasing function of length and width, minimum area of the floorplan implies minimum dead area.

A module can have many possible implementations. An implementation of an L-module is represented by a 4-tuple $(w^1, w^2, h^1, h^2)$, where $w^1, w^2 (h^1, h^2)$ represent the lengths of the longer and shorter horizontal (vertical edges, respectively. A rectangle is a degenerate L-module with $w^1 = w^2$ and $h^1 = h^2$. For the definitions of redundant implementation of a module and R- (L-) lists for irredundant lists of implementations for a rectangular (L-shaped) module, the reader is referred to[10]. For convenience of processing and to have a higher degree of pruning, the implementations in an R-list are arranged in order of increasing height and decreasing width, and that in an L-list are in order of increasing heights of left vertical and right vertical edges, and decreasing width of the larger horizontal edge.

The non-slicing structures generated by the proposed method are equivalent to the 5-wheel or nested 5-wheel. The different structures of floorplans generated by our method indicate the presence of five types of vertices in $T_F$. A brief description of these types (Figs. 4a - 4e) is given below.

**Type A :** a rectangle formed by the combination of two smaller rectangles, as in Stockmeyer's algorithm.

**Type B :** an L-module formed by combining two rectangular modules ( $\alpha$-node [10] ).

**Type C :** an L-module obtained by joining a rectangular module along either of the longer edges of an L-module ( $\beta$-node [10] ).

**Type D :** an L-module obtained by joining a rectangular module along either of the shorter vertical edges of an L-module ( $\gamma$-node [10] ).

**Type E :** a rectangle obtained by attaching two L-modules with facing concave edges.

An irreducible R-list or L-list at a vertex of $T_F$ is obtained by appropriately combining the irreducible R-lists or L-lists of the successors. There are five basic procedures, called type $i$ procedure, $i \in \{A, B, C, D, E\}$ for constructing a set of irreducible R-lists or L-lists for a type $i$ vertex. The procedures for types $A, B, C, D$ appear in the references mentioned in the above paragraph and that for the new type $E$ is briefly described below.

Let $m$ be a vertex of type $E$ with its successors $m_1$ and $m_2$ representing two L-modules which are combined to form a rectangle. The dimensions of $m_1$ and $m_2$ are given in two sets of L-lists $\mathcal{L}_{m_1} = \{L^1_{m_1}, L^2_{m_1}, \ldots, L^M_{m_1}\}$ and $\mathcal{L}_{m_2} = \{L^1_{m_2}, L^2_{m_2}, \ldots, L^N_{m_2}\}$. The dimensions of $m$ is formed by combining these L-lists sequentially and stored in a single R-list, which is pruned efficiently to remove redundant elements. If the $p$-th element of $L^i_{m_1}$ and the $q$-th element of $L^j_{m_2}$ are $(w^1_1, w^2_1, h^1_1, h^2_1)$ and $(w^1_2, w^2_2, h^1_2, h^2_2)$ respectively, the composite rectangle will have dimensions given by $(\max(w^1_1 + w^2_2, w^2_1 + w^1_2), \max(w^1_1 + w^2_2, w^2_1 + w^1_2), \max(h^1_1, h^1_2, h^2_1 + h^2_2), \max(h^1_1, h^1_2, h^2_1 + h^2_2))$ for vertical Z-cut and $(\max(w^1_1, w^1_2, w^2_1 + w^2_2), \max(w^1_1, w^1_2, w^2_1 + w^2_2), \max(h^1_1 + h^2_2, h^1_2 + h^2_1), \max(h^1_1 + h^2_2, h^1_2 + h^2_1))$ for horizontal Z-cut.

# 4 Experimental Results

The algorithm has been implemented in C on DEC Alphastation 250 4/266 with 266 MHz clock rate. Table I summarizes for some benchmark examples and INS graphs the performance in terms of total CPU time for both phases, search space requirements and wasted area. The results are highly encouraging. Moreover, the topologies obtained (some shown in Figs. 5-8), have safe routing order and a minimal number of Z-cuts (may not be minimum always). For slicible graphs, the method outputs ei-

**Table I : Experimental results on benchmarks – topology generation and sizing**

| Problem | # blocks | # implementations | Area obtained | Wasted (%) area | CPU time (secs.) | Distinct nodes generated | # sZ |
|---|---|---|---|---|---|---|---|
| Xerox [12] | 10 | $3^{10}$ | 30 | 0.00 | 0.008 | 82 | 0 |
| Ex. in [8] (Fig. 5) | 16 | $3^{16}$ | 90 | 0.00 | 0.27 | 790 | 2 |
| Ex. in [11] | 22 | $3^{22}$ | 11730 | 1.64 | 10.2 | 6437 | 0 |
| EX6 [10] (Fig. 8) | 24 | $2.03 \times 10^{16}$ | 1024 | 0.00 | 3.24 | 2373 | 5 |
| EX10 [10] | 40 | $4^{40}$ | 242 | 0.83 | 1121.34 | 88962 | 0 |
| Ex. in [8] (Fig. 6) | 18 | $3^{18}$ | 72 | 0.0 | 0.36 | 918 | 2 |
| EX2 [10] (Fig. 7) | 25 | $4^{25}$ | 160 | 6.67 | 2.76 | 2351 | 1 |
| EX5 [10] | 25 | $8^{25}$ | 638 | 6.33 | 29.28 | 2335 | 1 |

ther a slicible (entries 1,3,5 in Table I) or a nonslicible (entries 2 & 4) floorplan depending on the sizing constraints, with (near)-minimum wasted area. For INS graphs (entries 6,7,8), it always yields a nonslicible floorplan – in fact, a canonical embedding except for entry 6.

## 5 Conclusion

This paper demonstrates an AI-based graph search method to a well-known problem of VLSI layout design along with the result on sufficiency of slices and Z-cuts for topology generation. It has produced promising results; the bottom-up algorithm runs fast for the examples tested. Our search algorithm can be improved further by doing depth-first search for rectangular graphs having exponential number of possible slices at the root. An inadmissible heuristic may guarantee polynomial termination of search with some degradation in the solution quality. A minor modification in the path-search routine can deal with rectangular graphs with weighted adjacencies.

## References

[1] A. Bagchi and A. Mahanti, Admissible Heuristic Search in AND/OR Graphs, *Theoretical Computer Science*, 24, 1983, 207-219.

[2] P.S. Dasgupta, S. Sur-Kolay and B.B. Bhattacharya, *manuscript*, 1995.

[3] ——, VLSI Floorplan Generation and Area Optimization using AND-OR Graph Search, *Proc. 8th Intl. Conf. on VLSI Design*, 1995, 370-375.

[4] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout Design*, Wiley, 1990.

[5] P. Pan and C.L. Liu, Area Minimization for Floorplans, *IEEE TCAD*, Vol. 14, No. 1, 1995, 123-132.

[6] P. Pan, W. Shi, C.L. Liu, Area Minimization for Hierarchical Floorplans, *Proc. ICCAD*, 1994, 436-440.

[7] S. Sur-Kolay and B.B. Bhattacharya, The Cycle Structure of Channel Graphs in Nonslicible Floorplans and A Unified Algorithm for Feasible Routing Order, *Proc. ICCD*, 1991, 524-527.

[8] ——, On the family of Inherently Nonslicible Floorplans in VLSI Layout Design, *Proc. ISCAS*, June 1991, Singapore, 2850-2853.

[9] ——, Canonical Embedding of Rectangular Duals with Applications to VLSI Floorplanning,*Proc. DAC*, June 1992, 69-74.

[10] T.C. Wang and D.F. Wong, Optimal Floorplan Area Optimization, *IEEE TCAD*, Vol. 11, No. 8, 1992, 992-1002.

[11] K.H. Yeap and M. Sarrafzadeh, Sliceable Floorplanning by Graph Dualization, *SIAM J. on Discrete Mathematics*, May 1995.

[12] ——, A Unified Approach to Floorplan Sizing and Enumeration, *IEEE TCAD*, Vol. 12, No. 12, 1993, pp. 1858-67.

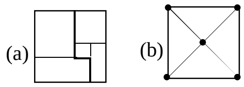

**Fig. 1:** sZ-cut Complex $C_4$

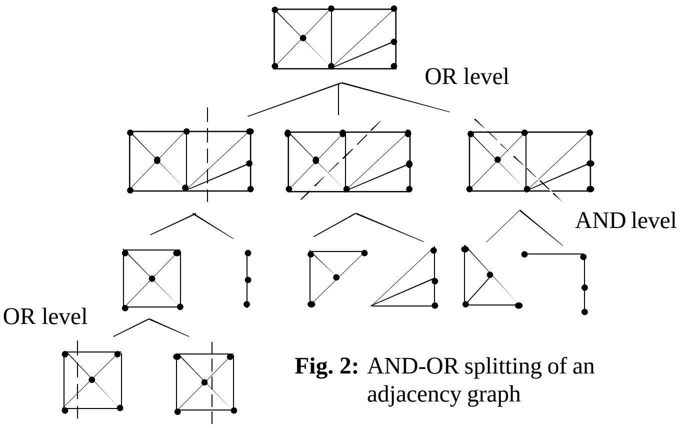

**Fig. 2:** AND-OR splitting of an adjacency graph

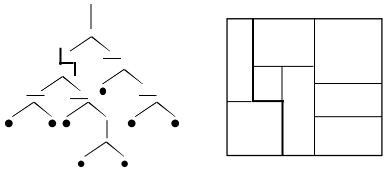

**Fig. 3:** Floorplan tree with slices and Z-cuts

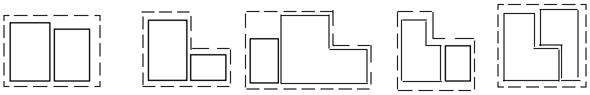

**Fig. 4 :** (a) Type A (b) Type B (c) Type C (d) Type D (e) Type E

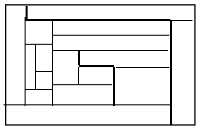

**Fig. 5 :** Example of [8]

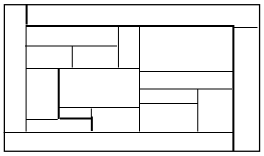

**Fig. 6:** Topology of an INS graph

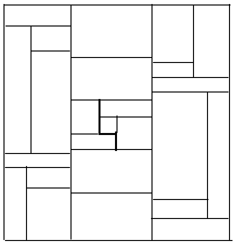

**Fig. 7:** EX 2 [10]

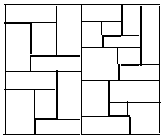

**Fig. 8 :** EX 6 [10]