Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

ELSEVIER

Computational Geometry 26 (2003) 143–171

Computational Geometry
Theory and Applications

www.elsevier.com/locate/comgeo

On finding an empty staircase polygon of largest area (width) in a planar point-set

Subhas C. Nandy *, Bhargab B. Bhattacharya

Advanced Computing and Micro-electronics Unit, Indian Statistical Institute, 203 B.T. Road, Calcutta 700 108, India

Received 17 October 2001; received in revised form 6 February 2003; accepted 14 February 2003

Communicated by J.-R. Sack

Abstract

This paper presents an algorithm for identifying a *maximal empty-staircase-polygon* (MESP) of largest area, among a set of n points on a rectangular floor. A staircase polygon is an isothetic polygon bounded by two monotonically rising (falling) staircases. A *monotonically rising staircase* is a sequence of alternatingly horizontal and vertical line segments from the bottom-left corner of the floor to its top-right corner such that for every pair of points $\alpha = (x_\alpha, y_\alpha)$ and $\beta = (x_\beta, y_\beta)$ on the staircase, $x_\alpha \leqslant x_\beta$ implies $y_\alpha \leqslant y_\beta$. A *monotonically falling staircase* can similarly be defined from the bottom-right corner of the floor to its top-left corner. An empty staircase polygon is a MESP if it is not contained in another larger empty staircase polygon. The problem of recognizing the largest MESP is formulated using permutation graph, and a simple $O(n^3)$ time algorithm is proposed. Next, based on certain novel geometric properties of the problem, an improved algorithm is developed that identifies the largest MESP in $O(n^2)$ time and space. The algorithm can be easily tailored for identifying the widest MESP in a similar environment. The general problem of locating the largest area/width MESP among a set of isothetic polygonal obstacles, can be solved easily. These geometric optimization problems have several applications to VLSI layout design, robot motion planning, to name a few.

Keywords: Geometric graph theory; Permutation graphs; Algorithms; Complexity

1. Introduction

The problem of finding an empty convex k-gon of maximum area or perimeter, amidst a set of points [1,4] is an important problem in computational geometry, and has lot of applications to computer-aided design. In VLSI layout design, isothetic polygons play a major role; the sides of these polygons are

* Corresponding author.
E-mail addresses: nandysc@isical.ac.in (S.C. Nandy), bhargab@isical.ac.in (B.B. Bhattacharya).

0925-7721/$ – see front matter
doi:10.1016/S0925-7721(03)00015-4

parallel to the coordinate axes [3]. Among a set of n point obstacles, the problem of locating the largest empty isothetic rectangle is studied extensively (see [2,6,20]). The best known algorithm for this problem runs in $O(n \log^2 n)$ time [2]. A related problem of locating all possible maximal-empty-rectangles among a set of isothetic rectangular obstacles can be solved in $O(R + n \log n)$ time, where R is the size of the output [21].

In this paper, we address the problem of recognizing the largest-area/width empty staircase polygon among a set of isothetic obstacles distributed on a rectangular floor. A staircase polygon is a special type of orthoconvex polygon [22]. Given a set P of n points as obstacles, we define a *maximal-empty staircase polygon* (*MESP*) as an empty isothetic polygon bounded by two monotone *staircase paths* from one corner of the floor to its diagonally opposite corner, such that no other empty staircase polygon completely inscribes it. The objective is to identify the largest area (width) MESP on the floor.

The problem is formulated using a permutation graph [14] whose nodes correspond to the points in P, and the edges capture the monotonicity property among the pairs of points. Each MESP is shown to correspond to a path between two designated nodes in a weighted digraph, called *staircase graph* (*SG*), which is obtained from the permutation graph. Each edge in the permutation graph corresponds to a node in *SG*; the edges of *SG* and their weights depend on the problem definition. A simple algorithm is first described that computes the largest area MESP as a maximum weighted path in the graph *SG*; its time and space complexities are $O(n^3)$ and $O(n^2)$, respectively. Later, it is observed that if the edges in the staircase graph are processed in a particular order, then the time complexity can be reduced significantly by exploiting certain geometric properties. We explored three important geometric properties, namely footprint propagation, inheritance, and dominance, and formulated our second algorithm. The time and space complexities of the improved algorithm are both $O(n^2)$.

A slightly different scheme can be used to solve the problem of finding a MESP whose width is maximum. The width of a MESP is defined in Section 4.1. A slightly modified algorithm can be used to locate the largest-area/width MESP among a set of *isothetic polygonal obstacles* with the same time and space complexities.

The paper is organized as follows. In Section 2, the motivation and possible applications of the problem are discussed. In Section 3, the problem of finding the largest-area MESP using permutation graph is introduced, and a simple algorithm is presented. In Section 4, the improved algorithm and the requisite data structures are presented. In Section 5, modified algorithms are outlined for finding the largest-width MESP, and for tackling polygonal obstacles. Concluding remarks appear in Section 6.

2. Applications

A VLSI floorplan is a rectangular dissection of a bounding rectangle with isothetic cut lines. Each indivisible cell (block) in the floorplan represents a circuit module. Two cut lines are assumed to meet at T-junctions only. A floorplan is said to be slicible if it is obtained by using isothetic cut lines as in binary space partitioning (see Fig. 1(a)), otherwise, it is non-slicible (see Fig. 1(b)). The cut lines denote the routing channels through which interconnecting wires are to be routed. If the floorplan is slicible, then hierarchical partitioning by the cut lines may be used to facilitate global routing [23]. For non-slicible floorplans, such a partitioning does not exist. Furthermore, directed cycles appear in the channel digraph, leading to infeasible channel routing order [17,23,25]. However, if the definition of channel is generalized to a monotone staircase polygon, the routing order always become acyclic, and

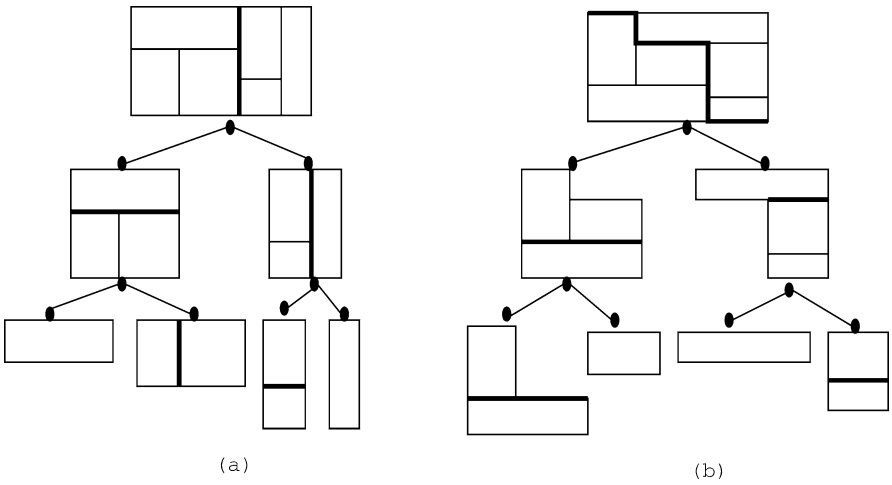


Fig. 1. (a) A slicible floorplan, (b) a non-slicible floorplan.

can be obtained easily by identifying staircase channels hierarchically (see Fig. 1(b)) [25]. At each step of recursion, the objective is to identify the largest-width channel so that maximum number of nets can be routed through that channel. This motivates us to study a more general theoretical problem of recognizing the largest area/width empty staircase polygon among a set of points or isothetic obstacles. In context to VLSI layout design, an obstacle represents a circuit block on silicon. The *width* of a staircase polygon is defined as the minimum euclidean distance (clearance) between the two stairs. A *MESP* among a set of non-overlapping isothetic polygonal obstacles can be defined in an analogous manner.

Recently, several problems involving staircase channels in VLSI layout design have been reported. The problem of identifying a staircase channel in a VLSI floorplan minimizing the number of crossing nets (i.e., the number of different nets whose terminals appear on both sides of the channel) can be solved in $O((n+k) \times T)$ time, where n, k, and T are respectively the number of blocks (rectangular circuit modules), the number of nets, and the number of terminals on the floor [18]. Another relevant problem is to find a staircase cut on the floorplan such that the two partitions consist of almost equal number of blocks, and simultaneously minimizing the number of crossing nets [19]. It is observed that for a floorplan with n blocks, the problem of finding a partition by a staircase channel that minimizes the difference in the number of blocks lying in the two halves, can be solved in $O(n)$ time [12]. However, staircase partitioning that minimizes the area difference is NP-hard even if minimization of the number of nets crossing the cut is not considered [19]. An efficient heuristic is reported in [13] based on acyclic graph search with unrestricted (positive or negative) edge cost. Further, efficient wire-routing algorithms through a staircase channel using Manhattan-diagonal model appear in [10,11].

Apart from classical channel routing, locating a MESP may find another application to the more recent repeater (buffer) placement problem. Buffer insertion is a very important problem in interconnect-driven floorplanning, and is needed to reduce delay and preserve signal integrity [7–9]. Empty staircase regions on silicon are considered for solving the buffer insertion problem [24].

The problem of recognizing the largest-width MESP is relevant to robot motion planning, where the objective is to navigate a circular robot through a staircase path. A related problem of identifying the widest empty L-shaped corridor amidst a set of point obstacles can be solved in $O(n^3)$ time [5].

3. Formulation of the problem

3.1. Definitions and preliminaries

Let $P = \{b, p_1, p_2, \ldots, p_n, t\}$ be a set of points, where $b = (0, 0)$ and $t = (x_t, y_t)$ are respectively the bottom-left and top-right corners of a rectangular floor, and $\{p_i = (x_i, y_i),\ i = 1, 2, \ldots, n\}$ denote n point-obstacles on the floor. For the sake of simplicity, we assume that the points in P are in general positions, i.e., for every two points p_i and p_j, $x_i \neq x_j$ and $y_i \neq y_j$. Henceforth, we shall refer a horizontal line passing through a point p_i by the line $Y = y_i$.

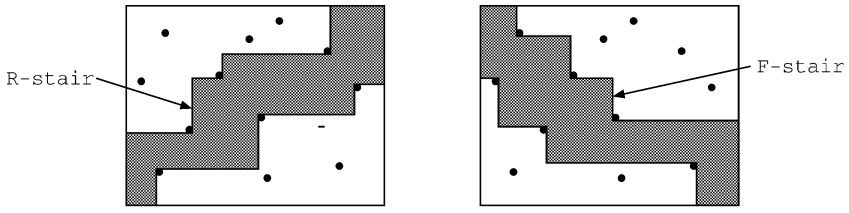
Definition 1. An *isothetic curve* is a rectilinear path consisting of alternating horizontal and vertical line segments. An isothetic curve is a *monotonically rising staircase* (*R-stair*) if for all pairs of points $\alpha = (x_\alpha, y_\alpha)$ and $\beta = (x_\beta, y_\beta)$ and on the curve, $x_\alpha \leqslant x_\beta$ implies $y_\alpha \leqslant y_\beta$. Similarly in a *monotonically falling staircase* (*F-stair*), for all pairs of points $\alpha = (x_\alpha, y_\alpha)$ and $\beta = (x_\beta, y_\beta)$ and on the curve, $x_\alpha \leqslant x_\beta$ implies $y_\alpha \geqslant y_\beta$.

Definition 2. A *staircase polygon* is an isothetic polygon either bounded by two R-stairs or by two F-stairs. The former one is called a *R-staircase polygon* and the latter one is a *F-staircase polygon*. A staircase polygon is *empty* if it does not include any point of P in its interior. An empty staircase polygon is said to be *maximal* (*MESP*) if it is not contained in another larger empty staircase polygon.

In Fig. 2, two types of MESPs among a set of points are shown. If the floor is rectangular, then the bottom-most and top-most edges of both the R-stairs of a *maximal* empty R-staircase polygon meet at the bottom-left and top-right corners of the floor, respectively. Similarly, the top-most and bottom-most edges of both the F-stairs of a *maximal* empty F-staircase polygon meet at the top-left and bottom-right corners of the floor, respectively. A corner point (q) between two adjacent edges of the staircase polygon is said to be *convex* (respectively *concave*) if the internal angle at q is $\pi/2$ (respectively $3\pi/2$). In a MESP, *each concave corner must coincide with some member in the set* $\{p_1, p_2, \ldots, p_n\}$. Furthermore, the perimeter of each MESP is the same and is equal to the perimeter of the bounding rectangular floor.

Theorem 1. *For any MESP, if the lower stair is fixed, the upper stair becomes unique, and vice-versa.*

Proof. Follows from maximality and emptiness of the polygon. □



(a) R-staircase polygon (b) F-staircase polygon

Fig. 2. Two types of staircase polygons.

3.2. Representing MESP using permutation graphs

In order to obtain the largest-area MESP, the largest-empty R-staircase polygon and the largest-empty F-staircase polygon are located separately, and then the larger of these two is reported. Here, we shall consider only the problem of finding the largest empty R-staircase polygon.

The points in P are labeled in increasing order of their y-coordinates. According to Theorem 1, the objective reduces to identifying an appropriate lower R-stair such that the corresponding polygon is empty and its area is maximum. It is easy to show that the total number of MESPs may be exponential in n. Thus, complete enumeration of all MESPs would not be computationally feasible. The graph-theoretic formulation of the problem is motivated by the fact that if $p_i = (x_i, y_i)$ and $p_j = (x_j, y_j)$ are a pair of consecutive concave corners on the lower stair of an empty R-staircase polygon, then $p_i, p_j \in P$; moreover if $x_i < x_j$ then $y_i < y_j$.

Let us now consider a directed graph $G = (V, E)$ with $V = \{p_i \mid p_i \in P\}$ and $E = \{(p_i, p_j) \mid (x_i < x_j)$ and $(y_i < y_j)\}$. The digraph G is acyclic, in which b (t) is the only node that has indegree (outdegree) 0. The indegree (respectively outdegree) of a node p_i is denoted by $in(p_i)$ (respectively $out(p_i)$). Figs. 3(a) and (b) show an example of a floor with the point obstacles and the corresponding digraph G. A directed path from b to t in G is called a *complete path*.

Lemma 1. *Every complete path in G determines a MESP uniquely, and every MESP corresponds to a unique complete path in G which indicates the lower stair of the MESP.*

Proof. Consider a complete path $\{b, p_1, p_2, \ldots, p_k, t\}$ in G. Clearly, there exists a unique R-stair whose concave corners coincide with the points $p_1, p_2, \ldots, p_k$. The MESP whose lower stair is the above R-stair, is unique by Theorem 1. Conversely, for every MESP, the set of points which appear on the concave corners of its lower R-stair, are members of P, and they form a complete path in G. □

Example. The complete path $b \to p_1 \to p_2 \to t$ in the graph of Fig. 3(b), represents a MESP which is shown in Fig. 3(d).

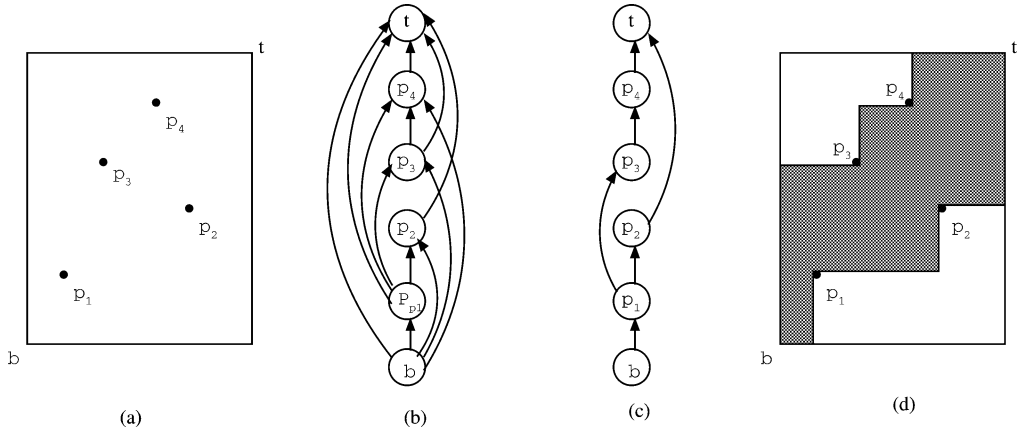


Fig. 3. (a) Example of a floor with point-obstacles, (b) the corresponding digraph G, (c) G_T: the transitive reduction of G, and (d) the MESP corresponding to the complete path $b \to p_1 \to p_2 \to t$.

The digraph G is essentially a transitively-oriented permutation graph, as defined below.

Definition 3 [14]. Suppose $\pi = [\pi_1, \pi_2, \ldots, \pi_N]$ is a permutation of the numbers $1, 2, \ldots, N$. Denote by π_i^{-1} as the position of i in π. The *permutation graph* $G_\pi = (V_\pi, E_\pi)$ is an undirected graph with $V_\pi = \{1, 2, \ldots, N\}$; an edge $(i, j) \in E_\pi$ implies $(i - j)(\pi_i^{-1} - \pi_j^{-1}) < 0$, and vice-versa.

Lemma 2. *The undirected version of the digraph G is a permutation graph.*

Proof. Let us consider the floor with n points along with the bottom-left and the top-right corners of the floor, and the graph G produced from it. Let the points be labeled as $\{1, 2, \ldots, N\}$ $(N = |P| = n + 2)$ in increasing order of their y-coordinates. Now, a line is swept from the right boundary of the floor to its left boundary, and the labels of points, as they appear, are recorded. The sequence of labels, thus observed, plays the role of π. □

A permutation graph is known to be transitively orientable, i.e., $(i, j) \in E_\pi$ and $(j, k) \in E_\pi$ implies that $(i, k) \in E_\pi$ [14].

As a matter of fact, the digraph G, defined as above, captures the transitive orientation. The transitive reduction $(G_T = (V, E_T), E_T \subseteq E)$ of the digraph G in Fig. 3(b), is shown in Fig. 3(c).

Definition 4. For a pair of points p_i and p_j with $(p_i, p_j) \in E$, the rectangle with p_i and p_j at its bottom-left and top-right corners, respectively, is denoted as *rectangle*$[p_i, p_j]$.

Observation 1. *For an edge $(p_i, p_j) \in E$, if $(p_i, p_j) \in E_T$ then rectangle$[p_i, p_j]$ will contain no point of P in its interior, and if $(p_i, p_j) \notin E_T$ then rectangle$[p_i, p_j]$ contains at least one member of P in its interior.*

The number of complete paths in graph G, may be exponential in n in the worst case, and so is the number of MESPs. Our aim is to locate an appropriate complete path in G as the lower stair of a MESP, such that its area is maximum among all the MESPs on the floor.

3.3. MESP as the max-weight path problem

Since every MESP corresponds to a directed complete path in the digraph G, identification of the largest(area)-MESP seems to have a natural formulation as a max-weight path problem in a weighted digraph. Unfortunately, a weighted version of the digraph G capturing the essence of the problem cannot be constructed, as we shall demonstrate shortly. However, a new digraph called the *staircase graph* derived from G, will suit our purpose.

3.3.1. Primitive staircase polygon

A staircase polygon can be partitioned into a set of smaller or primitive staircase polygons, called *L_polygons*, as described below.

Definition 5. Let $\alpha = (x_\alpha, y_\alpha)$ and $\beta = (x_\beta, y_\beta)$ be two points on the floor, such that $x_\alpha < x_\beta$, $y_\alpha < y_\beta$ (α, β may or may not belong to P). An *L_path*(α, β) is a rectilinear path from the point α to the point β with exactly one corner at (x_β, y_α).

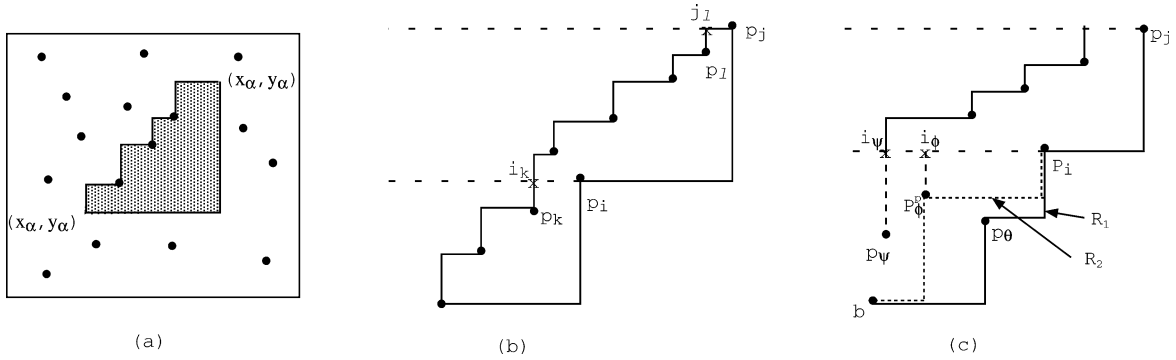


Fig. 4. (a) L_polygon, (b) subpolygon of a MESP corresponding to an L_path in the lower stair, (c) motivation of footprints.

Definition 6. The largest empty-staircase-polygon whose lower stair is the $L\_path(\alpha, \beta)$ and the upper stair is a rising-staircase from α to β is denoted as $L\_polygon[\alpha, \beta]$ (see Fig. 4(a)).

Consider a staircase polygon, and split it by drawing a horizontal line through every concave vertex on its lower stair. Each of these subpolygons is an L_polygon, defined by an L_path on the lower stair of the staircase polygon. Let us consider such a subpolygon defined by $L\_path(p_i, p_j)$, $(p_i, p_j) \in E$ (see Fig. 4(b)). Its upper stair is a staircase path which spans between two points, i_k and j_ℓ, where i_k is the projection of a point $p_k \in P$ (where $(p_k, p_i) \in E$) on the line $Y = y_i$, and j_ℓ is the projection of a point $p_\ell \in P$ (where $(p_\ell, p_j) \in E$) on the line $Y = y_j$. This subpolygon contributes $Area(L\_polygon[i_k, p_j])$ in the entire staircase polygon.

It may be noted that if the lower stair of a staircase polygon passes through $L\_path(p_i, p_j)$, then its area bounded by the horizontal lines $Y = y_i$ and $Y = y_j$ may not be unique. To illustrate this, let us consider two staircase polygons whose lower stairs pass through $L\_path(p_i, p_j)$. Let R_1 and R_2 be the portions of the lower stairs of these two staircase polygons from b to p_i (see Fig. 4(c)). R_1 reaches p_i using $L\_path(p_\theta, p_i)$ and R_2 reaches p_i with $L\_path(p_\phi, p_i)$, where $p_\theta, p_\phi \in P$. The upper stairs of the two MESPs meet the line $Y = y_i$ at i_ϕ and i_ψ, respectively. The subpolygons of these two staircase polygons between the horizontal lines $Y = y_i$ and $Y = y_j$, are $L\_polygon[i_\phi, p_j]$ and $L\_polygon[i_\psi, p_j]$, respectively, as shown in Fig. 4(c). Thus, a unique weight *cannot* be assigned to the edge (p_i, p_j) of the digraph G, and hence, the problem of finding the largest MESP cannot be formulated as finding the max-weight path in an weighted version of G. This motivates us to introduce a new weighted digraph, called *staircase graph*, whose nodes correspond to the *footprints* of the points in P as defined below.

Definition 7. Let p_i and p_k be two points in P such that $x_k < x_i$ and $y_k < y_i$, i.e., $(p_k, p_i) \in E$. Then the point (x_k, y_i) (where the vertical line through the point p_k meets the line $Y = y_i$) is called the *footprint* of p_i contributed by p_k, and is denoted by i_k (see Fig. 5(a)). The footprint of b (the bottom-left corner point of the floor) is b itself.

The set of footprints of point p_i will be denoted as $FP(p_i)$. The number of footprints of a point $p_i \in P \backslash \{b\}$ is the indegree of node p_i in the digraph G, or in other words, $|FP(p_i)| = in(p_i)$. In Fig. 5(b), the set of footprints of p_i ($\in P \setminus \{b\}$) contributed by the points in P, is shown.

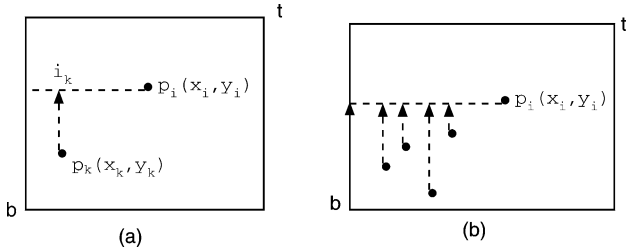


Fig. 5. Footprints.

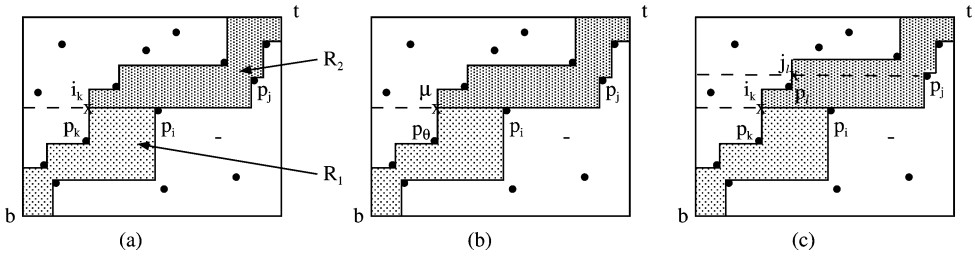


Fig. 6. Proof of Lemma 3.

Lemma 3. *Consider an L_path*(p_i, p_j) *such that* $p_i, p_j \in P$. *Then,*

(a) *for every footprint* i_k *of* p_i, *there exists at least one MESP whose lower stair passes through L_path*(p_i, p_j), *and the upper stair passes through* i_k;
(b) *for every MESP whose lower stair passes through the L_path*(p_i, p_j), *the point where the upper stair intersects the line* $Y = y_i$, *must be a footprint of* p_i;
(c) *for every MESP whose lower stair passes through L_path*(p_i, p_j), *if its upper stair passes through the footprint* i_k (*of* p_i, *contributed by* p_k), *then the upper stair also passes through a unique footprint, say* j_ℓ *of* p_j, *for some* $p_\ell \in P$, *where* $x_\ell \geqslant x_k$ & $y_\ell > y_k$.

Proof.

(a) Let i_k be a footprint of p_i contributed by p_k. From Definition 7, p_k lies on the vertical line through i_k, and $x_k < x_i$ & $y_k < y_i$. Now, consider a subfloor with bottom-left and top-right corners at b and p_i respectively, and consider a complete path from b to p_i in G whose last edge is (p_k, p_i). By Lemma 1, this complete path corresponds to the upper stair of a unique maximal-empty staircase polygon, say R_1, whose bottom-left and top-right corners coincide with b and p_i, respectively (see Fig. 6(a)). Now consider another maximal-empty staircase polygon (R_2) from i_k to t whose lower stair passes through the $L\_path(p_i, p_j)$. Concatenation of R_1 and R_2 is a MESP from b to t, which proves this part of the lemma.

(b) Consider a MESP whose lower stair passes through the $L\_path(p_i, p_j)$, and the upper stair intersects the line $Y = y_i$ at μ (see Fig. 6(b)). The upper stair cannot have a concave corner at μ because, in that case $\mu \in P$, and will have the same y-coordinate as that of p_i, which contradicts our assumption. Let $p_\theta = (x_\theta, y_\theta)$ be the last concave corner on the upper stair of the MESP before reaching μ. Clearly,

$p_\theta \in P$ and the vertical line through p_θ passes through μ. Also $x_\theta < x_i$ and $y_\theta < y_i$. Hence, μ must be the footprint of p_i contributed by p_θ.

(c) Consider a MESP between i_k and p_j, whose lower stair is the $L\_path(p_i, p_j)$. From Theorem 1, the point where the upper stair meets the line $Y = y_j$, is unique. By part (b) of this lemma, this intersection point must be a footprint of p_j. The remaining part of the lemma follows from the monotonicity of the upper R-stair. □

Consider an edge (p_i, p_j) in the digraph G. Lemma 3 implies that all the MESPs whose lower stairs pass through the $L\_path(p_i, p_j)$, and the upper stairs pass through the footprint i_k of p_i, have the $L\_polygon[i_k, p_j]$ in common (see Fig. 6(c)). Furthermore, the upper stair intersects the line $Y = y_j$ at a unique footprint of p_j, say j_ℓ contributed by some point $p_\ell \in P$. Thus, the $L\_polygon[i_k, p_j]$ is uniquely characterized by the pair of footprints (i_k, j_ℓ). A MESP can be viewed as a concatenation of a set of such disjoint L_polygons. We now introduce a new directed graph, called the *staircase graph* $SG = (V', E')$, which is an enhanced and weighted version of the digraph G, and show that the location of largest MESP is equivalent to finding the max-weight path between two designated nodes in SG.

3.3.2. The staircase graph

Definition 8. The *staircase graph* $SG = (V', E')$ for a given digraph $G = (V, E)$ is a weighted digraph with nodes $V' = \bigcup_{p_i \in P} FP(p_i) = \{$the set of footprints of all the points in $P\}$. A footprint $i_k \in FP(p_i)$ has a directed edge to a footprint $j_\ell \in FP(p_j)$, if $(p_i, p_j) \in E$ (i.e., (p_i, p_j) is an L_path), and the upper stair of the $L\_polygon[i_k, p_j]$ meets the line $Y = y_j$ at the footprint j_ℓ. The weight of the edge $(i_k, j_\ell) \in E'$, denoted as $w(i_k, j_\ell)$, is equal to the area of the $L\_polygon[i_k, p_j]$.

In the staircase graph SG, defined above, the node b (footprint of b by Definition 7) will have indegree 0, and the nodes $t_b, t_1, \ldots, t_n \in FP(t)$ will have outdegree 0. We now augment SG with a dummy *sink* node and put directed edges of weight 0 from each node in $FP(t)$ to the sink node.

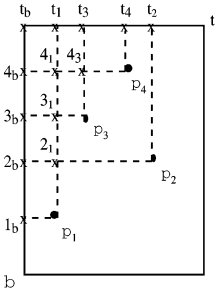
Clearly, the staircase graph corresponding to a given digraph G is unique. For the example of Fig. 3(a), the set of footprints and the corresponding staircase graph are shown in Fig. 7(a) and Fig. 7(c). The illustration of edge weights in SG is shown in Fig. 7(b). The staircase graph $SG = (V', E')$ can be constructed directly from the digraph $G = (V, E)$ as stated in the following lemma.

Lemma 4.

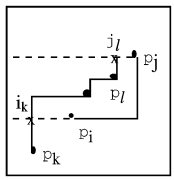
(a) *There will be a directed edge from the node i_k to the node j_ℓ $(k \neq \ell)$ in the staircase graph SG if and only if $(p_k, p_i) \in E$, $(p_i, p_j) \in E$ and $p_l = (x_l, y_l)$ is a point in P such that $x_l = \max_r\{x_r \mid p_r \in X_{ijk}\}$, where $X_{ijk} = \{p_s \mid s > i,\ (p_k, p_s) \in E$ and $(p_s, p_j) \in E_T\} \neq \phi$;*

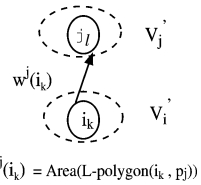
(b) *There exists a directed edge from i_k to j_k in SG if and only if $(p_k, p_i) \in E$, $(p_i, p_j) \in E$ and the set X_{ijk}, defined in part* (a) *of this lemma, is empty.*

Proof. The proof immediately follows from the geometrical significance of the lemma. Let R denote the rectangle whose bottom-left and top-right corners lie at $i_k = (x_k, y_i)$ and $p_j = (x_j, y_j)$, respectively. From the definition of staircase graph $SG = (V', E')$, it follows that a directed edge (i_k, j_ℓ), $k \neq \ell$, will be present in E', if and only if the point $p_\ell (\in P)$ lies within the rectangle R, and x_ℓ is maximum among

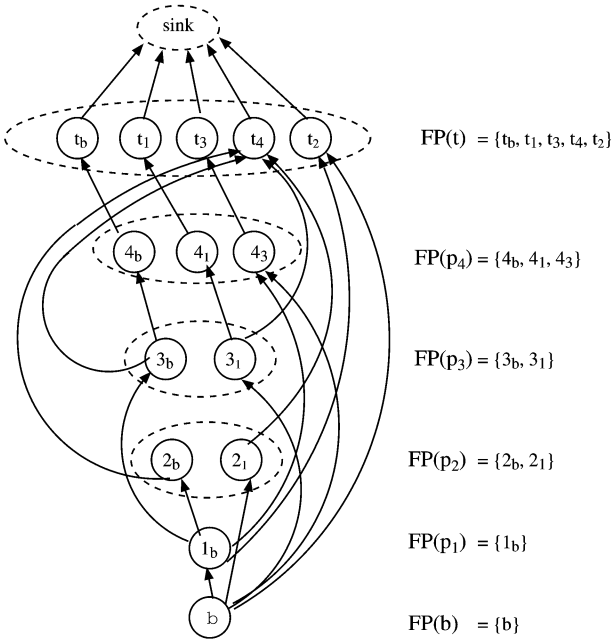


(a) Footprints for the example of Figure 2a





(b) directed edges in the staircase graph



(c) The corresponding staircase graph (edge-weights are not shown)

Fig. 7. The staircase graph.

all points lying inside R (see Fig. 8(a)). On the other hand, $(i_k, j_k) \in E'$ if and only if R contains no point of P (see Fig. 8(b)). □

The graph SG is acyclic. If $(p_i, p_j) \in E$, Then there exists at least one edge from the set of nodes $FP(p_i)$ to the set of nodes $FP(p_j)$ in the graph SG. The edges from the nodes in $FP(p_i)$ to those in $FP(p_j)$ satisfy the following properties:

Corollary 4.1. *The edge* (i_k, j_ℓ) *of* $SG(V', E')$ *corresponds to the* $L\_polygon[i_k, p_j]$.

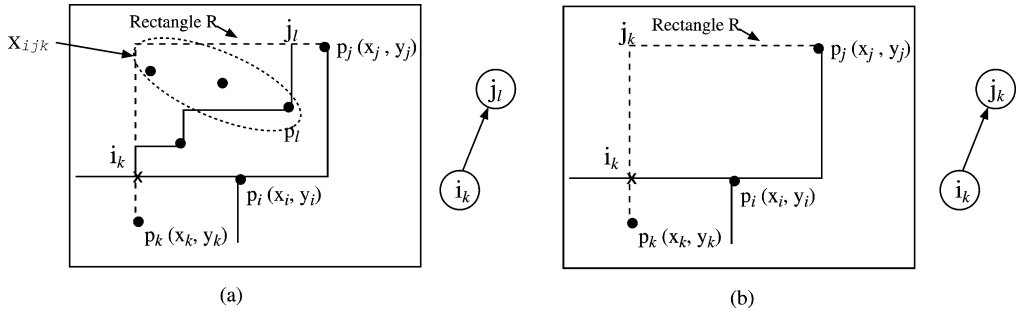


Fig. 8. Geometric significance of Lemma 4.

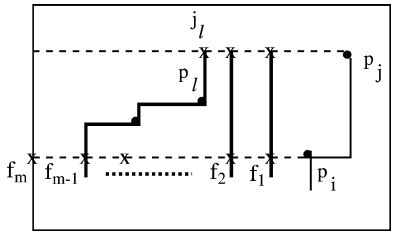


Fig. 9. L_polygons for various footprints.

Corollary 4.2. *If $(i_k, j_\ell) \in E'$ and i_k to the left of j_ℓ, then for all $i_\alpha \in FP(p_i)$ which lie to the left of i_k on the line $Y = y_i$, $(i_\alpha, j_\ell) \in E'$.*

Corollary 4.3. *In the digraph SG, the outdegree of every footprint $i_\alpha \in FP(p_i)$ is equal to out(p_i), and every footprint $i_\alpha \in FP(p_i)$ has exactly one successor (say j_β) in the set $FP(p_j)$, whenever $(p_i, p_j) \in E$.*

Since all the footprints of a point $p_i \in P$ appear on the horizontal line $Y = y_i$, it is convenient to label them in a linearly-ordered fashion in the decreasing order of their x coordinates.

Let $FP(p_i) = \{f_1, f_2, \ldots, f_m\}$ and $FP(p_j) = \{g_1, g_2, \ldots, g_{m^*}\}$ be the set of footprints of p_i and p_j, respectively, ordered from right to left; $m = in(p_i)$, $f_m = i_b$, $m^* = in(p_j)$, and $g_{m^*} = j_b$. Corollary 4.3 indicates that every footprint $f_\alpha \in FP(p_i)$ has exactly one successor in $FP(p_j)$. The weights of the edges from the set $FP(p_i)$ to the set $FP(p_j)$ satisfy the following inequality.

Corollary 4.4. *If $FP(p_i) = \{f_1, f_2, \ldots, f_m\}$, then $w(f_1, g_{1^*}) < w(f_2, g_{2^*}) < \cdots < w(f_m, g_{m^*})$, where the successor of f_α in the set $FP(p_j)$ is denoted as g_{α^*}.*

Proof. We need to show that

$$Area\big(L\_polygon[f_1, p_j]\big) < Area\big(L\_polygon[f_2, p_j]\big) < \cdots < Area\big(L\_polygon[f_m, p_j]\big).$$

From Fig. 9, it trivially follows that $L\_polygon[f_1, p_j] \subset L\_polygon[f_2, p_j] \subset \cdots \subset L\_polygon[f_m, p_j]$, where "$\subset$" denotes the relation "contained in". Hence the result follows. □

The size of the staircase graph $SG = (V', E')$ can be expressed in terms of the number of vertices and edges of the graph $G = (V, E)$ as stated in the next theorem.

Theorem 2. $|V'| = |E| + 1$, *and* $|E'| = O(n|E|)$.

Proof. The first part of the theorem is clear. By Corollary 4.3, the total number of outgoing edges of i_k in the graph *SG* is equal to $out(p_i)$. Again, the total number of footprints of a point p_i is equal to $in(p_i)$. Thus, the number of edges $|E'| = \sum_{i=1}^{n} in(p_i) \times out(p_i)$ which, in the worst case, is $O(n|E|)$. □

Theorem 3. *For every complete path in G, there exists a unique directed path from b to the sink node in SG and vice-versa.*

Proof. Follows from the construction of the digraph *SG*. □

Consider the path $b \to p_1 \to p_3 \to p_4 \to t$, in G (Fig. 3(b)). The corresponding path in *SG* is $b \to 1_b \to 3_b \to 4_b \to t_b \to$ *sink* (see Fig. 7(c)). Similarly, for the directed path $b \to 1_b \to 4_3 \to t_3 \to$ *sink* of *SG*, the corresponding path in G is $b \to p_1 \to p_4 \to t$.

Theorem 4.

(a) *Every directed path in SG from b to the sink node represents a MESP uniquely, and vice-versa.*
(b) *The sum of edge-weights along a directed path in SG is equal to the area of the corresponding MESP.*

Proof. By Theorem 3, each directed path in *SG* corresponds to a unique complete path in G which, in turn, corresponds to a unique MESP (by Lemma 1). Each edge in *SG* corresponds to an L_polygon (Corollary 4.1). A MESP corresponding to a complete path in *SG* is the concatenation of the L_polygons corresponding to the edges on that directed path, and its area is the sum of weights of the edges on that path. □

For the sake of notational simplicity, we shall define the *weight of a directed path* in *SG* by the sum of edge-weights along that path. The *max-weight path* is a path from *b* to the *sink* node of *SG* whose weight is maximum among all such paths in *SG*.

Theorem 4 leads to the fact that the largest MESP from the bottom-left corner of the floor to its top-right corner can be found by determining the max-weight path in the digraph *SG*. Note that, the max-weight path describes both the upper and the lower stair of the largest MESP. If a node $i_k \in V'$ is on a path, the lower stair of the corresponding MESP passes through p_i, and the upper stair passes through p_k. Thus, we have the following theorem.

Theorem 5. *The largest MESP can be recognized in* $O(n|E|)$ *time using* $O(|E'|)$ *space.*

Proof. Follows from the fact that the construction time of the graph *SG* is $O(n|E|)$. The time for locating the max-weight path in an acyclic digraph is $O(|E'|) = O(n|E|)$, which may be $O(n^3)$ in the worst case. The space complexity follows from the number of nodes in *SG* (see Theorem 2). □

4. Improved algorithm based on geometric properties

The method of finding the largest MESP can be accelerated significantly based on certain geometric properties of the ensemble of points on the floor. The staircase graph $SG = (V', E')$ need not be constructed explicitly. The max-weight path in SG can be recognized by sweeping a horizontal line on the floor from bottom to top in the first pass. The execution starts from the bottom-left corner b. Each point p_i is picked up sequentially from the set P in increasing order of their y-coordinates; it is *completely processed* (i.e., all its outgoing edges in the graph G are processed), and then the next point is selected for processing. The process continues until the top-right corner t is reached, and finally, the area of the largest MESP is reported. In the second pass, the lower and upper stairs of the largest MESP are reported.

The core of the revised algorithm is the *complete processing* of a point, say p_i, which involves two phases. In Phase-1, a preprocessing is done. In Phase-2, all the outgoing edges of p_i in graph G are processed.

In order to formalize the Phase-1 (the preprocessing phase), we define a directed graph $G^c = (V^c, E^c)$; $V^c = P \cup \{b', t'\}$, $b' = (x_t, 0)$ and $t' = (0, y_t)$ are respectively the bottom-right and top-left corners of the floor, and $E^c = \{(p_i, p_j) \mid p_i, p_j \in V^c,\ x_i > x_j \text{ and } y_i < y_j\}$. Clearly, G^c is also a permutation graph. We shall denote the transitive reduced graph of G^c by $G^c_T = (V^c, E^c_T)$.

We split the set of points *above* p_i into two subsets by a vertical line through p_i (see Fig. 10). In Phase-1, the points in the left subset are processed by sweeping a horizontal line from $Y = y_i$ upwards, and a data structure is created with the set of points having directed edges from p_i in the graph G^c_T. In Phase-2, the points in the right subset are processed by sweeping a vertical line from $X = x_i$ towards the right, and all the outgoing edges from p_i in the graph G are inspected in order to choose an appropriate point p_j such that $L\_path(p_i, p_j)$ appears on the lower stair of the MESPs passing through p_i. We exploit three important geometric properties, namely (i) *dominance*, (ii) *footprint propagation*, and (iii) *inheritance* to accelerate processing in Phase-2. At the beginning, for a point $p_i \in P$, the footprints of p_i (i.e., the set $FP(p_i)$) is available. Each member $f_\alpha \in FP(p_i)$ is assigned with the area of the largest MESP from b to p_i and whose upper stair reaches the horizontal line $Y = y_i$ at f_α. The dominance property helps to reduce the size of $FP(p_i)$ by eliminating footprints that can not produce the largest MESP. Next, for each edge $(p_i, p_j) \in E$, we apply the method of footprint propagation to generate new footprints to the point p_j, and the MESPs reaching at p_i are extended up to p_j. A suitable data structure is created in Phase-1 to expedite this process. The area is progressively computed using the inheritance property.

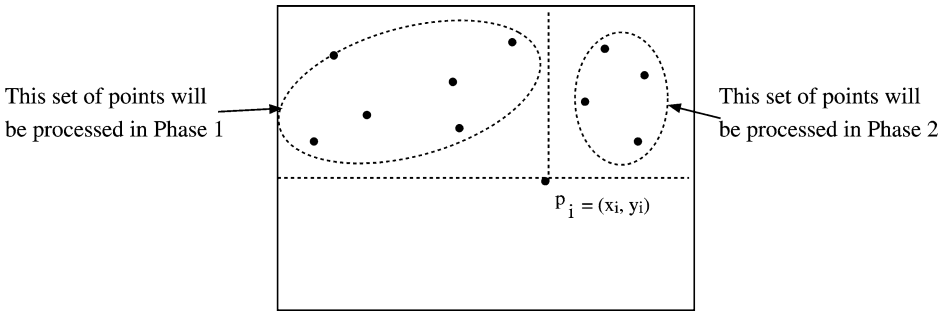


Fig. 10. Complete processing of a point p_i.

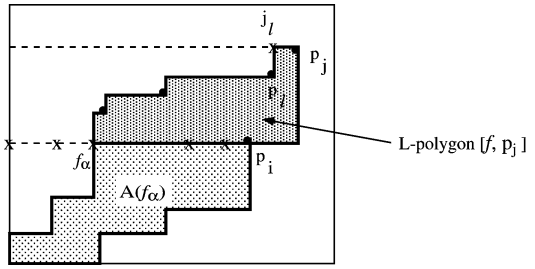


Fig. 11. Demonstration of LMESP and its area computation.

4.1. Geometric properties

Consider the point $p_i \in P$ currently under processing, and let $FP(p_i)$ be the set of footprints generated prior to processing of p_i. By Lemma 3, for every MESP whose lower stair passes through p_i, the upper stair must pass through some footprint $f_\alpha \in FP(p_i)$. Conversely, for every footprint on p_i, there exists at least one such MESP. If $f_\alpha \in FP(p_i)$, $A(f_\alpha)$ denotes the area of the largest MESP from b to p_i, whose upper stair hits the line $Y = y_i$, at footprint f_α. Thus, for a node $f_\alpha \in V'$, $A(f_\alpha)$ denotes the sum of weights of the maximum weighted path from b to f_α in $SG = (V', E')$.

Consider now an edge $(p_i, p_j) \in E$ which corresponds to an L_path in the lower stair. For each footprint $f_\alpha \in FP(p_i)$, we use LMESP$[p_i, p_j, f_\alpha]$ to denote the largest MESP from b to p_j, whose lower stair terminates in the $L\_path(p_i, p_j)$, and the upper stair goes through f_α (see Fig. 11). Then,

$$Area\big(\text{LMESP}[p_i, p_j, f_\alpha]\big) = A(f_\alpha) + Area\big(L\_polygon[f_\alpha, p_j]\big).$$

In the following subsection, we show that all the members in $FP(p_i)$ need not to be considered while processing p_i. Next, we consider the processing of outgoing edges $(p_i, p_j) \in E$. In Sections 4.1.2 and 4.1.3, we describe the actions if $(p_i, p_j) \in E_T$ and $(p_i, p_j) \notin E_T$, respectively.

4.1.1. Dominance

Lemma 5. *For a pair of footprints f_α and f_β on p_i, if $\alpha > \beta$ and $A(f_\alpha) > A(f_\beta)$, then Area*(LMESP$[p_i, p_j, f_\alpha]$) > *Area*(LMESP$[p_i, p_j, f_\beta]$).

Proof. Follows from Corollary 4.4. □

Thus, in order to determine the largest MESP, the footprints f_β $(\beta < \alpha)$ need not be considered further if $A(f_\beta) \leqslant A(f_\alpha)$. This shows that only a subset of footprints on p_i, say $\{f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_p}\}$ (ordered from right to left) is to be retained provided $A(f_{\alpha_1}) > A(f_{\alpha_2}) > \cdots > A(f_{\alpha_p})$. This reduced set of footprints will be referred to as the *prime set of footprints*.

4.1.2. Footprint propagation

Consider an edge $(p_i, p_j) \in E_T$. The set of footprints $FP(p_i) = \{f_1, f_2, \ldots, f_m\}$ are ordered from right to left. The following lemma describes an efficient mechanism of *propagating* $FP(p_i)$ in the footprint lists of the points p_j by projecting them on the line $Y = y_j$.

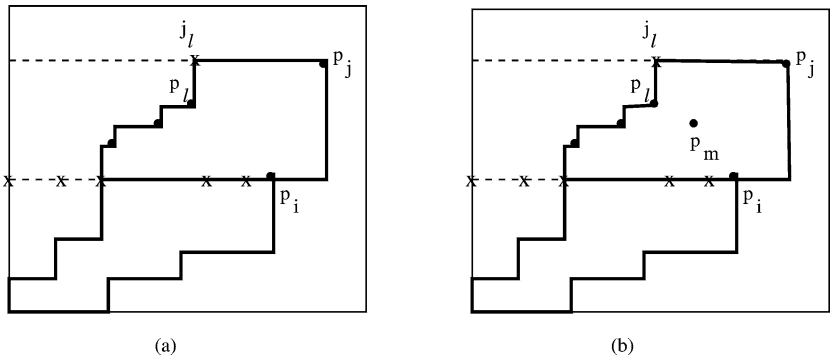


Fig. 12. Illustration of Lemma 6: (a) $(p_i, p_\ell) \in E_T^c$ and (b) $(p_i, p_\ell) \notin E_T^c$.

Lemma 6. *Consider a point $p_\ell \in P$ such that $(p_i, p_\ell) \in E^c$, i.e., $x_\ell < x_i$ and $y_\ell > y_i$. Now,*

(a) *if $(p_i, p_\ell) \in E_T^c$, then there exists at least one MESP whose lower stair passes through p_i, and upper stair passes through p_ℓ;*
(b) *if $(p_i, p_\ell) \notin E_T^c$, then there exists no such MESP.*

Proof. Consider the rectangle with p_i and p_ℓ at its bottom-right and top-left corners, respectively. If $(p_i, p_\ell) \in E_T^c$, then the aforesaid rectangle is empty (see Fig. 12(a)). In such a case, a MESP, as stated in the lemma, exists. But if $(p_i, p_\ell) \notin E_T^c$, then the rectangle contains another point (say p_m, as in Fig. 12(b)) in its interior, and the existence of a MESP is impossible. □

Let P_i^c denote the set of points having directed edges from p_i in G_T^c. Let the members of P_i^c be arranged in increasing order of their y-coordinates. The following facts are important.

Fact 1. *Let $(p_i, p_j) \in E_T$; $FP(p_i)$ and $FP(p_j)$ be the sets of existing footprints prior to processing of (p_i, p_j). Then during processing of (p_i, p_j),*

(a) *if there does not exist any point $p_s \in P_i^c$ such that $y_s < y_j$, then by Lemma* 4(b)*, all the members of $FP(p_i)$ will be projected on the horizontal line $Y = y_j$ to create new footprints in $FP(p_j)$ (see Fig.* 13(a)*). In the remaining discussion, this event will be referred to as $FP(p_i)$ being propagated to $FP(p_j)$;*
(b) *otherwise, let $p_\ell \in P_i^c$ be the point such that $y_\ell < y_j$, and having the maximum x-coordinate value. There may exist a subset of $FP(p_i)$, say $\{f_\alpha, \alpha = 1, 2, \ldots, r\}$, whose x-coordinates are greater than x_ℓ. For each of them, the rectangle$[f_\alpha, p_j]$ is empty (see Fig.* 13(b)*). So, all of them are projected on the line $Y = y_j$ to create footprints in $FP(p_j)$. In addition, one new footprint j_ℓ is created in $FP(p_j)$, which is obtained by projecting p_ℓ on the line $Y = y_j$ (by Lemma* 6(a)*).*

Fact 2. *While processing an edge $(p_i, p_j) \in E_T$, if p_j is found above $p_\ell \in P_i^c$ then p_j will not inherit any footprint from $FP(p_i)$ whose x-coordinates are less than x_ℓ (see Fig.* 13(b)*).*

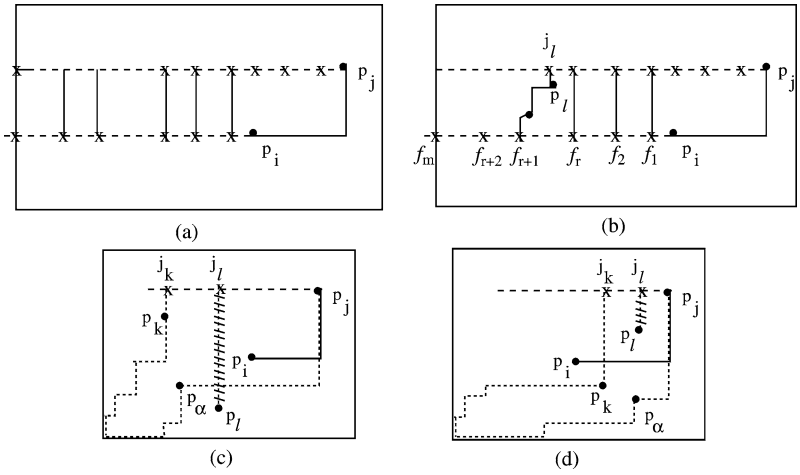


Fig. 13. Illustration of footprint propagation.

We scan the footprint list $FP(p_i)$ from right to left starting from f_1 and propagate them to $FP(p_j)$. The termination of this procedure is determined using the case stated in Fact 1(b).

Fact 3. *All the newly generated footprints of p_j have x-coordinates less than that of the left-most of the existing footprints of $FP(p_j)$. Thus, they can be added to the left of the existing footprints of $FP(p_j)$, in the order they have been generated.*

Proof. (By contradiction) Let $FP(p_j)$ be the list of footprints that are generated prior to the processing of the edge $(p_i, p_j) \in E_T$. We show that, while processing the edge (p_i, p_j), if a new footprint j_ℓ is generated, it cannot lie to the right of an existing footprint $j_k \in FP(p_j)$.

Suppose such a situation happened. Here, j_k has been generated while processing another edge $(p_\alpha, p_j) \in E$, such that $y_\alpha < y_i$ (since the points are being processed by sweeping a horizontal line from bottom to top). Now, the following two situations may arise:

$(p_\alpha, p_i) \in E$: Here, the staircase polygon whose lower stair ends with $L\_path(p_\alpha, p_j)$, and upper stair passes through p_k is not empty; it contains p_i in its interior (see Fig. 13(c)).

$(p_\alpha, p_i) \notin E$: In this case, the staircase polygon whose lower stair ends with $L\_path(p_\alpha, p_j)$, and upper stair passes through p_k is not empty; it contains either p_ℓ or both p_i and p_ℓ in its interior (see Fig. 13(d)).

Thus, j_ℓ cannot be a footprint of p_j. In Figs. 13(c) and (d), this event is depicted by scratched lines. □

Let $FP(p_j) = \{q_1, q_2, \ldots, q_s\}$ be the set of footprints of p_j prior to the processing of the edge (p_i, p_j). Then the new footprints of $FP(p_j)$, created by the projections of $\{f_1, f_2, \ldots, f_r\} \subseteq FP(p_i)$ are numbered as $\{q_{s+\alpha}, \alpha = 1, \ldots, r\}$ (by Fact 3). In the graph SG, $(f_\alpha, g_{s+\alpha})$ is an edge for all $\alpha = 1, \ldots, r$ (by Fact 1). The weights of these edges are computed as follows:

(i) $w(f_\alpha, g_{s+\alpha}) = Area(rectangle[f_\alpha, p_j])$, for all $\alpha = 1, \ldots, r$.

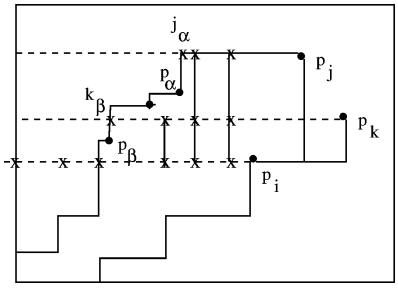


Fig. 14. Illustration of Lemma 7.

(ii) $w(f_{r+1}, j_\ell) = Area(L\_polygon[f_{r+1}, p_j])$.
(iii) For each of the other footprints f_α, $\alpha > r+1$ (if at all exists), the upper stair of the $L\_polygon[f_\alpha, p_j]$ meets the line $Y = y_j$ at j_ℓ (Lemma 6(a)). So, (f_α, j_ℓ) is an edge in *SG*, and its weight $w(f_\alpha, j_\ell) = Area(L\_polygon[f_\alpha, p_j])$ for all $\alpha > r+1$ (see Fig. 13(b)).

Definition 9. For an edge $(p_i, p_j) \in E_T$, a point p_ℓ is said to be a *footprint blocking point* for p_j with respect to p_i if it is the rightmost point in P_i^c satisfying $y_\ell < y_j$ (see Fact 2).

Lemma 7. *Let us consider two MESPs whose lower stairs contain* $L\_path(p_i, p_j)$ *and* $L\_path(p_i, p_k)$, *respectively*; *the edges* (p_i, p_j) *and* (p_i, p_k) *are both in* E_T, *and* $x_j < x_k$. *If* p_α *and* p_β *are respectively the footprint blocking points for the points* p_j *and* p_k *with respect to* p_i, *then* $y_\alpha \geqslant y_\beta$.

Proof. Follows from Fact 2 and the fact that $y_j > y_k$ (see Fig. 14). □

Lemma 7 implies that if the successors of p_i in G are processed from left to right, then in order to get the footprint blocking points for all the points $\{p_j \mid (p_i, p_j) \in E_T\}$, the list P_i^c is to be traversed only once.

4.1.3. Inheritance

Let $(p_i, p_j) \in E$. By Corollary 4.3, every node $f_\alpha \in FP(p_i)$ will have exactly one successor node g_β in $FP(p_j)$ in the graph *SG*; the weight associated with this edge is $w(f_\alpha, g_\beta) = Area(L\_polygon[f_\alpha, p_j])$. In the earlier subsection, a method of calculating the edge weights among the nodes in $FP(p_i)$ and $FP(p_j)$ in *SG* has been discussed for $(p_i, p_j) \in E_T$. Here, we consider the case where $(p_i, p_j) \notin E_T$, and show that if the successors of p_i is processed from left to right, then the computation of edge-weight for one successor of p_i often aids determining the same for other successors of p_i. This process, explained below, will be referred to as *inheritance* in subsequent discussions.

Lemma 8. *If* $(p_i, p_j) \notin E_T$, *then all the footprints in* $FP(p_i)$ *have a common successor in* $FP(p_j)$.

Proof. Since the edge $(p_i, p_j) \notin E_T$, there must exist a point p_k such that $(p_i, p_k) \in E$ and $(p_k, p_j) \in E_T$ (see Fig. 15). Draw a vertical line through p_k which meets the lines $Y = y_i$ and $Y = y_j$ at p^* and j_k, respectively. By Definition 7, j_k is a footprint in $FP(p_j)$, and the *rectangle*$[p^*, p_j]$ is empty. By Lemma 4(a), each of the footprints $f_\alpha \in FP(p_i)$ will have a directed edge to the footprint $j_k \in FP(p_j)$. Furthermore, the weights of these edges can be determined as follows:

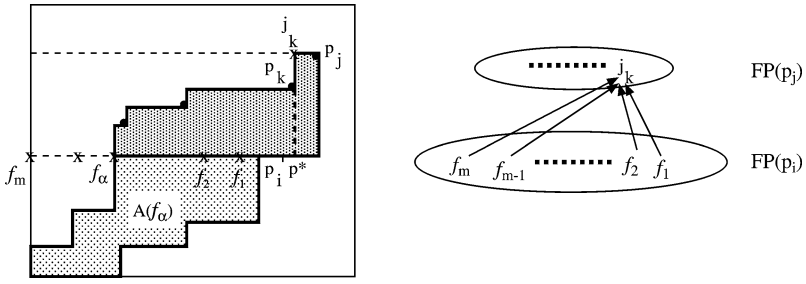


Fig. 15. Illustration of inheritance.

$$w(f_\alpha, j_k) = \mathit{Area}\big(L\_polygon[f_\alpha, p_j]\big)$$
$$= \mathit{Area}\big(L\_polygon[f_\alpha, p_k]\big) + \mathit{Area}\big(\mathit{rectangle}[p^*, p_j]\big). \qquad \square$$

Thus, if the edge $(p_i, p_k) \in E$ is already processed, then $\mathit{Area}(L\_polygon[f_\alpha, p_k])$ is already available, and we can use this incremental scheme of computing the edge weights between the nodes in $FP(p_i)$ and $FP(p_j)$. Such an effect of inheritance can be accomplished provided for every point $p_i \in P$, its successors in the graph G are processed in the increasing order of their x-coordinates. Furthermore,

$$\mathrm{LMESP}[p_i, p_j, f_\alpha] = \mathrm{LMESP}[p_i, p_k, f_\alpha] + \mathit{Area}\big(\mathit{rectangle}[p^*, p_j]\big).$$

4.2. Data structure

We maintain a linear array, called $\mathcal{P}$, which contains the set of points $P \cup \{b, t\}$ in the increasing order of their y-coordinates. With each member $p_i \in \mathcal{P}$, apart from its coordinates, a pointer to the list of footprints $FP(p_i)$ is attached. Initially, $FP(b) = b$, and $FP(p_i) = \phi$, for all $p_i \in P \setminus \{b\}$. These lists grow dynamically during the execution of the algorithm. As all the incoming edges of a point p_i in the graph G are processed prior to the processing of its outgoing edges, all the footprints in $FP(p_i)$ are available at the beginning of processing p_i. As mentioned in Section 4.1.1, we need to retain only the prime set of footprints of p_i; henceforth, $FP(p_i)$ will be used to denote the prime set of footprints of p_i. The fields associated with each member $f_\alpha \in FP(p_i)$ are as follows:

(a) the x coordinate of f_α;
(b) $A(f_\alpha)$, the area of the largest MESP from b to the point p_i whose upper stair meets the footprint f_α;
(c) a pointer *pred*, indicating the immediate predecessor of f_α in the graph *SG*, which contributes $A(f_\alpha)$.

Field (c) is used for retracing the path in the backward pass for identifying the largest MESP.

In order to process the successors of a point p_i in G in the left-to-right order, we maintain another array $\mathcal{P}^*$ with the members of P in increasing order of their x coordinates. Attached to each member of $\mathcal{P}$, there is a pointer, called *self-indicator*, that points to its own presence in $\mathcal{P}^*$.

In addition, two stacks, namely S_1 and S_2 are created, during the processing of a point.

4.3. Complete processing of a point

As mentioned earlier, the *complete processing* of a point $p_i \in \mathcal{P}$, involves two phases, details of which are described below. The main steps of the entire algorithm are enumerated in Appendix A.

4.3.1. Phase-1

In this phase, we create a stack S_1 that contains the set of points $P_i^c = \{p_\ell \mid (p_i, p_\ell) \in E_T^c\}$ in decreasing order of their y-coordinates. Let us consider an edge $(p_i, p_j) \in E$ (which corresponds to an L_path in the lower stair), and the *maximum weighted path* in SG which passes through $FP(p_i)$ and $FP(p_j)$. If the upper stair passes through $p_\ell \in P_i^c$ $(y_\ell < y_j)$, then by Fact 1(b), an area

$$\mathcal{B}(p_\ell) = \max_{k>\alpha}\big(A(f_k) + \mathit{Area}\big(\mathit{L\_polygon}[f_k, p_\ell]\big)\big)$$

must be contributed to the above mentioned maximum weighted path, where f_α is the rightmost element in $FP(p_i)$ whose x-coordinate is less than that of p_ℓ (see Fig. 13(b)).

Thus, with each point $p_\ell \in P_i^c$, we attach two fields (i) $\mathcal{B}(p_\ell)$, and (ii) a pointer to f_α.

We compute $\mathcal{B}(p_\ell)$ for all points $p_\ell \in P_i^c$ by sweeping a horizontal line $\mathcal{L}$ from $Y = y_i$ upwards. An *active span* of $\mathcal{L}$ is maintained during the sweep, which is a closed interval of the form $[a, x_i]$. The right boundary of the active span is fixed at x_i, whereas its left boundary changes during the sweep. Initially, a is set to 0. If a new point p_ℓ, encountered by $\mathcal{L}$, lies outside the *active span*, it is ignored (since either (i) $x_\ell > x_i$—here $p_\ell \notin P_i^c$, or (ii) $x_\ell < a$—in this case $(p_i, p_\ell) \notin E_T^c$). Otherwise, (i.e., if $(p_i, p_\ell) \in E_T^c$), we compute $\mathcal{B}(p_\ell)$, and push the triplet $(p_\ell, \mathcal{B}(p_\ell), f_\alpha)$ in stack S_1. The *active span* is then reduced to the interval $[x_\ell, x_i]$, and the sweep of $\mathcal{L}$ is advanced. Note that, if S_1 is non-empty before adding p_ℓ, with p_k being the top-most element in S_1, then the upper stair reaches p_ℓ either from p_k, or directly from any of the footprints of p_i lying in the interval $[x_k, x_\ell]$ (see Fig. 16). Thus,

$$\mathcal{B}(p_\ell) = \max\Big\{\big(\mathcal{B}(p_k) + \mathit{Area}\big(\mathit{rectangle}[(x_k, y_i), p_\ell]\big)\big),\ \max_{f_\beta \in [x_k, x_\ell]}\big(A(f_\beta) + \mathit{Area}\big(\mathit{rectangle}[f_\beta, p_\ell]\big)\big)\Big\}.$$

This phase terminates when the sweep line reaches the top boundary of the bounding box. At this moment, S_1 will contain all the points in P_i^c.

Lemma 9. *Completion of Phase-1 for a point p_i requires $O(n)$ time in the worst case.*

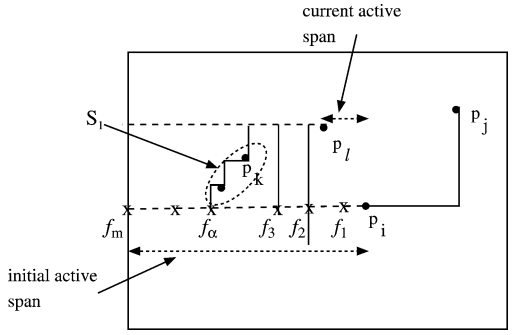


Fig. 16. Computation of $\mathcal{B}(p_\ell)$.

Proof. Follows from the following two facts: (i) the number of points in P_i^c is at most $O(n)$, and (ii) the maintenance of active span during the sweep ensures that while computing $\mathcal{B}(p_\ell)$ for all points $p_\ell \in P_i^c$, the list of footprints $FP(p_i)$ is effectively traversed only once from left to right. □

4.3.2. Phase-2

In this phase, we process the successors of p_i in the graph G by sweeping a vertical line from left to right starting from p_i, establish necessary edges in SG, and finally compute the largest MESP with $L\_path(p_i, p_j)$ in the lower stair for all the edges $(p_i, p_j) \in E$. Let (p_i, p_j) be an edge that is currently under Phase 2 of *complete processing* for the point p_i. Now, the following observation is important.

Observation 2. *The concave vertices of the upper stair in the L_polygon*$[p_i, p_j]$ *correspond to the points* (*nodes*) *on a path from* p_i *to* p_j *in* G_T.

In order to exploit the effect of inheritance as described in Section 4.1.3, we use Observation 2, and maintain a stack S_2 while processing p_i as follows:

> Let the edge $(p_i, p_k) \in E$ be processed just before (p_i, p_j), and $\{p_i(= p_{\alpha_0}), p_{\alpha_1}, p_{\alpha_2}, \ldots, p_k (= p_{\alpha_m})\}$ be the set of points on a path from p_i to p_k in G_T, such that $p_{\alpha_r} = p_s(= (x_s, y_s))$ if $y_s = \min\{y_\alpha \mid (p_{\alpha_{r-1}}, p_\alpha) \in E_T\}$ for all $r = 1, \ldots, m$. The stack S_2 will contain the set of points $\{p_{\alpha_0}(= p_i), p_{\alpha_1}, p_{\alpha_2}, \ldots, p_{\alpha_m}(= p_k)\}$ prior to the processing of the edge (p_i, p_j) (see Fig. 17(a)).

During processing of p_i, we reach the point $p_i \in \mathcal{P}^*$ using the *self-indicator* attached to $p_i \in \mathcal{P}$. Next, we start inspecting the members in $\mathcal{P}^*$ to the right of p_i by sweeping a vertical line from p_i towards right. When the sweep line encounters a point $p_j \in \mathcal{P}^*$, two situations may arise: (i) $y_j < y_i$, and (ii) $y_j > y_i$. In the first case, $(p_i, p_j) \notin E$; we ignore this case and the vertical line sweep is advanced. In the second case, $(p_i, p_j) \in E$; hence, it may form an L_path in the lower stair. Thus, we need to consider the following two sub-cases:

$(p_i, p_j) \in E_T$: In this case, the footprints of p_i are propagated to $FP(p_j)$ as mentioned in Section 4.1.2 (Fig. 13(a)). In order to search for the footprint blocking point, we need to pop the elements

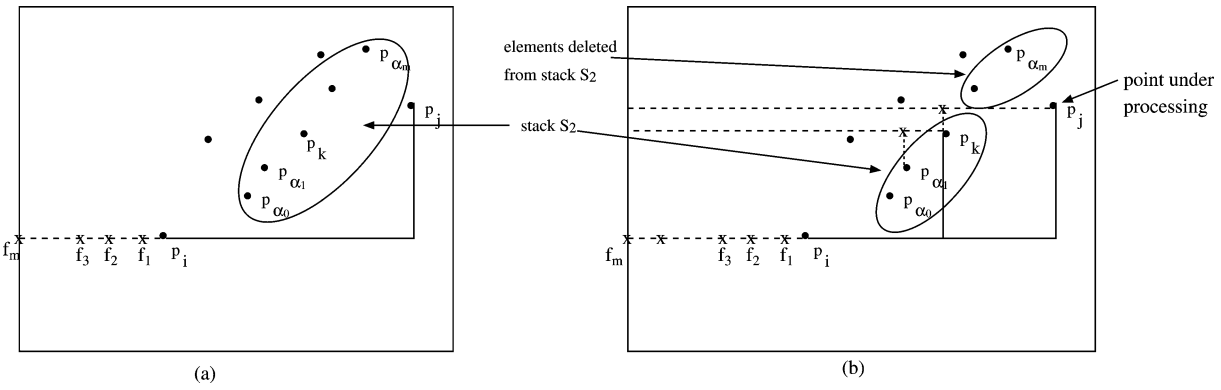


Fig. 17. Processing of an edge $(p_i, p_j) \notin E_T$ in Phase 2; (a) prior to the processing of (p_i, p_j), (b) after processing of (p_i, p_j).

from S_1 until we get a point p_k such that $y_k < y_j$. Let $\{f_1, \ldots, f_r\}$ be the set of footprints in $FP(p_i)$ whose x-coordinates are greater than x_k. All of them are propagated to $FP(p_j)$; the corresponding elements are numbered as $\{g_{s+1}, \ldots, g_{s+r}\}$, whereas $\{g_1, g_2, \ldots, g_s\}$ denotes the set of existing footprints in $FP(p_j)$ prior to processing of (p_i, p_j). In addition, j_k is added to $FP(p_j)$ as the left-most member. We compute the area of the largest MESP from b to p_j whose lower stair terminates at $L\_path(p_i, p_j)$ and upper stair terminates at all the newly generated foorprints of p_j as follows:

$$A(g_{s+\alpha}) = Area\big(\text{LMESP}[p_i, p_j, f_\alpha]\big) = A(f_\alpha) + Area\big(rectangle[f_\alpha, p_j]\big),$$
$$\forall \alpha = 1, \ldots, r,$$
$$A(j_k) = \max_{\beta \geqslant r+1} Area\big(\text{LMESP}[p_i, p_j, f_\beta]\big) = \max_{\beta \geqslant r+1} \big(A(f_\beta) + Area\big(L\_polygon[f_\beta, p_j]\big)\big).$$

By Lemma 7, we may ignore all the elements that are popped out from S_1, since none of them can be a footprint blocking point to the successors of p_i that appear to the right of p_j. Finally, we apply the dominance rule (Lemma 5) to eliminate the set of non-prime footprints from $FP(p_j)$.

The *pred* field of $g_{s+\alpha}$ are set to point to f_α for all $\alpha = 1, 2, \ldots, r$, and the *pred* field of j_k is set to f_β which has contributed to $A(j_k)$.

$(p_i, p_j) \notin E_T$: Here, we must get a point p_k such that $(p_i, p_k) \in E$ and $(p_k, p_j) \in E_T$ (see Fig. 17(a)). The new footprint j_k is added to $FP(p_j)$ (see Lemma 8). By the inheritance rule, $w(f_\alpha, j_k) = w(f_\alpha, g_1) + C$, for all $f_\alpha \in FP(p_i)$, where g_1 is the rightmost footprint in $FP(p_k)$, and $C = Area(rectangle[p^*, p_j])$, and p^* is defined in the proof of Lemma 8. By dominance rule, $A(g_1) = \max_{f_\alpha \in FP(p_i)} Area(\text{LMESP}[p_i, p_k, f_\alpha])$. So,

$$A(j_k) = A(g_1) + Area\big(rectangle[p^*, p_j]\big).$$

The point p_k is obtained by popping the elements of S_2 one by one until we get a point whose y coordinate is less than y_j. Note that, the popped elements of S_2 will not contribute any footprint to the points that appear after p_j during Phase 2 of processing p_i (see Fig. 17(b) for illustration). Finally, the *pred* field of j_k is set to point to g_1 ($\in FP(p_k)$).

Lemma 10. *Phase-2 of complete processing for a point p_i can be completed in $O(T_i + n)$ time, where T_i is the total number of footprints generated in the footprint lists of all the successors of p_i.*

Proof. Let the edge $(p_i, p_j) \in E_T$; then its processing requires $\chi_i(p_j) + O(\theta_1(p_j))$, where $\chi_i(p_j)$ is the number of footprints in $FP(p_i)$, that are propagated to $FP(p_j)$, and $\theta_1(p_j)$ is the number of elements popped from stack S_1. Thus, the total time required for processing all the successors of p_i in G_T is $\sum_{p_j|(p_i,p_j)\in E_T} \chi_i(p_j) + \sum_{p_j|(p_i,p_j)\in E_T} \theta_1(p_j)$. The first term is equal to T_i, and the second term can be at most $O(n)$ from Lemma 9.

If $(p_i, p_j) \in E$, but $\notin E_T$, then the required processing time is $O(\theta_2(p_j))$, where $\theta_2(p_j)$ is the number of elements popped from stack S_2. During complete processing of point p_i, the total number of points inserted in S_2 may be $O(n)$ in the worst case, and an element once deleted from S_2 does not enter again. So, the total time required for processing all such successors of p_i is also $O(n)$. □

4.3.3. Recognition of stairs

While processing p_i, an edge from a footprint f_α ($\in FP(p_i)$) to g_β ($\in FP(p_j)$) is indicated by a backward edge that is created by setting the *pred* field of g_β with the address of f_α. The *pred* field of

g_β is updated if another incoming edge of g_β is recognized, which makes the value of $A(g_\beta)$ higher. At the end of complete processing of all the points in P, if τ is the right-most footprint of t then $A(\tau)$ is maximum (by dominance rule). We report $A(\tau)$ as the area of the largest MESP. Next, we report the lower and upper stairs by traversing the nodes of *SG* from τ through the *pred* fields attached to the max-weight path until b is reached. If $g_\beta \in FP(p_j)$ is reached during backward traversal, and its *pred* field points to $f_\alpha \in FP(p_i)$, then the $L\_path(p_i, p_j)$ is in the lower stair. The upper stair is obtained by sweeping a horizontal line segment $[x_{f_\alpha}, x_{g_\beta}]$ from f_α to g_β in a manner similar to Phase 1. It can be shown that the recognition of the stairs in this pass takes $O(n)$ time in the worst case.

4.3.4. Complexity of the algorithm

Theorem 6. *The largest-area MESP amidst a planar set of n points can be determined in* $O(n^2)$ *time and space.*

Proof. Phase 1 of the complete processing of a point $p_i = (x_i, y_i)$ requires $O(n)$ time (by Lemma 9), and Phase 2 requires $T_i + O(n)$ time (by Lemma 10). Thus, complete processing of all the points in P requires $\sum_{i=1}^{n} T_i + O(n^2)$. Again, by Theorem 2, $\sum_{i=1}^{n} T_i =$ the number of nodes in $SG = O(n^2)$ in the worst case.

The space complexity result follows from the fact that for each element $p_i \in P$, its prime set of footprints $FP(p_i)$ are to be stored for use in the backward pass. □

5. Other applications

5.1. Maximum-width staircase polygon

As mentioned in Section 2, determination of an empty staircase polygon having maximum width turns out to be an useful problem in VLSI layout design. This subsection deals with the method of finding such a MESP. The width of a polygon may be defined using a suitable metric depending on the nature of the problem. Here, we shall define the width of a polygon as follows.

Definition 10. Let α be a point on the lower stair of an empty staircase polygon. The *clearance* of α (denoted as $\omega(\alpha)$) is defined as follows. We use $d(\alpha, \beta)$ to denote the euclidean distance between a pair of points α and β.

- If α is a corner point on the lower stair then $w(\alpha) = d(\alpha, \beta)$, where β is a point on the upper stair, such that $d(\alpha, \beta) \leqslant d(\alpha, \beta')$ for all points β' $(\neq \beta)$ on the upper stair.
- If α is a point on a horizontal edge of the lower stair then $w(\alpha) = d(\alpha, \beta)$, where β is a point on the upper stair such that the vertical line drawn from α meets the upper stair at β.
- If α is a point on a vertical edge of the lower stair then $w(\alpha) = d(\alpha, \beta)$, where β is a point on the upper stair such that the horizontal line drawn from α meets the upper stair at β.

The *width* of the staircase polygon is the minimum clearance among all points on the lower stair.

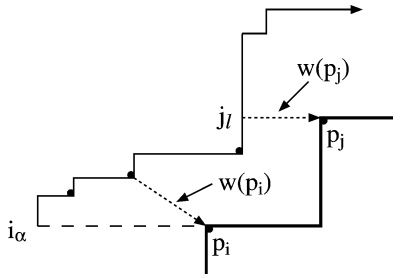


Fig. 18. Width of $L\_polygon[i_\alpha, p_j] = \min\{w(p_i), w(p_j)\}$.

Observation 3. *Let λ be a horizontal* (*respectively vertical*) *edge on the lower stair of a R-staircase polygon, and α be the leftmost* (*respectively top-most*) *point λ. Then $w(\alpha) \leqslant w(\beta)$, for any point β $(\neq \alpha)$ on λ.*

Thus, to compute the width of the R-staircase polygon, the clearances should be computed only at the concave corners of its lower stair and at the bottom-left and top-right corners. The staircase graph (SG) is constructed as before, but the weight of an edge $(i_k, j_\ell) \in E'$ is determined by the width of the $L\_polygon[i_k, p_j]$, which in turn is equal to $\min\{w(p_i), w(p_j)\}$ (see Fig. 18). A directed path in SG from b to the sink node, is now found such that the minimum edge-weight along the path is maximum among all paths in the graph. In order to apply our earlier algorithm for this problem, we need to show that the dominance and inheritance rules, defined earlier, remain valid for this problem.

Assume that the predecessors of p_i in the graph G has already been processed, and let $FP(p_i) = \{f_1, f_2, \ldots, f_m\}$ be the set of footprints of p_i, ordered from right to left. As in the earlier problem, for each member $f_\alpha \in FP(p_i)$, the scalar field $A(f_\alpha)$ denotes the width of the staircase polygon, whose lower stair spans from b to p_i, and the upper stair terminates at f_α.

Consider an edge $(p_i, p_j) \in E$. Let WMESP$[p_i, p_j, f_\alpha]$ denote the widest MESP from b to p_j whose lower stair terminates at $L\_path(p_i, p_j)$, and the upper stair passes through f_α. Then, $width(\text{WMESP}[p_i, p_j, f_\alpha]) = \min\{A(f_\alpha), w(p_i), w(p_j)\}$ (see Fig. 18).

Lemma 11. *For a pair of footprints f_α and f_β of a point p_i, if $\alpha < \beta$ and $A(f_\alpha) < A(f_\beta)$, then $width(\text{WMESP}[p_i, p_j, f_\alpha]) < width(\text{WMESP}[p_i, p_j, f_\beta])$ for all points $\{p_j \mid (p_i, p_j) \in E\}$.*

Proof. If $\alpha < \beta$, then f_α is closer to p_i than f_β in the footprint list $FP(p_i)$. We need to consider two cases:

- If the upper stair of WMESP$[p_i, p_j, f_\alpha]$ does not pass through any point inside the pair of horizontal lines $Y = y_i$ and $Y = y_j$, then $width(L\_polygon[f_\alpha, p_j]) < width(L\_polygon[f_\beta, p_j])$.
- If the upper stair of WMESP$[p_i, p_j, f_\alpha]$ passes through any point inside the pair of horizontal lines $Y = y_i$ and $Y = y_j$, then the upper stair of WMESP$[p_i, p_j, f_\beta]$ will also pass through that point. In that case, $width(L\_polygon[f_\alpha, p_j]) \leqslant width(L\_polygon[f_\beta, p_j])$.

The result follows from the fact that

$$width\big(\text{WMESP}[p_i, p_j, f_\alpha]\big) = \min\big\{A(f_\alpha),\ width\big(L\_polygon[f_\alpha, p_j]\big)\big\}. \qquad \square$$

Lemma 12. *Let* $(p_i, p_k) \in E$, $f_\alpha \in FP(p_i)$ *and* $g_\beta \in FP(p_k)$ *be such that* $(f_\alpha, g_\beta) \in E'$. *In other words,* g_β *is a footprint of* p_k *such that the upper stair of* WMESP$[p_i, p_k, f_\alpha]$ *terminates at* g_β. *Now, if* $(p_i, p_j) \in E$ *be such that* $(p_k, p_j) \in E_T$, *then* $(f_\alpha, j_k) \in E'$ *and* $A(j_k) = \min\{A(g_\beta), (x_j - x_k)\}$.

Proof. Follows from the definition of the *width* of a WMESP. □

Lemma 11 suggests that the *dominance rule* holds among the footprints of any point in P. Lemma 12 says that the inheritance property also holds in this case. So, our earlier algorithm can easily be tailored for recognizing the widest MESP in $O(n^2)$ time and space.

5.2. Largest/widest MESP amidst rectangular obstacles

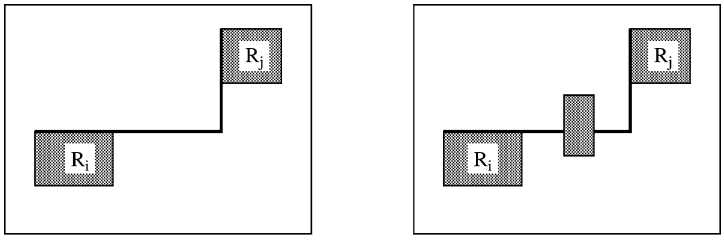
We now outline the modifications of the earlier algorithm and necessary preprocessing for identifying the largest/widest MESP amidst a set of n non-overlapping isothetic rectanglular obstacles. Such a polygon is of interest to VLSI layout design, where the circuit modules are abstracted as solid rectangles on the chip floor, which play the role of obstacles; the empty space among the modules is used for wire routing or repeater placement. Several routing algorithms use the monotone-channel model [16], which essentially is an empty staircase polygon among a set of rectangular obstacles. A solid rectangle is denoted by $R_i[(a_i, b_i), (c_i, d_i)]$, where the points (a_i, b_i) and (c_i, d_i) are its top-left and bottom-right corners, respectively. We will describe the method for finding the largest/widest empty R-staircase polygon only. Each concave corner of the upper (lower) stair of a maximal empty R-staircase polygon coincides with the bottom-right (top-left) corner of a solid rectangle.

To formulate the problem, we need a digraph called the L_visibility graph the role of which is similar to the permutation graph used earlier.

A point $p(x, y)$ is said to be *intercepted* by a solid rectangle $R_i[(a_i, b_i), (c_i, d_i)]$ if $a_i < x < c_i$ and $b_i > y > d_i$.

Definition 11. A solid rectangle $R_j[(a_j, b_j), (c_j, d_j)]$ is L_visible from another solid rectangle $R_i[(a_i, b_i), (c_i, d_i)]$ if (i) $a_i < a_j$, (ii) $b_i < b_j$ and (iii) no point on the *L_path*$[(a_i, b_i), (a_j, b_j)]$ is intercepted by any solid rectangle.

In Fig. 19, we demonstrate the concept of L_visibility. A digraph, called the L_visibility graph, is constructed, each vertex of which represents a solid rectangle. The bottom-left (b) and top-right (t)



(a) R_j is L-visible to R_i (b) R_j is not L-visible to R_i

Fig. 19. Demonstration of L_visibility.

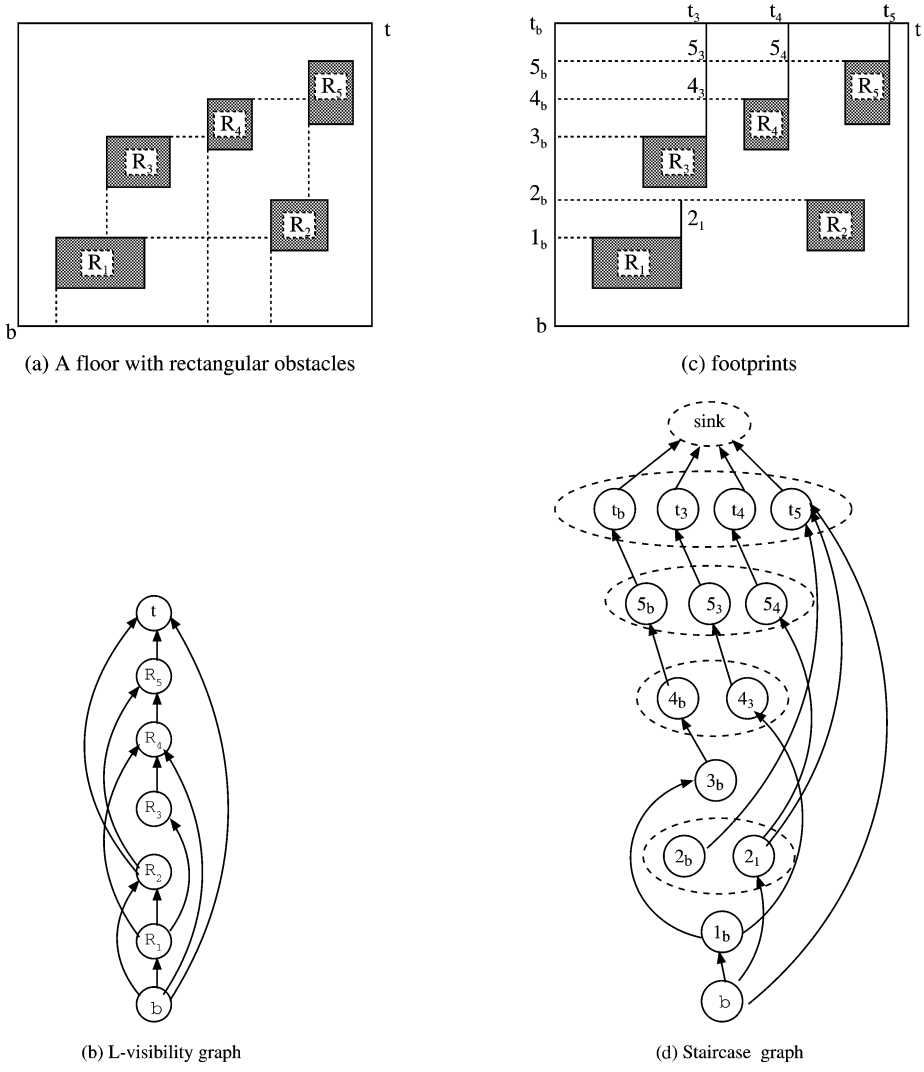


Fig. 20. Determining the MESP among rectangular obstacles.

corners of the floor also appear as vertices representing two degenerate solid rectangles. Two vertices R_1 and R_2 in the L_visibility graph are connected by a directed edge (R_1, R_2) if the rectangle R_2 is L_visible from R_1. An example floor containing a set of solid rectangular obstacles, and its corresponding L_visibility graph are shown in Figs. 20(a) and (b), respectively. The L_visibility graph, among a set of solid rectangles, plays a similar role as that of the digraph G in an ensemble of points, and can be constructed by sweeping a vertical line L from left to right in $O(|E| + n \log n)$ time. The set of footprints and the staircase graph, suited for this situation, are defined as follows.

Definition 12. Let $R_i[(a_i, b_i), (c_i, d_i)]$ and $R_k[(a_k, b_k), (c_k, d_k)]$ denote two solid rectangles such that $c_k \leqslant a_i$, $b_k \leqslant b_i$. If the vertical line segment obtained by joining the points (c_k, b_k) and (c_k, b_i) are not

intercepted by any other rectangles on the floor, then R_k creates a footprint i_k on the horizontal line $Y = b_i$ (the line passing through the top boundary of R_i) at the point (c_k, b_i).

In Fig. 20(c), the footprints are highlighted by putting their labels as in Fig. 7(a). Following the convention of the earlier problem, the set of footprints appeared on the horizontal line adjacent to the top boundary of a rectangle, say $R_i[(a_i, b_i), (c_i, d_i)]$, is denoted as $FP(R_i)$.

The staircase graph is formed with the set of footprints as vertices. For a pair of footprints $i_k \in FP(R_i)$ and $j_\ell \in FP(R_j)$, corresponding to two rectangles $R_i[(a_i, b_i), (c_i, d_i)]$ and $R_j[(a_j, b_j), (c_j, d_j)]$, a directed edge (i_k, j_ℓ) is present in the staircase graph, if the upper stair of a maximal empty *L_polygon*$[(c_k, b_i), (a_j, b_j)]$ passes through j_ℓ, where (c_k, b_i) and (a_j, b_j) are the coordinates of i_k and the top-left corner of the rectangle R_j, respectively. The unweighted staircase graph for the above example is shown in Fig. 20(d). The weight of an edge in this staircase graph is the area of the corresponding L_polygon. It can be easily verified that all the three important geometric properties, i.e., dominance, footprint propagation, and inheritance, defined for the largest MESP problem among point obstacles, hold for this problem. Thus, the earlier algorithm also works in this case with the same time and space complexities. The same technique is applicable to isothetic polygonal obstacles.

6. Conclusion

We have reported an algorithm for identifying the largest empty staircase polygon on a rectangular floor containing a set of n point obstacles. The worst-case time and space complexities of the algorithm are both $O(n^2)$. Novel geometric concepts, namely dominance, footprint propagation, and inheritance are employed for designing the algorithm. A modified version of the same algorithm can be used for identifying the MESP of maximum width with the same time and space complexities. The algorithm is then generalized for finding the largest empty staircase polygon among rectangular or isothetic polygonal obstacles. Identification of empty staircase polygons has manifold applications to VLSI physical design and other areas. The paradigm, described here, may be used for searching other classes of empty zones, for instance, the largest empty orthoconvex polygon of general shape. An open problem is to show whether or not the MESP problem is 3SUM-hard [15].

Acknowledgements

The authors wish to thank the anonymous referees of the paper whose critical comments and suggestions have helped them to improve the presentation of the paper considerably.

Appendix A

Algorithm *MESP*
Input: The point set P.
Output: The two stairs of the MESP.

1. Create an array $\mathcal{P}^*$ by sorting the points in $P \cup \{b, t\}$ in increasing order of their x-coordinates;
2. Next, sort the points in $\mathcal{P}^*$ with respect to their y-coordinates in another array $\mathcal{P}$;
   This helps in attaching to each point in $\mathcal{P}$ its self-indicator in $\mathcal{P}^*$;
2: (* Initialize *)
   $FP(b) \leftarrow \{b\}$; $A(f_0^b) \leftarrow 0$; $FP(p_i) \leftarrow \phi$, $\forall p_i \in P - \{b\}$;
3: (* Forward Pass: Location algorithm *)
   for $i = 1, \ldots, n$ **do** (* Perform complete processing for each point $p_i \in \mathcal{P}$ in order *)
      (* $p_i = (x_i, y_i)$ be the point under complete process *)
3.1: Perform *Phase_1*(p_i) (input: $FP(p_i)$, output: stack S_1);
3.2: Perform *Phase_2*(p_i) (input: $FP(p_i)$, stack S_1, output: the augmented data structure);
   endfor
4: Perform *Recognize_stair* (input: the final data structure, output: the two stairs);
end.

Procedure *Phase_1*(p_i) (* Phase_1 of complete processing *)
Input: $FP(p_i)$
Output: stack S_1; its each element is $(p_\ell, \mathcal{B}(p_\ell), f_\alpha)$ where $\mathcal{B}(p_\ell)$ and f_α are defined in the text
1: *active_span* $= [0, x_i]$; $S_1 = \phi$;
   initialize $\alpha = in(p_i)$ (* it is used for the right to left traversal of $FP(p_i)$ *);
2: **for** $\ell = i, i+1, \ldots, n$ **do** (* point p_ℓ is under processing of Phase 1 *)
      if $x_\ell \in$ *active_span* then
2.1: if $S_1 \neq \phi$ then
      (* let $TOP(S_1) = (p_k, \mathcal{B}(p_k), f_\alpha)$ *) $\mathcal{B}(p_\ell) = \mathcal{B}(p_k) + Area(rectangle[(x_k, y_i), p_\ell])$;
2.2: (* Scan $FP(p_i)$ starting from α *)
      while not $x_{f_\alpha} < x_\ell$ **do**
         $TEMP = A(f_\alpha) + Area(rectangle[f_\alpha, p_k])$;
         if $TEMP > \mathcal{B}(p_\ell)$ **then** $\mathcal{B}(p_\ell) = TEMP$;
         $\alpha = \alpha - 1$;
      endwhile
2.3: push $(p_\ell, \mathcal{B}(p_\ell), f_\alpha)$ in stack S_1;
   endfor
end (* of Phase_1(p_i) *).

Procedure *Phase_2*(p_i) (* Phase_2 of complete processing *)
Input: the data structure $\mathcal{P}$, and stack S_1;
Output: the updated data structure $\mathcal{P}$;
intermediate data structure: a stack S_2.
   (* It contains a path in G_T from p_i to the point whose processing has just completed. *)
1: use *self_indicator* of $p_i \in \mathcal{P}$ to reach $p_i \in \mathcal{P}^*$;
2: **for** $j =$ (index of $p_i \in \mathcal{P}^*$) $\ldots, n$ **do** (* Process the elements in left to right order *)
2.1: pop the elements of stack S_2 until a point p_k is observed such that $y_k < y_j$;
2.2: **if** no such element is found **then** (* $(p_i, p_j) \in E_T$ *)
2.2.1: pop elements from stack S_1 until a point p_k is observed such that $y_k < y_j$;
      (* The triplet corresponding to p_k is $(p_k, \mathcal{B}(p_k), \alpha)$ *)

2.2.2: add j_k in $FP(p_j)$ (* j_k is a footprint in $FP(p_j)$ *);
2.2.3: $A(j_k) = \mathcal{B}(p_k) + Area(rectangle[(x_k, y_i), p_j])$;
2.2.4: set *pred* field of j_k to point the foorprint $f_{\alpha+1} \in FP(p_i)$;
2.2.5: (* eliminate foorprints from $FP(p_j)$ using dominance rule *)
scan elements in $FP(p_j)$ from left towards right to eliminate all $g_\beta \in FP(p_j)$
such that $A(g_\beta) < A(j_k)$
2.2.6: set $\alpha = 1$; (* propagate footprints of $FP(p_i)$ whose x-coordinates are
less than x_k, in $FP(p_j)$ *)
repeat
2.2.6.1: **if** $x_{f_\alpha} > x_k$ then
2.2.6.2: $TEMP = A(f_\alpha) + Area(rectangle[f_\alpha, p_j])$;
2.2.6.3: **if** $TEMP > A(j_k)$ then
2.2.6.3.1: add projection of f_α, say g, in $FP(p_j)$ with $A(g) = TEMP$;
2.2.6.3.2: scan the elements in $FP(p_j)$ from g_β towards right to eliminate
all the footprints $g^* \in FP(p_j)$ such that $A(g^*) < TEMP$;
endif
endif
2.2.7: $\alpha = \alpha + 1$;
until $x_{f_\alpha} < x_k$
2.3: else (* $(p_i, p_j) \notin E_T$—the popped element is p_k *)
2.3.1: if $j_k \notin FP(p_j)$ then add j_k in $FP(p_j)$;
(* apply inheritance rule to calculate $A(j_k)$ *)
2.3.2: $TEMP = A(\tau) + Area(rectangle[(x_k, y_i), p_j])$, where τ be the rightmost element in $FP(p_k)$;
2.3.3: if $TEMP > A(j_k)$ then assign $A(j_k) = TEMP$;
2.3.4: set *pred* pointer of (j_k) to point the footprint $\tau \in FP(p_k)$;
endif
end (* of Phase_2(p_i) *).

Procedure *Recognize_Stair* (* Backward pass *)
1: (* τ is the rightmost element in $FP(t)$ *)
output $A(\tau)$ as the area of the largest MESP;
(* Reporting of staircase: this pass starts from τ and proceeds through the *pred* field
of the footprints until the bottom-left corner b is reached. *)
for all edges on the path from τ to b in SG **do**
(* Let the *pred* field of i_α points to j_β *)
2.1: report $L\_path(i, j)$ as a downward step in the lower stair;
2.2 sweep a horizontal line segment $[x_{i_\alpha}, x_{j_\beta}]$ to recognize the upper stair in
downward direction;
endfor
end Recognize_Stair.

References

[1] A. Aggarwal, M. Klawe, S. Moran, P. Shor, R. Wilber, Geometric applications of a matrix-searching algorithm, Algorithmica 2 (1987) 195–208.

[2] A. Aggarwal, S. Suri, Fast algorithm for computing the largest empty rectangle, in: Proc. 3rd Annual ACM Symposium on Computational Geometry, 1987, pp. 278–290.

[3] A. Asano, M. Sato, T. Ohtsuki, Computational geometric algorithms, in: T. Ohtsuki (Ed.), Layout Design and Verification, Advances in CAD for VLSI, Vol. 4, North-Holland, Amsterdam, 1986, pp. 295–347.

[4] J.E. Boyce, D.P. Dobkin, R.L. Drysdale, L.J. Guibas, Finding extremal polygons, SIAM J. Comput. 14 (1985) 134–147.

[5] S.-W. Cheng, Widest empty L-shaped corridor, Inform. Process. Lett. 58 (1996) 277–283.

[6] B. Chazelle, R.L. Drysdale, D.T. Lee, Computing the largest empty rectangle, SIAM J. Comput. 15 (1986) 300–315.

[7] J. Cong, T. Kong, D.Z. Pan, Buffer block planning for interconnect-driven floorplanning, in: Proc. ACM/IEEE Int. Conf. on Computer-Aided Design (ICCAD), 1999, pp. 358–363.

[8] J. Cong, T. Kong, D.Z. Pan, Interconnect delay estimation models for synthesis and design planning, in: Proc. Asia South Pacific Design Automation Conference, 1999, pp. 97–100.

[9] C.C.N. Chu, D.F. Wong, Closed form solution to simultaneous buffer insertion/sizing and wire sizing, in: Proc. Int. Symp. on Physical Design, 1997, pp. 192–197.

[10] S. Das, B.B. Bhattacharya, Channel routing in Manhattan-diagonal model, in: Proc. 9th Int. Conf. on VLSI Design, IEEE CS Press, 1996, pp. 43–48.

[11] S. Das, S. Sur-Kolay, B.B. Bhattacharya, Routing through a staircase in Manhattan-diagonal model, in: Proc. 11th Int. Conf. on VLSI Design, IEEE CS Press, 1998, pp. 65–70.

[12] P.S. Dasgupta, P. Pan, S.C. Nandy, B.B. Bhattacharya, Monotone bipartitioning problem in a planar point set with applications to VLSI, ACM Trans. Design Automation of Electronic Systems (TODAES) 7 (2) (2002) 231–248.

[13] P.S. Dasgupta, A.K. Sen, S.C. Nandy, B.B. Bhattacharya, Searching networks with unrestricted arc costs, IEEE Trans. Systems Man Cybernat. Part A 31 (2001) 497–507.

[14] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs, Academic Press, New York, 1980.

[15] A. Gajentaan, M.H. Overmars, On a class of $O(n^2)$ problems in computational geometry, Computational Geometry 5 (1995) 165–185.

[16] M. Guruswamy, D.F. Wong, Channel routing order for building-block layout with rectilinear modules, in: Proc. Int. Conf. on Computer-Aided Design (ICCAD), 1988, pp. 184–187.

[17] Y. Kajitani, Order of channels for safe routing and optimal compaction of routing area, IEEE Trans. Computer-Aided Design CAD-2 (1983) 293–300.

[18] S. Majumder, S.C. Nandy, B.B. Bhattacharya, Partitioning VLSI floorplans by staircase channels for global routing, in: Proc. Int. Conf. on VLSI Design, IEEE CS Press, 1998, pp. 59–64.

[19] S. Majumder, S. Sur-Kolay, B.B. Bhattacharya, S.C. Nandy, Area(Number)-balanced hierarchy of staircase channels with minimum crossing nets, in: Proc. Int. Symp. on Circuits and Systems (ISCAS), Vol. 5, 2001, pp. 395–398.

[20] A. Naamad, D.T. Lee, L. Hsu, On the maximum empty rectangle problem, Discrete Appl. Math. 8 (1984) 267–277.

[21] S.C. Nandy, B.B. Bhattacharya, S. Ray, Efficient algorithms for identifying all maximal isothetic empty rectangles in VLSI layout design, in: Proc. 10th. International Conference on Foundation of Software Technology and Theoretical Computer Science, in: Lecture Notes in Computer Science, Vol. 472, Springer-Verlag, Berlin, 1990, pp. 255–269.

[22] G.J.E. Rawlins, D. Wood, Orthoconvexity and its generalizations, in: G. Toussaint (Ed.), Computational Morphology, Elsevier Science, Amsterdam, 1988, pp. 137–152.

[23] N. Sherwani, Algorithms for VLSI Physical Design Automation, 3rd Edition, Kluwer Academic, Boston, 1999.

[24] P. Sarkar, C.-K. Koh, Routability-driven repeater block planning for interconnect-centric floorplanning, IEEE Trans. Computer-Aided Design 20 (2001) 660–671.

[25] S. Sur-Kolay, B.B. Bhattacharya, The cycle structure of nonslicible floorplans and a unified algorithm for feasible routing order, in: Proc Int. Conf. on Computer Design (ICCD), 1991, pp. 524–527.