PAPER

On Two Problems of Nano-PLA Design

Anish Man Singh SHRESTHA†(a), Asahi TAKAOKA†, Nonmembers, Satoshi TAYU†, Member, and Shuichi UENO†, Fellow

SUMMARY The logic mapping problem and the problem of finding a largest sub-crossbar with no defects in a nano-crossbar with nonprogrammable-crosspoint defects and disconnected-wire defects are known to be NP-hard. This paper shows that for nano-crossbars with only disconnected-wire defects, the former remains NP-hard, while the latter can be solved in polynomial time.

key words: biclique problem, nano-crossbar, nano-PLA, orthogonal ray graphs, subgraph isomorphism problem

1. Introduction

Implementing a sum-of-product logic function in a conventional programmable logic array (PLA) is a straightforward task of arbitrarily assigning the literals and product terms to the wires of the crossbar and programming the appropriate crosspoints. However, in the case of nano-PLAs, this task is not trivial because of imperfections in the nano-wire crossbar. Defects in nano-wire crossbar have been broadly classified into two types: nonprogrammable-crosspoint defects, in which some crosspoints become unprogrammable, and disconnected-wire defects, in which each horizontal nanowire may not be connected to all vertical nano-wires [5]. The problem of mapping a sum-of-product logic function onto a defective nano-crossbar with nonprogrammable-crosspoint defects and disconnected-wire defects was first considered by Rao, Orailoglu, and Karri [5]. They proposed several heuristics since the problem is NP-hard. The problem of finding a maximum defect-free sub-crossbar in a nano-crossbar with nonprogrammable-crosspoint defects and disconnected-wire defects was first investigated by Tahoori [8]. Since the problem is also NP-hard, several heuristics have been proposed [1], [8].

This paper considers the complexity of the problems for nano-crossbars with only disconnected-wire defects.

1.1 LOGIC MAPPING

Let \( f \) be a logic function in a sum-of-product form. Let \( S \) be a nano-crossbar with disconnected-wire defects. The problem of implementing \( f \) in \( S \) is formulated as LOGIC MAPPING, which is the problem of assigning the literals and product terms of \( f \) to nano-wires of \( S \) so that containment relationships among the literals and product terms can be represented by crosspoint connections in \( S \). A graph model of LOGIC MAPPING can be obtained as follows.

Let \( L_f \) be the set of literals of \( f \), and \( P_f \) be the set of product terms of \( f \). A logic function graph \( G_f \) for \( f \) is a bipartite graph defined as follows: \( V(G_f) = L_f \cup P_f \), and \( (L_f, P_f) \) is a bipartition of \( G_f \); vertices \( l \in L_f \) and \( p \in P_f \) are connected by an edge if and only if literal \( l \) is contained in product term \( p \).

Let \( W_h \) be the set of horizontal nano-wires, and \( W_v \) be the set of vertical nano-wires of \( S \). A crossbar graph \( G_S \) of \( S \) is a bipartite graph defined as follows: \( V(G_S) = W_h \cup W_v \) and \( (W_h, W_v) \) is a bipartition of \( G_S \); vertices \( x \in W_h \) and \( y \in W_v \) are connected by an edge if and only if nano-wires \( x \) and \( y \) have a crosspoint. Then, LOGIC MAPPING can be modeled as the subgraph isomorphism problem, which is to find a subgraph of \( G_S \) isomorphic to \( G_f \). Examples of a logic function \( f \), a defective crossbar \( S \), and their corresponding bipartite graphs \( G_f \) and \( G_S \) are shown in Fig. 1.

1.2 SUB-CROSSBAR

SUB-CROSSBAR is the problem of finding a defect-free sub-crossbar consisting of given numbers of horizontal and vertical wires within the nano-crossbar with disconnected wire defects. SUB-CROSSBAR can be modeled as the \( K_{m,n} \)-biclique problem, which is to find a complete bipartite subgraph \( K_{m,n} \) contained in a crossbar graph \( G_S \).

Fig. 1 An instance of LOGIC MAPPING and the corresponding graphs.
1.3 Our Results

Although it is well known that both the subgraph isomorphism problem and the $K_{m,n}$-biclique problem are NP-hard for bipartite graphs [2], [3], the complexity of LOGIC MAPPING and SUB-CROSSBAR is not immediately clear since the graphs representing surviving sub-crossbars are a special kind of bipartite graph.

A bipartite graph $G$ with a bipartition $(U, V)$ is called an orthogonal ray graph if there exist a set of non-intersecting rays (half-lines) $R_u, u \in U$, parallel to the $x$-axis in the $xy$-plane, and a set of non-intersecting rays $R_v, v \in V$, parallel to the $y$-axis such that for any $u \in U$ and $v \in V$, $(u, v) \in E(G)$ if and only if $R_u$ and $R_v$ intersect. An orthogonal ray graph $G$ with a bipartition $(U, V)$ is called a two-directional orthogonal ray graph if $R_u$ is a downward ray $[(a_u, b_u) \mid x \geq a_u]$ for each $u \in U$, and $R_v$ is an upward ray $[(a_v, b_v) \mid y \geq b_v]$ for each $v \in V$, where $a_u$ and $b_u$ are real numbers for any $w \in U \cup V$.

Nano-wires such as $p$ and $q$ of a defective nano-crossbar shown in Fig. 2 cannot be controlled as they do not reach the boundary of the originally intended nano-crossbar. Since we cannot use such nano-wires, a graph representing a surviving sub-crossbar must be an orthogonal ray graph.

We show in Sect. 3 that LOGIC MAPPING is NP-hard by showing that the subgraph isomorphism problem is NP-hard even for orthogonal ray graphs. We show in Sect. 4 that SUB-CROSSBAR can be solved in polynomial time provided that the vertices of the orthogonal ray graph representing a surviving sub-crossbar are ordered to reflect the top-to-bottom order of horizontal nano-wires and left-to-right order of vertical nano-wires. This is a quite natural condition. We also show in Sect. 4 that in the case of two-directional orthogonal ray graphs, the $K_{m,n}$-biclique problem can be solved in polynomial time without the requirement of such an ordering, thereby providing a purely graph-theoretic solution for an interesting subproblem of SUB-CROSSBAR.

2. Orthogonal Ray Graphs

In this section, we shall discuss some properties of two-directional orthogonal ray graphs that will come in handy in the later sections. Some of the lemmas and theorems in this section also appear in our earlier work [7]. In order to make the paper self-contained, we revisit them and also provide direct explicit proofs for some of them.

The 3-claw is a tree obtained from a complete bipartite graph $K_{1,3}$ by replacing each edge with a path of length 3. (See Fig. 3 (a).)

**Lemma 1.** The 3-claw is not a 2-directional orthogonal ray graph.

**Proof.** Assume to the contrary that the 3-claw is a 2-directional orthogonal ray graph. Let the vertices of the 3-claw be named as in Fig. 3 (a). We shall refer to the endpoint of the ray corresponding to a vertex $v$ by $(a_v, b_v)$. Without loss of generality, suppose $R_{u_1}$ is a horizontal ray and that $R_{v_1}, R_{v_2}, R_{v_3}$ intersect with $R_{u_1}$ such that $R_{u_1}$ lies to the right of $R_{v_1}$ and to the left of $R_{v_2}$, (See Fig. 3 (b)). It is easy to observe that $b_{v_1} > b_{v_2} > b_{v_3}$, or else it is not possible to define $R_{u_1}, R_{u_2}$, and $R_{u_3}$. Since $R_{u_1}$ has to be defined such that $a_{u_1} > a_{v_1}$ and $b_{u_1} < b_{v_1}$, it is not possible to define $R_{v_3}$ such that it intersects with $R_{u_1}$ but not with $R_{u_1}$, a contradiction.

A path $P$ in a tree $T$ is called a spine of $T$ if every vertex of $T$ is within distance two from at least one vertex of $P$.

**Theorem 1.** A tree $T$ has a spine if and only if $T$ contains no 3-claw as a subtree.

**Proof.** The necessity is obvious. To prove the sufficiency, assume $T$ contains no 3-claw. Let $P$ be a longest path in $T$, and let $V(P) = \{v_1, v_2, \ldots, v_p\}$ and $(v_i, v_{i+1}) \in E(P), 1 \leq i \leq p - 1$. We claim that $P$ is a spine. We distinguish two cases: $|V(P)| \leq 6$ and $|V(P)| > 6$.

For the former, it is easy to see that $P$ is a spine because if there is a vertex $v \notin V(P)$ which is at a distance more than two from any vertex in $P$, then the assumption that $P$ is a longest path is contradicted.

We next take the case of $|V(P)| > 6$. Assume $P$ is not a spine. Let $F$ be a forest obtained from $T$ by deleting the edges in $E(P)$. Let $T_1$ be a tree in $F$ containing $v_i, 1 \leq i \leq p$. Since $P$ is a longest path in $T$, $T_1$ consists of only one vertex, $v_1$, and $T_p$ consists of only one vertex, $v_p$. Also all vertices in $T_2$ and $T_{p-1}$ are within distance one from $v_2$ and $v_{p-1}$, respectively; and all vertices in $T_3$ and $T_{p-2}$ are within distance two from $v_3$ and $v_{p-2}$, respectively. Since we assumed that $P$ is not a spine, there exists an integer $j$ $(4 \leq j \leq p - 3)$ such that $T_j$ contains a vertex $w_j$ whose distance from $v_j$ is three. Let $P'$ be the path from $v_j$ to $w_j$. Then the subgraph of $T$ induced by the vertices in $\{v_i \mid j-3 \leq i \leq j+3\} \cup V(P')$ is a 3-claw. This contradicts the assumption that $T$ contains no 3-claw as a subtree, and therefore $P$ is a spine. □
Theorem 2. A tree $T$ is a 2-directional orthogonal ray tree if and only if $T$ contains no 3-claw as a subtree.

Proof. The necessity follows from Lemma 1. We will show the sufficiency. Assume $T$ contains no 3-claw as a subtree. Then from Theorem 1, $T$ contains a spine $P$. Let $V(P) = \{v_1, v_2, \ldots, v_p\}$, and $(v_i, v_{i+1}) \in E(P)$, $1 \leq i \leq p - 1$. Corresponding to each vertex $v_i$ in $P$, define ray $R_{v_i} = \{(i, y) \mid y \geq i - 1\}$ if $i$ is odd, and define ray $R_{v_i} = \{(i, y) \mid x \geq i - 1\}$ if $i$ is even. Let $F$ be a forest obtained from $T$ by deleting the edges in $E(P)$. Let $T_i$ be a tree in $T$ containing $v_i$, $1 \leq i \leq p$. Consider $T_i$ to be rooted at $v_i$. Let $w_{ij}, w_{i2}, \ldots, w_{iqt(i)}$ be the children of $v_i$ in $T_i$, where $q(i)$ is the number of children of $v_i$ in $T_i$. Let $z_{ij}, z_{i2}, \ldots, z_{iqt(i)}$ be the children of $w_{ij}$ in $T_i$, where $r(i)$ is the number of children of $w_{ij}$ in $T_i$. The rays corresponding to $w_{ij}$ and $z_{ik}$, $(1 \leq i \leq q, 1 \leq j \leq r(i), 1 \leq k \leq r(i))$ can be placed in the region for $T_i$ as shown in Fig. 4. Thus $T$ is a 2-directional orthogonal ray graph. □

Lemma 2. A cycle $C_{2m}$ of length $2m$ is a two-directional orthogonal ray graph if and only if $m = 2$.

Proof. It is easy to see that $C_4$ is a 2-directional orthogonal ray graph.

We show that $C_{2m}$ is not a 2-directional orthogonal ray graph for any $m \geq 3$. Suppose to the contrary that $C_{2m}$ is a 2-directional orthogonal ray graph for some $m \geq 3$. Let $V(C_{2m}) = \{0, 1, \ldots, 2m - 1\}$ and $E(C_{2m}) = \{(i, i + 1 \pmod{2m}) \mid 0 \leq i \leq 2m - 1\}$. Suppose without loss of generality that $R_0 = \{(a_0, y) \mid y \geq b_0\}$, for some real numbers $a_0$ and $b_0$. Since $(0, 1) \in E(C_{2m})$, $R_1$ intersects with $R_0$ at some point. Similarly, $R_2$ intersects with $R_1$ at some other point. We distinguish two cases.

Case 1 When $R_2$ intersects with $R_1$ such that $R_2$ is to the left of $R_0$: Then $R_1$ must intersect with $R_2$ such that $R_1$ lies below the endpoint of $R_0$. Similarly, $R_4$ must intersect with $R_3$ such that $R_4$ lies to the left of the endpoint of $R_3$. Continuing in this manner, $R_i$ $(5 \leq i \leq 2m - 1)$ must lie below (to the left of) the endpoint of $R_{i-3}$ for odd (even) $i$. Therefore $R_{2m-1}$ lies in the region below the endpoint of $R_4$. However, $R_0$ is in the region right of $R_2$ and above $R_3$, making it impossible for $R_0$ to intersect with $R_{2m-1}$ without intersecting with $R_3, R_5, \ldots, R_{2m-3}$, a contradiction.

Case 2 When $R_2$ intersects with $R_1$ such that $R_2$ is to the right of $R_0$: We further distinguish two cases.

Case 2-1 When $R_1$ intersects with $R_2$ such that $R_1$ is below the endpoint of $R_2$: Then $R_4$ must lie to the left of the endpoint of $R_1$. This confines $R_0$ within the region left of $R_2$ and above $R_3$, making it impossible for ray $R_{2m-1}$ to intersect with $R_0$ without intersecting with $R_2$, a contradiction.

Case 2-2 When $R_1$ intersects with $R_2$ such that $R_1$ is above $R_2$: This case may be further broken down into two cases depending on whether $R_4$ is to the left of $R_2$ or right of $R_2$. In the former case, $R_4$ gets confined within the region left of $R_2$ and above $R_1$ making it impossible for $R_5$ to intersect with $R_4$ without intersecting with $R_2$, a contradiction. In the latter case, $R_5, \ldots, R_{2m-1}$ must lie in the region right of $R_2$ and above $R_3$, making it impossible for $R_{2m-1}$ to intersect with $R_0$ without intersecting with $R_2, R_4, \ldots, R_{2m-2}$, a contradiction.

Thus we conclude that $C_{2m}$ is not a 2-directional orthogonal ray graph for any $m \geq 3$. □

A bipartite graph is chordal if it contains no induced cycles of length greater than 4. A tree is chordal, by definition. Thus, by Lemma 2 and Theorem 2, we have:

Theorem 3. A class of two-directional orthogonal ray graphs is a proper subset of the class of chordal bipartite graphs. □

3. Intractability of LOGIC MAPPING

We show in this section the following.

Theorem 4. LOGIC MAPPING is NP-hard.

Theorem 4 follows from Theorem 5 below. A decision problem associated with the subgraph isomorphism problem is defined as follows:

SUBGRAPH ISOMORPHISM

INSTANCE: Graphs $H$ and $G$.

QUESTION: Does $G$ contain a subgraph isomorphic to $H$, that is, does there exist a one-to-one mapping $\phi : V(H) \rightarrow V(G)$ such that if $(u, v) \in E(H)$ then $(\phi(u), \phi(v)) \in E(G)$?

Theorem 5. SUBGRAPH ISOMORPHISM is NP-complete even if $G$ is a 2-directional orthogonal ray tree and $H$ is a forest.

Proof. It is easy to see that the problem is in NP. We show a polynomial-time reduction from 3-PARTITION, which has been shown to be strongly NP-complete in [2]. 3-PARTITION is defined as follows.
3-PARTITION

INSTANCE: A finite set $A$ of $3m$ elements, a bound $B \in \mathbb{Z}^+$, and a size $s(a) \in \mathbb{Z}^+$ for each $a \in A$, such that each $s(a)$ satisfies $B/4 < s(a) < B/2$ and such that $\sum_{a \in A} s(a) = mB$.

QUESTION: Does $A$ have a 3-partition, that is, can $A$ be partitioned into $m$ disjoint sets $S_1, S_2, \ldots, S_m$ such that, for $1 \leq i \leq m$, $\sum_{a \in S_i} s(a) = B$?

Let $C_1, C_2, \ldots, C_m$ be be complete binary trees of height two. For each $C_i$, let $T_1, T_2, \ldots, T_{|S_i|-1}$ be complete binary trees of height two rooted at vertices $v_{1,1}, v_{1,2}, \ldots, v_{1,|S_i|}$, respectively. Let $G$ be the graph defined as

$$ V(G) = \left( \bigcup_{i=1}^{m} V(C_i) \right) \cup \left( \bigcup_{i=1}^{m} V(T_i) \right), $$

$$ E(G) = \left( \bigcup_{i=1}^{m} E(C_i) \right) \cup \left( \bigcup_{i=1}^{m} E(T_i) \right) \cup \{(r_i, v_{i,1}), (r_i, v_{i,|S_i|}), \; 1 \leq i \leq m - 1\}. $$

(See Fig. 5(a).) Since the path in $G$ from $v_{1,1}$ to $v_{m,B}$ is a spine of $G$, it follows from Theorems 1 and 2 that $G$ is a two-directional orthogonal ray tree. Let $H$ be a forest consisting of $m-1$ complete binary trees of height two $T'_1, T'_2, \ldots, T'_{m-1}$, and $3m$ paths $P_1, P_2, \ldots, P_{3m}$, each $P_j$ corresponding to an element $a_j$ of $A$ and having $s(a_j)$ vertices. (See Fig. 5(b).) $G$ and $H$ can be constructed in time polynomial in $m$ and $B$.

We next prove that $A$ has a 3-partition if and only if $G$ contains a subgraph isomorphic to $H$.

Suppose first that $A$ can be partitioned into $m$ disjoint subsets $S_1, S_2, \ldots, S_m$ such that for each $i (1 \leq i \leq m)$, $\sum_{a \in S_i} s(a) = B$. An isomorphism from $H$ to a subgraph of $G$ can be obtained as follows. Since each path $C_i$ contains $B$ vertices, we can map the paths of $H$ corresponding to the elements of $S_i$ to the path $C_i$ in $G$. Each $T'_i$ in $H$ can be mapped to $T_i$ in $G$. It is easy to see that this is indeed an isomorphism from $H$ to a subgraph of $G$.

Next suppose that $H$ is isomorphic to a subgraph of $G$. Each $T'_i (1 \leq j \leq m - 1)$ in $H$ contains two vertices which have degree three and are at a distance two from each other. These vertices must be mapped to the children of vertex $r_i$ of $T_i$ for some $i (1 \leq i \leq m - 1)$. Therefore, each $T'_i$ in $H$ must be mapped to some $T_i$ in $G$. This means that paths $P'_1, P'_2, \ldots, P'_m$ in $H$ are mapped to paths $C_1, C_2, \ldots, C_m$ in $G$. For $1 \leq i \leq m$, let $S_i$ be the set of elements of $A$ corresponding to the paths of $H$ mapped to $C_i$. Since $C_i$ has $B$ vertices, $\sum_{a \in S_i} s(a) \leq B$, for all $i (1 \leq i \leq m)$. Moreover, since the instance of 3-PARTITION satisfies $\sum_{a \in A} s(a) = mB$, we can conclude that $\sum_{a \in S_i} s(a) = B$ for all $i (1 \leq i \leq m)$. Therefore $A$ has a 3-partition. □

4. Tractability of SQUARE SUB-CROSSBAR

Let $\mathcal{H}$ be a set of non-intersecting horizontal rays, and let $\mathcal{V}$ be a set of non-intersecting vertical rays. Let $\mathcal{K}_h \subseteq \mathcal{H}$ and $\mathcal{K}_v \subseteq \mathcal{V}$. $\mathcal{K}_h \cup \mathcal{K}_v$ is called a $|\mathcal{K}_h| \times |\mathcal{K}_v|$ sub-crossbar of $\mathcal{H} \cup \mathcal{V}$ if each $X \in \mathcal{K}_h$ intersects every $Y \in \mathcal{K}_v$. For a ray $R$, we shall denote the $x$- and $y$-coordinates of its endpoints by $x(R)$ and $y(R)$, respectively. We associate with $\mathcal{H} \cup \mathcal{V}$, a sequence $X_{nR}$ of the rays of $\mathcal{H} \cup \mathcal{V}$ sorted in the increasing order of $x$-coordinate values of the end points – ties are broken such that if a horizontal ray and a vertical ray have the same $x$-coordinate value, then the horizontal ray appears before the vertical ray in the sequence. We also associate with $\mathcal{H} \cup \mathcal{V}$, a sequence $Y_{nR}$ of the rays of $\mathcal{H} \cup \mathcal{V}$ sorted in the increasing order of $y$-coordinate values of the end points – ties are broken such that if a vertical ray and a horizontal ray have the same $y$-coordinate value, then the vertical ray appears before the horizontal ray in the sequence.

Our earlier observation that a nano-wire crossbar can be represented by a set of orthogonal rays allows us to use the terms “nano-wires” and “rays” interchangeably. Then an alternate, equivalent definition of SUB-CROSSBAR is as follows:

SUB-CROSSBAR

INSTANCE: A set $\mathcal{H}$ of horizontal rays, a set $\mathcal{V}$ of vertical rays, and positive integers $k_h$ and $k_v$. 
QUESTION: Show a $k_h \times k_r$ sub-crossbar of $\mathcal{H} \cup \mathcal{V}$, if any.

An interesting subproblem of SUB-CROSSBAR in which the instance is restricted to rightward and upward rays can be defined as follows:

**2-SUB-CROSSBAR**

**INSTANCE:** A set $\mathcal{R}$ of rightward rays, a set $\mathcal{U}$ of upward rays, and positive integers $k_r$ and $k_u$.

**QUESTION:** Show a $k_r \times k_u$ sub-crossbar of $\mathcal{R} \cup \mathcal{U}$, if any.

In the following subsections, we will discuss algorithms to solve these problems.

### 4.1 Algorithms for 2-SUB-CROSSBAR

Kloks and Kratsch [4] showed the following.

**Lemma 3.** [4] A chordal bipartite graph with $n$ vertices and $m$ edges contains at most $m$ maximal complete bipartite subgraphs which can be enumerated in $O(\min(m \log n, n^2))$ time. \hfill $\Box$

From Lemma 3 and Theorem 3, we have:

**Lemma 4.** The $K_{m,n}$ biclique problem can be solved in $O(\min(m \log n, n^2))$ time for $n$-vertex, $m$-edge 2-directional orthogonal ray graphs. \hfill $\Box$

Since the graph representing $\mathcal{R} \cup \mathcal{U}$ is a 2-directional orthogonal ray graph, we have the following theorem from the above lemma.

**Theorem 6.** 2-SUB-CROSSBAR can be solved in $O(\min(m \log n, n^2))$ time for a crossbar, where $n = |\mathcal{R}| + |\mathcal{U}|$ and $m$ is the number of crosspoints. \hfill $\Box$

This is a purely graph theoretic approach, which assumes no information about the endpoints of rays. Takahashi [9] showed that a computational geometry approach utilizing the coordinates of the endpoints yields a faster algorithm of time complexity $O(n \log n)$. We present Algorithm 1 (See Fig. 6), which is a linear-time algorithm to solve 2-SUB-CROSSBAR given that sequences $X_{RU}$ and $Y_{RU}$ are provided. Since $X_{RU}$ and $Y_{RU}$ can be computed in $O(n \log n)$ time, Algorithm 1 can be easily extended to solve 2-SUB-CROSSBAR in $O(n \log n)$ time. However, the main purpose of introducing Algorithm 1 is to use it as a subroutine to solve SUB-CROSSBAR, as shown in the next subsection.

A brief description of Algorithm 1 follows. Algorithm 1 begins with some preprocessing operations, in which the sequences $R$, $U$ and the sets $uEnd(i) (1 \leq i \leq |\mathcal{R}|)$, $rEnd(j) (1 \leq j \leq |\mathcal{U}|)$ are computed (see Steps 1 and 2). To search for a $k_r \times k_u$ sub-crossbar, Algorithm 1 uses two sweep lines to perform a left-to-right, bottom-to-top scan of the rays. The horizontal sweep line stops at $R_1, R_2, \ldots$, and it is represented by variable $h$, which indicates that it is at the position of ray $R_h$. The vertical sweep line stops at $U_1, U_2, \ldots$, and it is represented by variable $v$, which indicates that it is at the position of ray $U_v$. At each stop, the following processes are carried out.

Fig. 6 Algorithm 1.
computed (Step 4). Similarly, the number uCross of verticalrays that cross the horizontal sweep line and lie in the area
right of, and including, the vertical sweep line, is computed
(Step 5). Evidently, if rCross ≥ k_h and uCross ≥ k_v, then
there exists a k_h×k_v sub-crossbar, which is output (Step 6). If
rCross < k_h, then ray U_i is not a part of any k_i×k_v subcrossbar,
and therefore it is removed from further consideration
by updating the appropriate set uEnd(i) and uCross (Step 7).
The vertical sweep line then moves one step right to the po-
tion of ray U_{i+1}. If uCross < k_v, identical operations are
carried out for the horizontal case (Step 8).

Let $n = |\mathcal{R}| + |\mathcal{V}|$. The items in Steps 1 and 2 can be ob-
tained from the given sequences $X_{\mathcal{RL}}$ and $Y_{\mathcal{RL}}$ in $O(n)$ time.
Each operation in Steps 3 through 8 can be performed in
$O(1)$ time. Steps 4 through 9 are repeated until there are less
than $k_v$ vertical rays or less than $k_h$ horizontal rays remain-
ing to be scanned, which is $O(n)$ times. Thus the algorithm
is linear in the order of the total number of rays. The cor-
rectness of Algorithm 1 is obvious, and therefore we have
the following theorem.

**Theorem 7.** Algorithm 1 solves 2-SUB-CROSSBAR in
$O(|\mathcal{R}| + |\mathcal{V}|)$ time, provided that the sequences $X_{\mathcal{RL}}$ and $Y_{\mathcal{RL}}$
are given. 

4.2 Algorithm for SUB-CROSSBAR

Let $\mathcal{H}$ be a set of non-intersecting horizontal rays and $\mathcal{V}$ be
a set of non-intersecting vertical rays. For two rays $H \in \mathcal{H}$
and $V \in \mathcal{V}$ which intersect, say at point $(p_x, p_y)$, define

$$\mathcal{H}_{HV} = \{ R | R \in \mathcal{H}, R \text{ intersects } V, \text{ and } y(R) \leq p_y \}.$$ 

Similarly, define

$$\mathcal{V}_{HV} = \{ R | R \in \mathcal{V}, R \text{ intersects } H, \text{ and } x(R) \leq p_x \}.$$ 

Let $B$ be the bottommost ray in $\mathcal{H}_{HV}$, and let $L$ be the left-
most ray in $\mathcal{V}_{HV}$. For each ray $R \in \mathcal{H}_{HV}$, define ray $R'$ such
that if $R$ is a rightward ray, $R' = R$; and if $R$ is a leftward ray,
$R'$ is a rightward ray with $x(R') = x(L)$ and $y(R') = y(R)$. 
And for each ray $R \in \mathcal{V}_{HV}$, define ray $R'$ such that if $R$ is
an upward ray, $R' = R$; and if $R$ is a downward ray, $R'$ is
an upward ray with $x(R') = x(R)$ and $y(R') = y(B)$. Finally,
define

$$\mathcal{H}'_{HV} = \{ R' | R \in \mathcal{H}_{HV} \}$$

and

$$\mathcal{V}'_{HV} = \{ R' | R \in \mathcal{V}_{HV} \}.$$ 

Figure 7 shows an example of $\mathcal{H}_{HV}$, $\mathcal{V}_{HV}$, $\mathcal{H}'_{HV}$, and $\mathcal{V}'_{HV}$.

The following observation is obvious from the definitions
above.

**Observation 1.** Two rays in $\mathcal{H}_{HV} \cup \mathcal{V}'_{HV}$ intersect if and
only if their corresponding rays in $\mathcal{H}'_{HV} \cup \mathcal{V}'_{HV}$ intersect. 

**Observation 2.** $\mathcal{H} \cup \mathcal{V}$ contains a $k_h \times k_v$ sub-crossbar if
and only if there exists a pair of intersecting rays $H \in \mathcal{H}$
and $V \in \mathcal{V}$ such that $\mathcal{H}'_{HV} \cup \mathcal{V}'_{HV}$ contains a $k_h \times k_v$ subcrossbar.
The sufficiency is immediate from Observation 1. To see the necessity, set \( H \) and \( V \) to be the topmost and rightmost rays, respectively of a \( k_h \times k_v \) sub-crossbar of \( \mathcal{H} \cup \mathcal{V} \).

Since \( \mathcal{H}_{HV} \) contains only rightward rays and \( \mathcal{V}_{HV} \) contains only upward rays, we can use Algorithm 1 to find a \( k_h \times k_v \) sub-crossbar in \( \mathcal{H}_{HV} \cup \mathcal{V}_{HV} \). Algorithm 2 which solves SUB-CROSSBAR is shown in Fig. 8. It exhaustively checks all pairs of intersecting rays to determine if there exists a pair \( H \in \mathcal{H} \) and \( V \in \mathcal{V} \) such that \( \mathcal{H}_{HV} \cup \mathcal{V}_{HV} \) contains a \( k_h \times k_v \) sub-crossbar.

Let \( n = |\mathcal{H}| + |\mathcal{V}| \). Step 1 can be performed in \( O(n \log n) \) time. Step 2 takes \( O(n^2) \) time. The items in Step 4 can be computed in \( O(n) \) time from the sequences obtained in Step 1. Step 5 takes \( O(n) \) time. Steps 3 through 6 are repeated \( O(n^2) \) time. Then it follows from Observation 2 and Theorem 7 that:

**Theorem 8.** Algorithm 2 solves SUB-CROSSBAR in \( O((|\mathcal{H}| + |\mathcal{V}|)^3) \) time.

## 5. Concluding Remarks

The complexity of SUBGRAPH ISOMORPHISM in which \( G \) is a 2-directional orthogonal ray graph and \( H \) is a connected graph is open. Note that if both \( G \) and \( H \) are trees, then SUBGRAPH ISOMORPHISM is polynomial-time solvable [2]. Reducing the time complexity of SUB-CROSSBAR is another interesting open question.

A preliminary version of this paper has appeared in [6].

### References


