On the Recognition of Four-Directional Orthogonal Ray Graphs

Stefan Felsner\textsuperscript{1}, George B. Mertzios\textsuperscript{2}, and Irina Mustat\textashvili\textsuperscript{1}

\textsuperscript{1} Institut für Mathematik, Technische Universität Berlin, Germany. Email: \{felsner,mustata\}@math.tu-berlin.de
\textsuperscript{2} School of Engineering and Computing Sciences, Durham University, UK. Email: george.mertzios@durham.ac.uk

Abstract. Orthogonal ray graphs are the intersection graphs of horizontal and vertical rays (i.e. half-lines) in the plane. If the rays can have any possible orientation (left/right/up/down) then the graph is a 4-directional orthogonal ray graph (4-DORG). Otherwise, if all rays are only pointing into the positive $x$ and $y$ directions, the intersection graph is a 2-DORG. Similarly, for 3-DORGs, the horizontal rays can have any direction but the vertical ones can only have the positive direction. The recognition problem of 2-DORGs, which are a nice subclass of bipartite comparability graphs, is known to be polynomial, while the recognition problems for 3-DORGs and 4-DORGs are open. Recently it has been shown that the recognition of unit grid intersection graphs, a superclass of 4-DORGs, is NP-complete. In this paper we prove that the recognition problem of 4-DORGs is polynomial, given a partition \{L,R,U,D\} of the vertices of $G$ (which corresponds to the four possible ray directions).

For the proof, given the graph $G$, we first construct two cliques $G_1, G_2$ with both directed and undirected edges. Then we successively augment these two graphs, constructing eventually a graph $\tilde{G}$ with both directed and undirected edges, such that $G$ has a 4-DORG representation if and only if $\tilde{G}$ has a transitive orientation respecting its directed edges. As a crucial tool for our analysis we introduce the notion of an $S$-orientation of a graph, which extends the notion of a transitive orientation. We expect that our proof ideas will be useful also in other situations. Using an independent approach we show that, given a permutation $\pi$ of the vertices of $U$ ($\pi$ is the order of $y$-coordinates of ray endpoints for $U$), while the partition \{L,R\} of $V \setminus U$ is not given, we can still efficiently check whether $G$ has a 3-DORG representation.

1 Introduction

Segment graphs, i.e. the intersection graphs of segments in the plane, have been the subject of wide spread research activities (see e.g. [2,12]). More tractable

\footnotesize{*This work was partially supported by (i) the DFG ESF EuroGIGA projects COMPOSE and GraDR, (ii) the EPSRC Grant EP/K022660/1, (iii) the EPSRC Grant EP/G043434/1, and (iv) the Berlin Mathematical School.}
subclasses of segment graphs are obtained by restricting the number of directions for the segments to some fixed positive integer $k$ $[4,11]$. These graphs are called $k$-directional segment graphs. For the easiest case of $k = 2$ directions, segments can be assumed to be parallel to the $x$- and $y$-axis. If intersections of parallel segments are forbidden, then 2-directional segment graphs are bipartite and the corresponding class of graphs is also known as grid intersection graphs (GIG), see $[9]$. The recognition of GIGs is NP-complete $[10]$.

Since segment graphs are a fairly complex class, it is natural to study the subclass of ray intersection graphs $[1]$. Again, the number of directions can be restricted by an integer $k$, which yields the class of $k$-directional ray intersection graphs. Particularly interesting is the case where all rays are parallel to the $x$- or $y$-axis. The resulting class is the class of orthogonal ray graphs, which is the subject of this paper. A $k$-directional orthogonal ray graph, for short a $k$-DORG ($k \in \{2,3,4\}$), is an orthogonal ray graph with rays in $k$ directions. If $k = 2$ we assume that all rays point in the positive $x$- and the positive $y$-direction, if $k = 3$ we additionally allow the negative $x$-direction.

The class of 2-DORGs was introduced in $[19]$, where it is shown that the class of 2-DORGs coincides with the class of bipartite graphs whose complements are circular arc graphs, i.e. intersection graphs of arcs on a circle. This characterization implies the existence of a polynomial recognition algorithm (see $[13]$), as well as a characterization based on forbidden subgraphs $[5]$. Alternatively, 2-DORGs can also be characterized as the comparability graphs of ordered sets of height two and interval dimension two. This yields another polynomial recognition algorithm (see e.g. $[7]$), and due to the classification of 3-interval irreducible posets ($[6]$, $[21$, sec 3.7$]$) a complete description of minimally forbidden subgraphs. In a very nice recent contribution on 2-DORGs $[20]$, a clever solution has been presented for the jump number problem for the corresponding class of posets and shows a close connection between this problem and a hitting set problem for axis aligned rectangles in the plane.

4-DORGs in VLSI design. In $[18]$ 4-DORGs were introduced as a mathematical model for defective nano-crossbars in PLA (programmable logic arrays) design. A nano-crossbar is a rectangular circuit board with $m \times n$ orthogonally crossing wires. Fabrication defects may lead to disconnected wires. The bipartite intersection graph that models the surviving crossbar is an orthogonal ray graph.

We briefly mention two problems for 4-DORGs that are tackled in $[18]$. One of them is that of finding, in a nano-crossbar with disconnected wire defects, a maximal surviving square (perfect) crossbar, which translates into finding a maximal $k$ such that the balanced complete bipartite graph $K_{k,k}$ is a subgraph of the orthogonal ray graph modeling the crossbar. This balanced biclique problem is NP-complete for general bipartite graphs but turns out to be polynomially solvable on 4-DORGs $[18]$. The other problem, posed in $[16]$, asks how difficult it is to find a subgraph that would model a given logic mapping and is shown in $[18]$ to be NP-hard.

4-DORGs and UGIGs. A unit grid intersection graph (UGIG) is a GIG that admits an orthogonal segment representation with all segments of equal (unit)
length. Every 4-DORG is a GIG. This can be seen by intersecting the ray representation with a rectangle $R$, that contains all intersections between the rays in the interior. To see that every 4-DORG is a UGIG, we first fix an appropriate length for the segments, e.g. the length $d$ of the diagonal of $R$. If we only keep the initial part of length $d$ from each ray we get a UGIG representation. Essentially this construction was already used in [18].

Unit grid intersection graphs were considered in [15]. There it is shown that UGIG contains $P_6$-free bipartite graphs, interval bigraphs and bipartite permutation graphs. Actually, these classes are already contained in 2-DORG. Another contribution of [15] is to provide an example showing that the inclusion of UGIG in GIG is proper. In [17] it is shown that interval bigraphs belong to UGIG. Hardness of Hamiltonian cycle for inputs from UGIG and hardness of graph isomorphism for inputs from GIG have been shown in [22]. Very recently it was shown that the recognition of UGIGs is NP-complete [14]. With this last result we find 4-DORG nested between 2-DORG and UGIG with easy and hard recognition, respectively. This fact was central for our motivation to attack the open recognition problem for 4-DORGs [19].

**Our contribution.** In this paper we prove that, given a graph $G$ along with a partition $\{L, R, U, D\}$ of its vertices, it can be efficiently checked whether $G$ has a 4-DORG representation such that the vertices of $L$ (resp. the vertices of $R, U, D$) correspond to the rays pointing leftwards (resp. rightwards, upwards, downwards). To obtain our result, we first construct two cliques $G_1, G_2$ that have both directed and undirected edges. Then we iteratively augment $G_1$ and $G_2$, constructing eventually a graph $\tilde{G}$ with both directed and undirected edges. As we prove, the input graph $G$ has a 4-DORG representation if and only if $\tilde{G}$ has a transitive orientation respecting its directed edges. As a crucial tool for our results, we introduce the notion of an *S-orientation* of an arbitrary graph, which extends the notion of a transitive orientation. By setting $D = \emptyset$, our results trivially imply that, given a partition $\{L, R, U\}$ of the vertices of $G$, it can be efficiently checked whether $G$ has a 3-DORG representation according to this partition. With an independent approach, we show that if we are given a

---

**Fig. 1.** (a) A nano-wire crossbar with disconnected wire defects, (b) the bipartite graph modeling this crossbar, and (c) a 4-DORG representation of this graph. Note that vertex $t$ is not present, since the corresponding wire is not connected to the crossbar boundary, hence with the remaining circuit.
permutation \( \pi \) of the vertices of \( U \) (which represents the order of \( y \)-coordinates of ray-endpoints for the set \( U \)) but the partition \( \{L, R\} \) of \( V \setminus U \) is unknown, then we can still efficiently check whether \( G \) has a 3-DORG representation. The method we use to prove this result can be viewed as a particular partition refinement technique. Such techniques have various applications in string sorting, automaton minimization, and graph algorithms (see [8] for an overview).

**Notation.** We consider in this article simple undirected and directed graphs. For a graph \( G \), we denote its vertex and edge set by \( V(G) \) and \( E(G) \), respectively. In an undirected graph \( G \), the edge between vertices \( u \) and \( v \) is denoted by \( uv \), and in this case \( u \) and \( v \) are said to be adjacent in \( G \). The set \( N(v) = \{ u \in V : uv \in E \} \) is called the neighborhood of the vertex \( v \) of \( G \). If the graph \( G \) is directed, we denote by \( (uv) \) the oriented arc from \( u \) to \( v \). If \( G \) is the complete graph (i.e. a clique), we call an orientation \( \lambda \) of all (resp. of some) edges of \( G \) a (partial) tournament of \( G \). If in addition \( \lambda \) is transitive, then we call it a (partial) transitive tournament. Given two matrices \( A \) and \( B \) of size \( n \times n \) each, we call by \( O(MM(n)) \) the time needed by the fastest known algorithm for multiplying \( A \) and \( B \); currently this can be done in \( O(n^{2.376}) \) time [3].

Let \( G \) be a 4-DORG. Then, in a 4-DORG representation of \( G \), every ray is completely determined by one point on the plane and the direction of the ray. We call this point the endpoint of this ray. Given a 4-DORG \( G \) along with a 4-DORG representation of it, we may not distinguish in the following between a vertex of \( G \) and the corresponding ray in the representation, whenever it is clear from the context. Furthermore, for any vertex \( u \) of \( G \) we will denote by \( u_x \) and \( u_y \) the \( x \)-coordinate and the \( y \)-coordinate of the endpoint of the ray of \( u \) in the representation, respectively.

## 2 4-Directional Orthogonal Ray Graphs

In this section we investigate some fundamental properties of 4-DORGs and their representations, which will then be used for our recognition algorithm. The next observation on a 4-DORG representation is crucial for the rest of the section.

**Observation 1** Let \( G = (V, E) \) be a graph that admits a 4-DORG representation, in which \( L \) (resp. \( R, U, D \) ) is the set of leftwards (resp. rightwards, upwards, downwards) oriented rays. If \( u \in U \) and \( v \in R \) (resp. \( v \in L \) ), then \( uv \in E \) if and only if \( u_x > v_x \) (resp. \( u_x < v_x \) ) and \( u_y < v_y \). Similarly, if \( u \in D \) and \( v \in R \) (resp. \( v \in L \) ), then \( uv \in E \) if and only if \( u_x > v_x \) (resp. \( u_x < v_x \) ) and \( u_y > v_y \).

For the remainder of the section, let \( G = (V, E) \) be an arbitrary input graph with vertex partition \( V = L \cup R \cup U \cup D \), such that \( E \subseteq (L \cup R) \times (U \cup D) \).

**The oriented cliques \( G_1 \) and \( G_2 \).** In order to decide whether the input graph \( G = (V, E) \) admits a 4-DORG representation, in which \( L \) (resp. \( R, U, D \) ) is the set of leftwards (resp. rightwards, upwards, downwards) oriented rays, we first construct two auxiliary cliques \( G_1 \) and \( G_2 \) with \( |V| \) vertices each. We partition the vertices of \( G_1 \) (resp. \( G_2 \) ) into the sets \( L_x, R_x, U_x, D_x \) (resp. \( L_y, R_y, U_y, D_y \) ).
The intuition behind this notation for the vertices of $G_1$ and $G_2$ is that, if $G$ has a 4-DORG representation with respect to the partition $\{L, R, U, D\}$, then each of these vertices of $G_1$ (resp. $G_2$) corresponds to the $x$-coordinate (resp. $y$-coordinate) of the endpoint of a ray of $G$ in this representation.

We can now define some orientation of the edges of $G_1$ and $G_2$. The intuition behind these orientations comes from Observation 1: if the input graph $G$ is a 4-DORG, then it admits a 4-DORG representation such that, for every $u \in U \cup D$ and $v \in L \cup R$, we have that $u_x > v_x$ (resp. $u_y > v_y$) in this representation if and only if $\langle u_xv_x \rangle$ (resp. $\langle u_yv_y \rangle$) is an oriented edge of the clique $G_1$ (resp. $G_2$). That is, since all $x$-coordinates (resp. $y$-coordinates) of the endpoints of the rays in a 4-DORG representation can be linearly ordered, these orientations of the edges of $G_1$ (resp. $G_2$) build a transitive tournament.

Therefore, the input graph $G$ admits a 4-DORG representation if and only if some edges of $G_1, G_2$ are forced to have specific orientations in these transitive tournaments of $G_1$ and $G_2$, while some pairs of edges of $G_1, G_2$ are not allowed to have a specific pair of orientations in these tournaments. Motivated by this, we introduce in the next two definitions the notions of type-1-mandatory orientations and of forbidden pairs of orientations, which will be crucial for our analysis in the remainder of Section 2.

Definition 1 (type-1-mandatory orientations). Let $u \in U \cup D$ and $v \in L \cup R$, such that $uv \in E$. If $u \in U$ and $v \in R$ (resp. $v \in L$) then the orientations $\langle u_xv_y \rangle$ (resp. $\langle v_yu_x \rangle$) and $\langle v_yu_x \rangle$ of $G_1$ and $G_2$ are called type-1-mandatory. If $u \in D$ and $v \in R$ (resp. $v \in L$) then the orientations $\langle u_xv_y \rangle$ (resp. $\langle v_yu_x \rangle$) and $\langle u_yv_y \rangle$ of $G_1$ and $G_2$ are called type-1-mandatory. The set of all type-1-mandatory orientations of $G_1$ and $G_2$ is denoted by $M_1$.

Definition 2 (forbidden pairs of orientations). Let $u \in U \cup D$ and $v \in R \cup L$, such that $uv \notin E$. If $u \in U$ and $v \in R$ (resp. $v \in L$) then the pair $\{\langle u_xv_x \rangle, \langle v_yu_y \rangle\}$ (resp. the pair $\{\langle v_xu_x \rangle, \langle v_yu_y \rangle\}$) of orientations of $G_1$ and $G_2$ is called forbidden. If $u \in D$ and $v \in R$ (resp. $v \in L$) then the pair $\{\langle u_xv_x \rangle, \langle u_yv_y \rangle\}$ (resp. the pair $\{\langle v_xu_x \rangle, \langle u_yv_y \rangle\}$) of orientations of $G_1$ and $G_2$ is called forbidden.

For simplicity of notation in the remainder of the paper, we introduce in the next definition the notion of optional edges.

Definition 3 (optional edges). Let $\{(pq), (ab)\}$ be a pair of forbidden orientations of $G_1$ and $G_2$. Then each of the (undirected) edges $pq$ and $ab$ is called optional edges.

The augmented oriented cliques $G_1^*$ and $G_2^*$. We iteratively augment the cliques $G_1$ and $G_2$ into the two larger cliques $G_1^*$ and $G_2^*$, respectively, as follows. For every optional edge $pq$ of $G_1$ (resp. of $G_2$), where $p \in U_x \cup D_x$ and $q \in L_x \cup R_x$ (resp. $p \in U_y \cup D_y$ and $q \in L_y \cup R_y$), we add two vertices $r_{p,q}$ and $r_{q,p}$ and we add all needed edges to make the resulting graph $G_1^*$ (resp. $G_2^*$) a clique. Note that, if the initial graph $G$ has $n$ vertices and $m$ non-edges (i.e. $\binom{n}{2} - m$ edges),
then $G_1^*$ and $G_2^*$ are cliques with $n + 2m$ vertices each. We now introduce the notion of type-2-mandatory orientations of $G_1^*$ and $G_2^*$.

**Definition 4 (type-2-mandatory orientations).** For every optional edge $pq$ of $G_1^*$, the orientations $\langle pr_{p,q}\rangle$ and $\langle qr_{q,p}\rangle$ of $G_1^*$ are called type-2-mandatory orientations of $G_1^*$. For every optional edge $pq$ of $G_2^*$, the orientations $\langle rp_{p,q}\rangle$ and $\langle rq_{q,p}\rangle$ of $G_2^*$ are called type-2-mandatory orientations of $G_2^*$. The set of all type-2-mandatory orientations of $G_1^*$ and $G_2^*$ is denoted by $M_2$.

The coupling of $G_1^*$ and $G_2^*$ into the oriented clique $G^*$. Now we iteratively construct the clique $G^*$ from the cliques $G_1^*$ and $G_2^*$, as follows. Initially $G^*$ is the union of $G_1^*$ and $G_2^*$, together with all needed edges such that $G^*$ is a clique. Then, for every pair $\{pq, ab\}$ of forbidden orientations of $G_1^*$ and $G_2^*$ (where $pq \in E(G_1^*)$ and $ab \in E(G_2^*)$, cf. Definition 2), we merge in $G^*$ the vertices $rb_a$ and $rp_q$, i.e. we have $rb_a = rp_q$ in $G^*$. Recall that each of the cliques $G_1^*$ and $G_2^*$ has $n + 2m$ vertices. Therefore, since $G_1^*$ and $G_2^*$ have $m$ pairs $\{pq, ab\}$ of forbidden orientations, the resulting clique $G^*$ has $2n + 3m$ vertices. We now introduce the notion of type-3-mandatory orientations of $G^*$.

**Definition 5 (type-3-mandatory orientations).** For every pair $\{pq, ab\}$ of forbidden orientations of $G_1^*$ and $G_2^*$, the orientation $\langle r_{a,p}r_{a,b}\rangle$ is called a type-3-mandatory orientation of $G^*$. The set of all type-3-mandatory orientations of $G^*$ is denoted by $M_3$.

Whenever the orientation of an edge $uv$ of $G^*$ is type-1 (resp. type-2, type-3)-mandatory, we may say for simplicity that the edge $uv$ (instead of its orientation) is type-1 (resp. type-2, type-3)-mandatory. An example for the construction of $G^*$ from $G_1^*$ and $G_2^*$ is illustrated in Figure 2, where it is shown how two optional edges $pq \in E(G_1^*)$ and $ab \in E(G_2^*)$ are joined together in $G^*$, where $\{pq, ab\}$ is a pair of forbidden orientations of $G_1^*$ and $G_2^*$. For simplicity of the presentation, only the optional edges $pq$ and $ab$, the type-2-mandatory edges $pr_{p,q}, qr_{q,p}, ar_{a,b}, br_{b,a}$, and the edges $r_{p,q}r_{a,b}$ and $r_{q,p}r_{b,a}$ are shown in Figure 2. Furthermore, the type-2-mandatory orientations $\langle pr_{p,q}\rangle$, $\langle qr_{q,p}\rangle$, $\langle ar_{a,b}\rangle$, and $\langle br_{b,a}\rangle$, as well as the type-3-mandatory orientation $\langle r_{q,p}r_{a,b}\rangle$, are drawn with double arrows in Figure 2 for better visibility.

![Fig. 2.](image-url) An example of joining in $G^*$ the pair of optional edges $\{pq, ab\}$, where $pq \in E(G_1^*)$ and $ab \in E(G_2^*)$.

In the next theorem we provide a characterization of 4-DORGs in terms of a transitive tournament $\lambda^*$ of the clique $G^*$. The main novelty of the characterization-
tion of Theorem 1 is that it does not rely on the forbidden pairs of orientations. This characterization will be used in Section 4, in order to provide our main result of the paper, namely the recognition of 4-DORGs with respect to the vertex partition \( \{L, R, U, D\} \).

**Theorem 1.** The next two conditions are equivalent:

1. The graph \( G = (V, E) \) with \( n \) vertices has a 4-DORG representation with respect to the vertex partition \( \{L, R, U, D\} \).
2. There exists a transitive tournament \( \lambda^* \) of \( G^* \), such that \( M_1 \cup M_2 \cup M_3 \subseteq \lambda^* \), and in addition:
   
   \( a \) let \( pq \) be an optional edge of \( G_1 \) and \( pw \notin M_2 \) be an incident edge of \( pq \) in \( G_1^* \); then \( \langle wp, p \rangle \in \lambda^* \) implies that \( \langle wp \rangle \in \lambda^* \),
   
   \( b \) let \( pq \) be an optional edge of \( G_2 \) and \( pw \notin M_2 \) be an incident edge of \( pq \) in \( G_2^* \); then \( \langle r, pw \rangle \in \lambda^* \) implies that \( \langle pw \rangle \in \lambda^* \),
   
   \( c \) let \( pq \) be an optional edge of \( G_1^* \) (resp. \( G_2^* \)), where \( p \in U_x \cup D_y \) (resp. \( p \in U_y \cup D_x \)); then we have:
   
   \( i \) either \( \langle pq \rangle, \langle r, pq \rangle, \langle r, pq, r \rangle \in \lambda^* \) or \( \langle qp \rangle, \langle qr, p \rangle, \langle qr, r, p \rangle \in \lambda^* \),
   
   \( ii \) for any incident optional edge \( pq' \) of \( G_1^* \) (resp. \( G_2^* \)), either \( \langle pq \rangle, \langle r, pq, r \rangle, \langle pq, r \rangle \in \lambda^* \) or \( \langle pq, r \rangle, \langle qr, p \rangle, \langle qr, r, p \rangle \in \lambda^* \),
   
   \( iii \) for any incident optional edge \( pq' \) of \( G_1^* \) (resp. \( G_2^* \)), either \( \langle pq \rangle, \langle r, pq, r \rangle, \langle pq, r \rangle \in \lambda^* \) or \( \langle pq, r \rangle, \langle qr, p \rangle, \langle qr, r, p \rangle \in \lambda^* \).

Furthermore, as we can prove, given a transitive tournament \( \lambda^* \) of \( G^* \) as in Theorem 1, a 4-DORG representation of \( G \) can be computed in \( O(n^2) \) time. An example of the orientations of condition 2(c) in Theorem 1 (for the case of \( G_1^* \)) is shown in Figure 3. For simplicity of the presentation, although \( G_1^* \) is a clique, we show in Figure 3 only the edges that are needed to illustrate Theorem 1.

## 3 S-orientations of graphs

In this section we introduce a new way of augmenting an arbitrary graph \( G \) by adding a new vertex and some new edges to \( G \). This type of augmentation process is done with respect to a particular edge \( e_i = x_iy_i \) of the graph \( G \), and is called the deactivation of \( e_i \) in \( G \). In order to do so, we first introduce the crucial notion of an S-orientation of a graph \( G \) (cf. Definition 7), which extends the classical notion of a transitive orientation. For the remainder of this section, \( G \) denotes an arbitrary graph, and not the input graph discussed in Section 2.

**Definition 6.** Let \( G = (V, E) \) be a graph and let \((x_i, y_i), 1 \leq i \leq k, k \) ordered pairs of vertices of \( G \), where \( x_iy_i \in E \). Let \( V_{out}, V_{in} \) be two disjoint vertex subsets of \( G \), where \( \{x_i : 1 \leq i \leq k\} \subseteq V_{out} \cup V_{in} \). For every \( i = 1, 2, \ldots, k \):

- a special neighborhood of \( x_i \) is a vertex subset \( S(x_i) \subseteq \left( N(x_i) \cap \left( \cap_{j \neq i} N(y_j) \right) \right) \setminus \{x_j : 1 \leq j \leq k\} \),
- the forced neighborhood orientation of \( x_i \) is:
  - the set \( F(x_i) = \{ (x_i, z) : z \in S(x_i) \} \) of oriented edges of \( G \), if \( x_i \in V_{out}, \)
  - the set \( F(x_i) = \{ (zx_i) : z \in S(x_i) \} \) of oriented edges of \( G \), if \( x_i \in V_{in} \).
Fig. 3. An example of the orientations of the clique $G_i^1$ in the transitive tournament $\lambda^*$, where $p \in U_i \cup D_i$ (cf. condition 2(c) in Theorem 1): (a) both possible orientations where the optional edges $pq$ and $p'q'$ are incident and (b) both possible orientations where the optional edges $pq$ and $p'/q'$ are incident. In both (a) and (b), the orientations of the type-2-mandatory edges are drawn with double arrows. The case for $G_2$ is the same, except that the orientation of the type-2-mandatory edges is the opposite.

**Definition 7.** Let $G = (V, E)$ be a graph. For every $i = 1, 2, \ldots, k$ let $S(x_i)$ be a special neighborhood in $G$. Let $T$ be a transitive orientation of $G$. Then $T$ is an $S$-orientation of $G$ on the special neighborhoods $S(x_i)$, $1 \leq i \leq k$, if for every $i = 1, 2, \ldots, k$:  
1. $F(x_i) \subseteq T$ and  
2. for every $z \in S(x_i)$, $\langle x_i, y_i \rangle \in T$ if and only if $\langle z, y_i \rangle \in T$.

**Definition 8.** Let $G = (V, E)$ be a graph. For every $i = 1, 2, \ldots, k$ let $S(x_i)$ be a special neighborhood in $G$. Let $T$ be an $S$-orientation of $G$ on the sets $S(x_i)$, $1 \leq i \leq k$. Then $T$ is consistent if, for every $i = 1, 2, \ldots, k$, it satisfies the following conditions, whenever $zw \in E$, where $z \in S(x_i)$ and $w \in (N(x_i) \cap N(y_i)) \setminus S(x_i)$:  
- if $x_i \in V_{out}$, then $\langle wz \rangle \in T$ implies that $\langle wx_i \rangle \in T$,  
- if $x_i \in V_{in}$, then $\langle wz \rangle \in T$ implies that $\langle x_i, w \rangle \in T$.

In the next definition we introduce the notion of deactivating an edge $e_i = x_iy_i$ of a graph $G$, where $S(x_i)$ is a special neighborhood in $G$. In order to deactivate edge $e_i$ of $G$, we augment appropriately the graph $G$, obtaining a new graph $\tilde{G}(e_i)$ that has one new vertex.

**Definition 9.** Let $G = (V, E)$ be a graph and let $S(x_i)$ be a special neighborhood in $G$. The graph $\tilde{G}(e_i)$ obtained by deactivating the edge $e_i = x_iy_i$ (with respect to $S_i$) is defined as follows:  
1. $V(\tilde{G}(e_i)) = V \cup \{a_i\}$ (i.e. add a new vertex $a_i$ to $G$),  
2. $E(\tilde{G}(e_i)) = E \cup \{za_i : z \in N(x_i) \setminus S(x_i)\}$.
Algorithm 1 Recognition of 4-DORGs

**Input:** An undirected graph \( G = (V, E) \) with a vertex partition \( V = L \cup R \cup U \cup D \)

**Output:** A 4-DORG representation for \( G \), or the announcement that \( G \) is not a 4-DORG graph

1: \( n \leftarrow |V|; \quad m \leftarrow \binom{n}{2} - |E| \{m \text{ is the number of non-edges in } G\}
2: Construct from \( G \) the clique \( G_1 \) with vertex set \( L_x \cup R_x \cup U_x \cup D_x \) and the clique \( G_2 \) with vertex set \( R_y \cup U_y \cup D_y \)
3: Construct the set \( M_1 \) of type-1-mandatory orientations in \( G_1 \) and \( G_2 \)
4: Construct the \( m \) forbidden pairs of orientations of \( G_1 \) and \( G_2 \)
5: Construct from \( G_1, G_2 \) the augmented cliques \( G_1^*, G_2^* \) and the set \( M_2 \) of type-2-mandatory orientations
6: Construct from \( G_1^*, G_2^* \) the clique \( G^* \) and the set \( M_3 \) of type-3-mandatory orientations
7: for \( i = 1 \) to \( m \) do
8: Let \( p_i, q_i \in E(G_1) \), \( a_i, b_i \in E(G_2) \) be the optional edges in the \( i \)th pair of forbidden orientations, where \( p_i \in U_x \cup D_x, q_i \in L_x \cup R_x, a_i \in U_y \cup D_y, b_i \in L_y \cup R_y \)
9: \( (x_{2i-1}, y_{2i-1}) \leftarrow (p_i, q_i); \quad (x_{2i}, y_{2i}) \leftarrow (q_i, r_{p_i, q_i}) \)
10: \( (x_{2m+2i-1}, y_{2m+2i-1}) \leftarrow (a_i, b_i); \quad (x_{2m+2i}, y_{2m+2i}) \leftarrow (b_i, r_{a_i, b_i}) \)
11: \( S(x_i) \leftarrow \{r_{x_i, y_i} : x_j = x_i\} \)
12: Construct the graph \( \tilde{G}^* \) by iteratively deactivating all edges \( x_i y_i, 1 \leq i \leq 4m \)
13: if \( \tilde{G}^* \) has a transitive orientation \( \tilde{T} \) such that \( M_1 \cup M_2 \cup M_3 \subseteq \tilde{T} \) then
14: return the 4-DORG representation of \( G \) computed by Theorem 1
15: else
16: return “G is not a 4-DORG graph with respect to the partition \{L, R, U, D\}”

After deactivating the edge \( e_k \) of \( G \), obtaining the graph \( \tilde{G}(e_k) \), we can continue by sequentially deactivating the edges \( e_{k-1}, e_{k-2}, \ldots, e_1 \), obtaining eventually the graph \( \tilde{G} \).

**Theorem 2.** Let \( G = (V, E) \) be a graph and \( S(x_i), 1 \leq i \leq k \), be a set of \( k \) special neighborhoods in \( G \). Let \( M_0 \) be an arbitrary set of edge orientations of \( G \), and let \( \tilde{G} \) be the graph obtained after deactivating all edges \( e_i = x_i y_i \), where \( 1 \leq i \leq k \).

- If \( G \) has a consistent \( S \)-orientation \( T \) on \( S(x_1), S(x_2), \ldots, S(x_k) \) such that \( M_0 \subseteq T \), then \( \tilde{G} \) has a transitive orientation \( \tilde{T} \) such that \( M_0 \cup F(x_i) \subseteq \tilde{T} \) for every \( i = 1, 2, \ldots, k \).
- If \( G \) has a transitive orientation \( \tilde{T} \) such that \( M_0 \cup F(x_i) \subseteq \tilde{T} \) for every \( i = 1, 2, \ldots, k \), then \( G \) has an \( S \)-orientation \( T \) on \( S(x_1), S(x_2), \ldots, S(x_k) \) such that \( M_0 \subseteq T \).

4 Efficient Recognition of 4-DORGs

In this section we complete our analysis in Sections 2 and 3 and we present our 4-DORG recognition algorithm (cf. Algorithm 1). Let \( G = (V, E) \) be an arbitrary
input graph that is given along with a vertex partition \( V = L \cup R \cup U \cup D \), such that \( E \subseteq (L \cup R) \times (U \cup D) \). Assume that \( G \) has \( n \) vertices and \( m \) non-edges (i.e. \( \binom{n}{2} - m \) edges). First we construct from \( G \) the cliques \( G_1, G_2 \), then we construct the augmented cliques \( G_1^*, G_2^* \), and finally we combine \( G_1^* \) and \( G_2^* \) to produce the clique \( G^* \) (cf. Section 2). Then, for a specific choice of \( 4m \) ordered pairs \((x_i, y_i)\) of vertices, where \( 1 \leq i \leq 4m \) (cf. Algorithm 1), and for particular sets \( S(x_i) \) and neighborhood orientations \( F(x_i) \), \( 1 \leq i \leq 4m \) (cf. Definitions 6 and 7), we iteratively deactivate the edges \( x_i y_i \), \( 1 \leq i \leq 4m \) (cf. Section 3), constructing thus the graph \( \tilde{G} \). Then, we can prove that for a specific partial orientation of the graph \( \tilde{G} \), \( \tilde{G} \) has a transitive orientation that extends this partial orientation if and only if the input graph \( G \) has a 4-DORG representation with respect to the vertex partition \( \{L, R, U, D\} \). The proof of correctness of Algorithm 1 and the timing analysis are given in the next theorem.

**Theorem 3.** Let \( G = (V, E) \) be a graph with \( n \) vertices, given along with a vertex partition \( V = L \cup R \cup U \cup D \), such that \( E \subseteq (L \cup R) \times (U \cup D) \). Then Algorithm 1 constructs in \( O(\text{MM}(n^2)) \) time a 4-DORG representation for \( G \) with respect to this vertex partition, or correctly announces that \( G \) does not have a 4-DORG representation.

### 5 Recognizing 3-DORGs with partial representation restrictions

In this section we consider a bipartite graph \( G = (A, B, E) \), where \( |A| = m \) and \( |B| = n \), given along with an ordering \( \pi = (v_1, v_2, \ldots, v_m) \) of the vertices of \( A \).

The question we address is the following: “Does \( G \) admit a 3-DORG representation where \( A \) (resp. \( B \)) is the set of rays oriented upwards (resp. horizontal, i.e. either leftwards or rightwards), such that, whenever \( 1 \leq i < j \leq m \), the \( y \)-coordinate of the endpoint of \( v_i \in A \) is greater than that of \( v_j \in A \)?” Our approach uses the adjacency relations in \( G \) to recursively construct an \( x \)-coordinate ordering of the endpoints of the rays in the set \( A \). If during the process we do not reach a contradiction, we eventually construct a 3-DORG representation for \( G \), otherwise we conclude that such a representation does not exist.

**Definition 10.** Let \( P_1, P_2 \) be two ordered partitions of the same base set \( S \). Then \( P_1 \) and \( P_2 \) are compatible if there exists an ordered partition \( R \) of \( S \) which is refining and order preserving for both \( P_1 \) and \( P_2 \). A linear order \( L \) respects an ordered partition \( P \) of \( S \), if \( L \) and \( P \) are compatible.

Here we provide the main ideas and an overview of our algorithm. We start with the trivial partition of the set \( A \) (consisting of a single set including all elements of \( A \)). During the algorithm we process each vertex of \( V = A \cup B \) once, and each time we process a new vertex we refine the current partition of the vertices of \( A \), where the final partition of \( A \) implies an \( x \)-coordinate ordering of the rays of \( A \). In particular, the algorithm proceeds in \( |A| = m \) phases, where during phase \( i \) we process vertex \( v_i \in A \) (the sequence of the vertices in \( A \) is
Fig. 4. Construction of a 3-DORG representation. Top left of the figure: the bipartite graph $G$ with the given vertex ordering $\pi = (v_1, v_2, v_3, v_4, v_5)$. Top-right: the chain of partition refinements. Bottom left: The 3-DORG representation of $G$ as read from the partition chain.

According to the given ordering $\pi$). During phase $i$, we process sequentially every neighbor $u \in N(v_i) \subseteq B$ that has not been processed in any previous phase $j < i$. For every $i = 1, 2, \ldots, m$ let $A_i = \{v_i, v_{i+1}, \ldots, v_m\}$ be the set of vertices of $A$ that have not been processed before phase $i$. At the end of every phase $i$, we fix the position of vertex $v_i \in A$ in the final partition of $A$, and weignore $v_i$ in the subsequent phases (i.e. during the phases $j > i$ we consider only the restriction of the current partition to the vertices of $A_{i+1}$). Phase $i$ starts with the partition of $A_i$ that results at the end of phase $i-1$. For any vertex $u \in N(v_i)$ that we process during phase $i$, we check whether the current partition $P$ of $A_i$ is compatible with at least one of the ordered partitions $Q_1 = (N(u), A_i \setminus N(u))$ and $Q_2 = (A_i \setminus N(u), N(u))$. If not, then we conclude that $G$ is not a 3-DORG with respect to the given ordering $\pi$ of $A$. Otherwise we refine the current partition $P$ into an ordered partition that is also a refinement of $Q_1$ (resp. $Q_2$). In the case where $P$ is compatible with both $Q_1$ and $Q_2$, it is irrelevant if we compute a common refinement of $P$ with $Q_1$ or $Q_2$. If we can execute all $m$ phases of this algorithm without returning that a 3-DORG representation does not exist, then we can compute a 3-DORG representation of $G$ in which the $y$-coordinates of the endpoints of the rays of $A$ respect the ordering $\pi$. In this extended abstract, this construction is illustrated in the example of Figure 4.

**Theorem 4.** Given a bipartite graph $G = (V, E)$ with color classes $A, B$ and an ordering $\pi$ of $A$, we can decide in $O(|V|^2)$ time whether $G$ admits a 3-DORG representation where $A$ are the vertical rays and the $y$-coordinates of their endpoints respect the ordering $\pi$. 

11
References