Chapter 8

Dominance Drawings

8.1 Numbering of digraphs and st-graphs.

A numbering of a directed acyclic planar graph G is an assignment of numbers to the vertices of G, such that for every edge (u, v) of G, the number assigned to v is greater than the one assigned to u (i.e. $num(v) > num(u)$).As you can notice it is not necessary that vertex is assigned a distinct number.For an example see Figure 8.1. Of course it is possible to assign a distinct number to

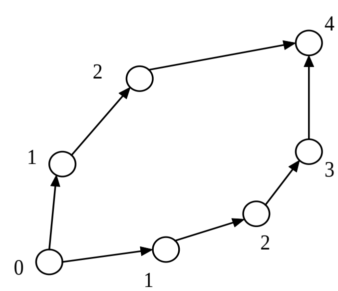

Figure 8.1: A numbering of a directed acyclic planar graph.

each vertex in the graph.Thus it is evident that every acyclic graph admits a numbering of the vertices it contains.

An st-graph is an acyclic directed graph with a single source s and a single sink t as shown in Figure 8.2. Since a planar st-graph is acyclic, it admits a numbering of its vertices.

8.2 Introduction to donimance drawings.

Here we study a drawing of planar st-graph called dominance drawing. A dominance drawing of a directed graph G is a representation of G such that for any two vertices u, v there exists a path from u to v in G if and only if the x-coordinate of u is less than or equal to the x-coordinate of v and y-coordinate

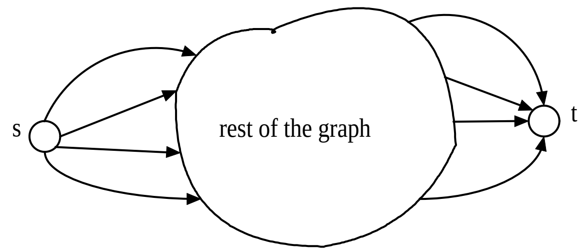

Figure 8.2: A general representation of an st-graph.

of u is less than or equal to the y-coordinate of v. (i.e. $x(u) \leq x(v) \wedge y(u) \leq y(v)$).(See Figure 8.3). It is evident that the vertices, which v "dominates" be-

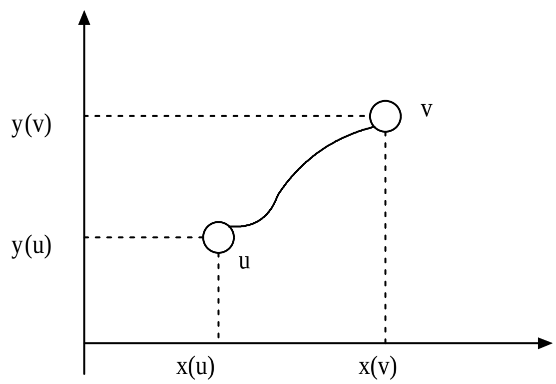

Figure 8.3: Vertex u and vertex v in a dominance drawing.

long to the third quadrant if we think a coordinate system starting on vertex v.(See Figure 8.4). All vertices that have a path towards v belong to the third quadrant. Inversely, all vertices that belong to the third quadrant have a path towards v(see Figure 8.4).

8.3 Dominance drawings of reduced digraphs.

A reduced digraph of a directed graph G (also called transitive reduction of G) is a representation of G with no transitive edges(see Figure 8.5). For example, e_1 in Figure 8.5 is a transitive edge and is eliminated in the reduced graph.

Lemma 8.3.1 *Any straight-line dominance drawing Γ of a reduced planar st-graph G is planar.*

Proof : We prove this lemma by contradiction.Therefore, to prove that G is planar, that is there is an embedding on the plane with no crossing edges,

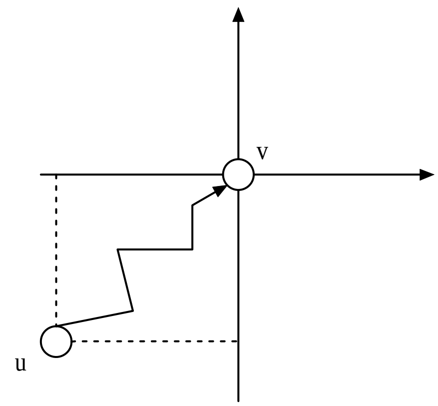

Figure 8.4: All vertices u that belong to the third quadrant have a path to v.

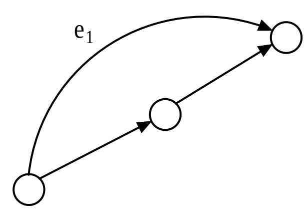

Figure 8.5: Transitive edge.

suppose there is a crossing between edges (u, v) and (w, z).Since in G there is not any transitive edge no vertex p is placed in the rectangle defined by points (x_u, y_u) and (x_v, y_v) of the drawing.Because if there was a vertex p in the aforementioned rectangle, there would be paths from $u \to p$ and $p \to v$ thus (u, v) would be a transitive edge. Thus, vertices w, z cannot be inside this rectangle. With a similar argument w cannot be in the bottom region of u, and z cannot be in the top region of v. Hence, the only possible case, taking also into account that Γ is a dominance drawing (see definition), is that $w \in leftregion(u) - leftregion(v)$ and $z \in rightregion(v) - rightregion(u)$.(See Figure 8.6). Figure 8.7 represents a particular instance of G. If we merge vertices $s - t - t'$ into vertex t' the resulting graph is isomorphic to $K_{3,3}$ thus not planar.Another way to explain, is that the graph in Figure 8.8 (without any merge) is homeomorphic to $K_{3,3}$(see Figure 8.8). □

Next we present the algorithm that gives the dominance drawing of a reduced graph G. The algorithm consists of three phases, Preprocessing, Preliminary Layout and Compaction.The Preprocessing phase sets up a linked data structure for G where each vertex v has a pointer to a list of its outgoing edges stored

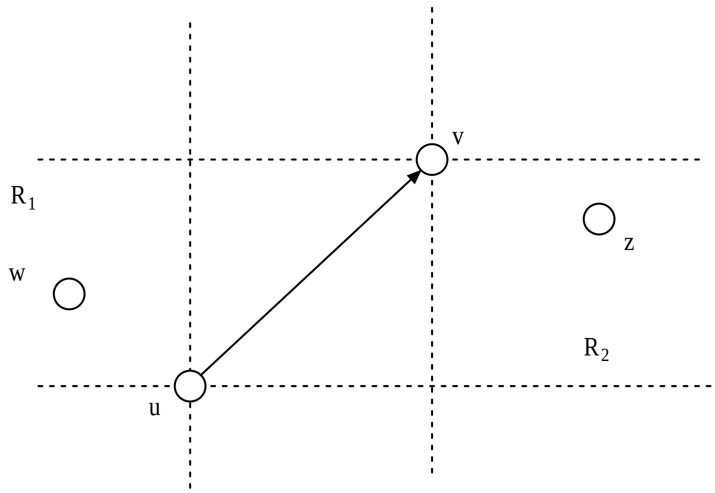

Figure 8.6: The figure shows where w and z can be placed.

according to their clockwise sequence around v(see Figure 8.9 and Figure 8.10). This list is doubly connected by means of pointers[1] $next[e]$ and $prev[e]$ and is accessed by means of pointers $firstout[v]$ and $lastout[v]$ to its leftmost and rightmost edges, respectively.Also v has pointers $firstin[v]$, $lastin[v]$ to its leftmost and rightmost incoming edges respectively(see Figure 8.11).

The preliminary layout assigns to each vertex v a distinct X- and Y-coordinate (note that we use capital X and Y for reasons that will be shortly clarified). Essentially it performs two sortings of the vertices of G, by first scanning the successors of each vertex from left to right (e.g. clockwise) and second from right to left (e.g. counterclockwise).

Algorithm 8.3.1: *Dominance-Straight-Line*
Input: reduced planar st-graph G
Output: straight-line dominance drawing Γ of G

```
DominanceStraightLine(G)
   Preprocessing
   PreliminaryLayout
   Compaction
```

```
PreliminaryLayout( )
   count ← 0
   LabelX(s)
   count ← 0
```

[1]We assume that v is a struct that has various pointers, such as fistout.Also e is a struct with pointers such as $next$, $prev$.

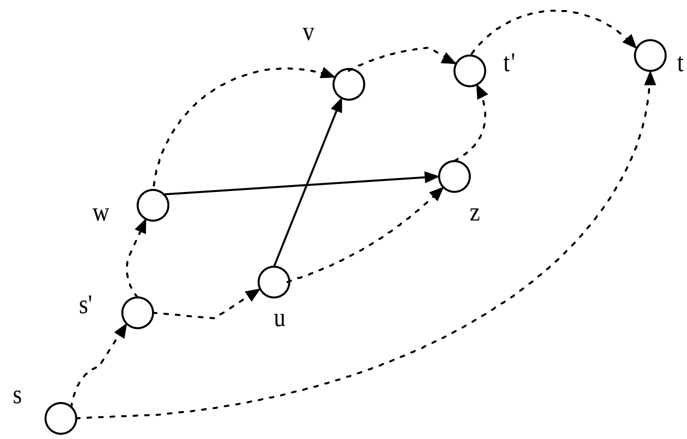

Figure 8.7: The drawing should not have crossings.

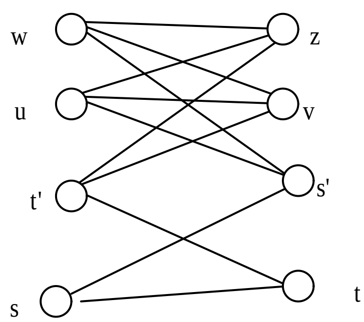

Figure 8.8: Homeomorphic to $K_{3,3}$.

```
LabelY(s)

LabelX(v)
X[v] ← count
count ← count + 1
if v ≠ t
   then e ← firstout[v]
        repeat
              w ← head[e]
              if e = lastin[w]
                 then LabelX(w)

              e ← next[e]
          until e = NIL
```

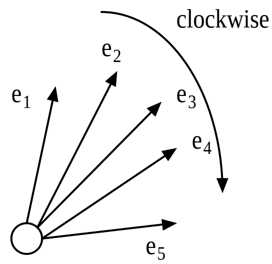

Figure 8.9: The outgoing edges.

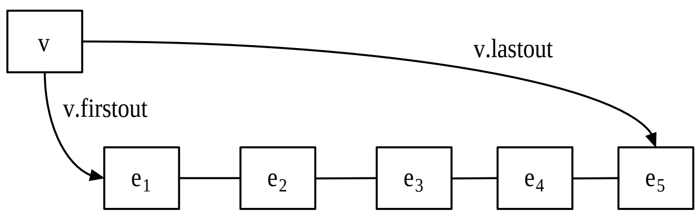

Figure 8.10: A possible data structure.

In a similar way we assign to each vertex v a distinct Y-coordinate. In essence we now scan the successors of each vertex in a counterclockwise fashion.

```
LABELY(v)
Y[v] ← count
count ← count + 1
if v ≠ t
   then e ← lastout[v]
        repeat
              w ← head[e]
              if e = firstin[w]
                 then LABELY(w)

              e ← prev[e]
          until e = NIL
```

For a run of the above procedures see Figure 8.12.

The compaction phase scans the verices according to the order given by the preliminary X- and Y-coordinates. In this phase we set up two lists of vertices sorted by increasing X- and Y-coordinate.Each vertex has a pointer to its next ($v.nextX$) in the horizontal axis and a pointer to its next ($v.nextY$) in the vertical axis (see Figure 8.13).

```
COMPACTIONX( )
let u be the vertex with X[u] = 0
```

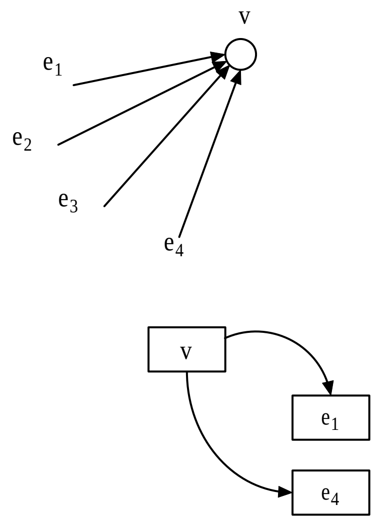

Figure 8.11: The incoming edges.

```
x[u] ← 0
while nextX[u] ≠ NIL
    do v ← nextX(u)
        if Y[u] > Y[v] or (firstout[u] = lastout[u] and firstin[v] = lastin[v])
          then x[v] ← x[u] + 1
          else  x[v] ← x[u]

        u ← v
```

Similarly for the y coordinates[2].

COMPACTIONY( )

```
let u be the vertex with Y[u] = 0
y[u] ← 0
while nextX[u] ≠ NIL
    do v ← nextY(u)
        if X[u] > X[v] or (firstout[u] = lastout[u] and firstin[v] = lastin[v])
          then y[v] ← y[u] + 1
          else  y[v] ← y[u]

        u ← v
```

In the compaction phase we scan the lists of vertices with concecutive (preliminary) X- and Y-coordinates. First, the list of vertices with concecutive X-coordinates and then the list of the y-axis (see Figure 8.13).The final x-coordinate of a vertex v (see algorithm) is not incremented by one if (u, v) is

[2]With capital X and Y we present the preliminary coordinates of the vertices and with lower case x, y the final coordinates after the compaction phase.

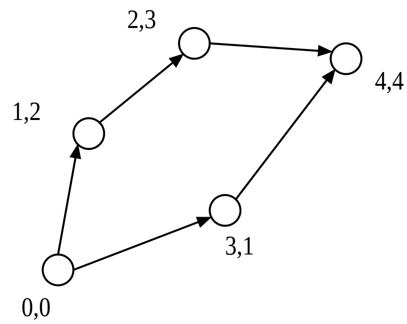

Figure 8.12: The coordinates after a run of the Preliminary Layout phase.

an edge. Therefore the v vertex is going to have the same x-coordinate with u (see algorithm) and appears on top of u (that is on the same vertical line). However, in the special case when (u, v) is the only outgoing edge of u and the only incoming edge of v, the x-coordinate is incremented by one.This is done to prevent the undesired effect of the coincidence of the coordinates of two vertices.

In the following we give some definitions: Let G be a graph and u a vertex of this graph.

- $B(u)$: is the set of vertices that can reach u by a directed path.
- $T(u)$: is the set of vertices that can be reached from u by a directed path.
- $L(u)$: is the set of vertices that are on the left of every path from s to t via u.
- $R(u)$: is the set of vertices that are on the right of every path from s to t via u.

Note that the singleton $\{u\}$ and the sets $B(u)$, $T(u)$, $L(u)$, $R(u)$ constitute a partition of V the set of vertices of G (see Figure 8.14).

Lemma 8.3.2 *The X- and Y-coordinates computed in the Preliminary Layout phase have the following properties:*

1. $X(u) < X(v) \Longleftrightarrow u \in B(v) \cup L(v)$.
2. $Y(u) < Y(v) \Longleftrightarrow u \in B(v) \cup R(v)$.

From the above Lemma we can deduce that $u \in B(u)$ if and only if both $X(u) \leq X(v)$ and $Y(u) \leq Y(v)$.Thus, we have (according to the definition of dominance drawing):

Theorem 8.3.1 *The drawing of G described by the X- and Y-coordinates computed in the Preliminary Layout phase is a dominance drawing.*

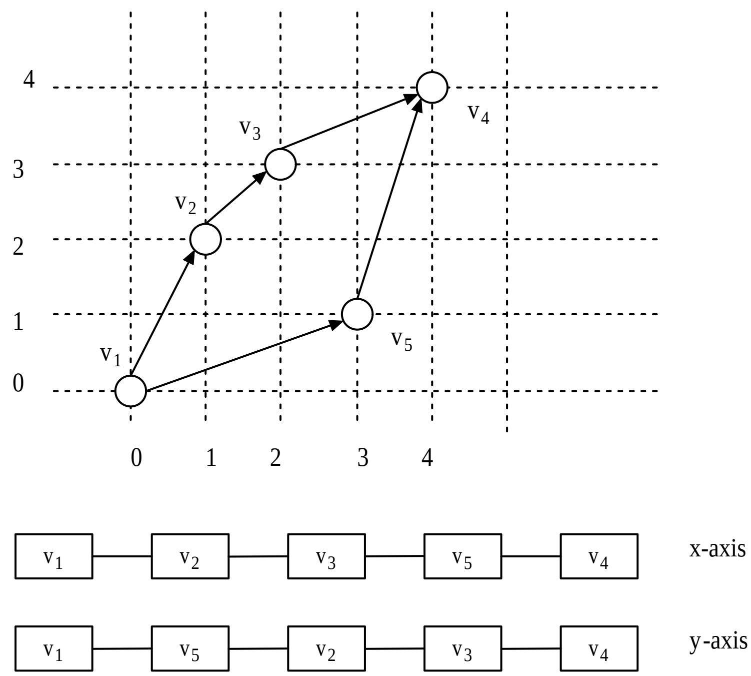

Figure 8.13: The output of the algorithm and the lists for the Compaction phase.

8.4 Exposure of symmetries.

Along with producing a dominance drawing, Algorithm Dominance-Straight-Line has the important feature of exposing the symmetries (if any) and isomorphic parts of the graph.

A directed graph G is weakly connected if its underlying undirected graph (constructed by forgetting the directions of the edges) is connected (see Figure 8.15). Let G be a planar st-graph.An open component of G is a maximal weakly-connected subgraph G' of G by removing a separating pair (p, q) of vertices, such that G' does not contain s or t. (see Figure 8.16). A closed component of G is a subgraph G' such that:

1. G' is a planar pq-graph.
2. G' contains every vertex of G that is on some path from p to q.
3. G' contains every outgoing edge of p, every incoming edge of q and every incident edge of the remaining vertices of G'.

The directed graph obtained from a closed component by removing its source and sink is not necessarily an open component, but in general the union of several open components(see Figure 8.17 and Figure 8.18).

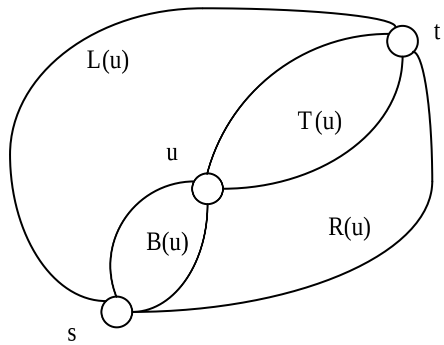

Figure 8.14: The regions of vertex u.

Figure 8.15: (left) weakly connected, (right) obviously connected.

Let C_1 and C_2 be two components of a planar st-graph G that are isomorphic if we ignore the directions of the edges. C_1 and C_2 are said to be *simply isomorphic* if the isomorphism preserves the direction of the edges and the embedding of the two components(see Figure 8.19). C_1 and C_2 are said to be *axially isomorphic* if the isomorphism preserves the directions of the edges and the two components are symmetric with respect to an axis.(see Figure 8.20). C_1 and C_2 are said to be *rotationally isomorphic* if the isomorphism inverts the directions of the edges and preserves the embedding of the two components.(see Figure 8.21). Because if we rotate C_2 by 180° the two graphs coincide. (if we ignore the directions of the edges).

Theorem 8.4.1 *Let G be a reduced planar st-graph, and Γ be the drawing constructed by Algorithm Dominance-Straight-Line. We have:*

1. *Simply isomorphic components of G have drawings in Γ that are congruent (similar) up to a translation.*

2. *Axially isomorphic components of G have drawings in Γ that are congruent up to a translation and reflection.*

3. *Rotationally isomorphic components of G have drawings in Γ that are congruent up to a translation* 180° *rotation.*

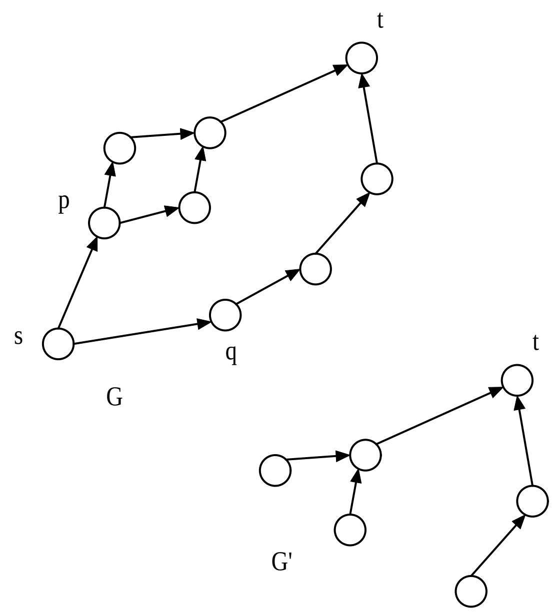

Figure 8.16: An open component G' of G.

4. *The drawing of an axially symmetric component of G is symmetric with respect to the straight line that passes through its source and sink.*

5. *The drawing of a rotationally symmetric component of G is symmetric with respect to a $180°$ rotation around its centroid (that is center of mass).*

Proof : In the Preprocessing phase, the vertices of a component are visited consecutively by procedures LabelX and LabelY.Hence, the layout of a component is is independent from the rest of the digraph.This proves Property 1.As regards Property 2 and 4, reversing the orientation of the faces exchanges the set $L(u)$ with $R(u)$ for every vertex u.This corresponds to exchanging the X-coordinate with the Y-coordinate, and similarly for the final x- and y-coordinates.This yields drawings that are congruent up to a translation and a reflection with respect to a $45°$ slope line.Now we consider Properties 3 and 5.Reversing the direction of the edges exchanges $B(u)$ with $T(u)$ and $L(u)$ with $R(u)$ for every vertex u. Hence, the X-lists of two rotationally isomorphic components are the reverse of one another and similarly for the Y-list.This implies that Properties 3 and 5 hold for the preliminary layout.The set E_L, E_H and E_R stay the same after reversing the direction of the edges.Thus the final x- and y-coordinates are incremented for the same pairs of vertices and Properties 3 and 5 hold for the final layout. □

Theorem 8.4.2 *Algorithm Dominance-Straight-Line constructs in $O(n)$ time a planar dominance drawing Γ of G with $O(n^2)$ area.*

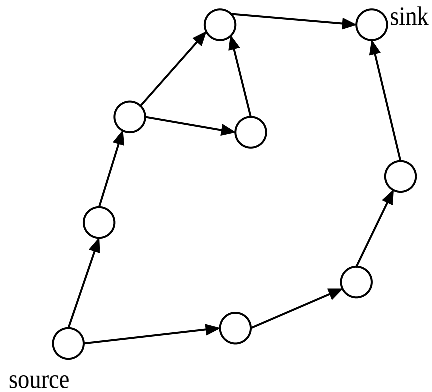

Figure 8.17: An st-graph.

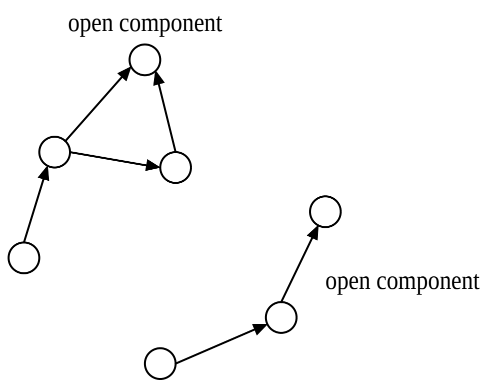

Figure 8.18: Two open components.

8.5 Dominance Drawings of st-graphs with transitive edges.

Up to now we have assumed that the graphs we work with have no transitive edges (reduced).We extend the idea to general st-graphs by inserting a dummy vertex on every transitive edge and then we follow the same procedure.In the resulting drawing we replace every dummy vertex with a bend.The number of bends is equal to the number of transitive edges in the input graph.

Algorithm: *Dominance-Polyline*
Input: planar st-graph G (may have transitive edges).
Output: polyline dominance drawing Γ of G.

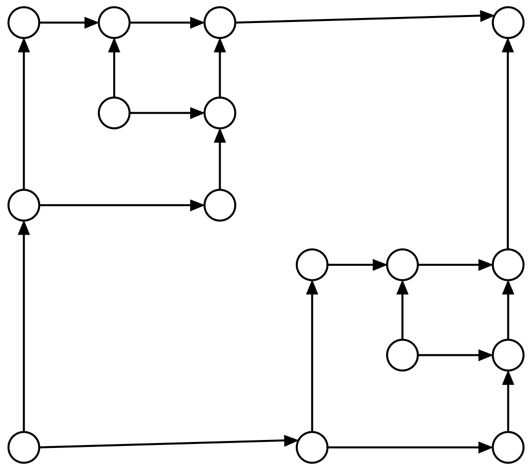

Figure 8.19: Two simply isomorphic components.

1. If G is not reduced, replace each transitive edge (u, v) with a chain of two edges by placing a new vertex w along (u, v) that is the edge (u, v) is tranformed to two edges (u, w) and (w, v)(see Figure 8.22).

2. Call Dominance-Staight-Line.

3. Remove dummy vertices.

□ ...

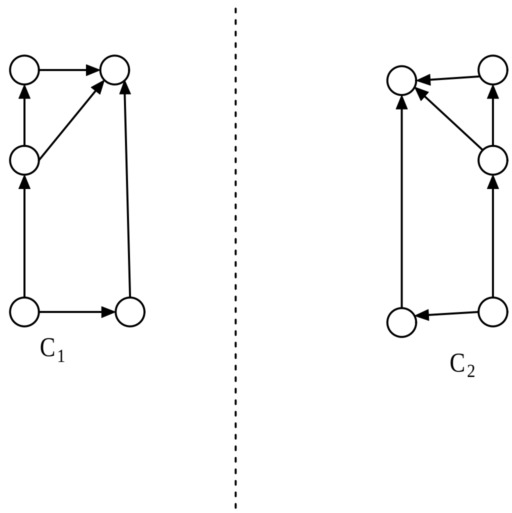

Figure 8.20: Axially isomorphic components.

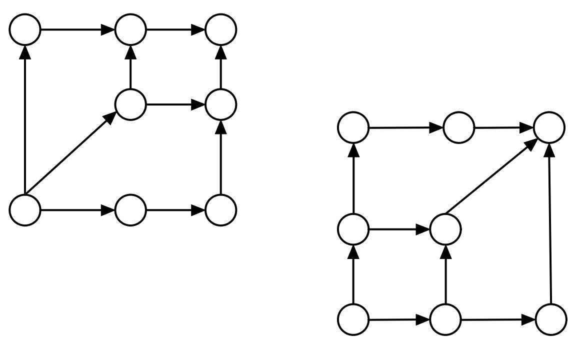

Figure 8.21: Rotationally isomorphic components.

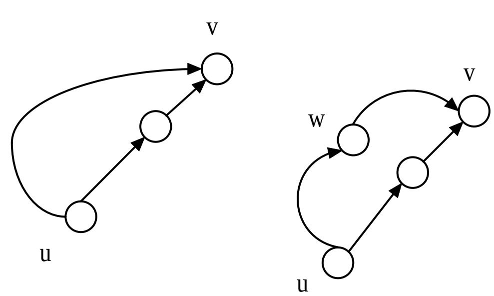

Figure 8.22: The trick for the transitive edges.