# Edge–Disjoint Routing in Plane Switch Graphs in Linear Time

JAN M. HOCHSTEIN [1] and KARSTEN WEIHE [2]

By a *switch graph* we mean an undirected graph $G = (P \dot\cup W, E)$ such that all vertices in $P$ (the *plugs*) have degree one and all vertices in $W$ (the *switches*) have even degrees. We call $G$ *plane* if $G$ is planar and can be embedded such that all plugs are in the outer face. Given a set $(s_1, t_1), \ldots, (s_k, t_k)$ of pairs of plugs, the problem is to find edge-disjoint paths $p_1, \ldots, p_k$ such that every $p_i$ connects $s_i$ with $t_i$.

The best asymptotic worst-case complexity known so far is quadratic in the number of vertices. In this paper, a linear, and thus asymptotically optimal, algorithm is introduced. This result may be viewed as a concluding "keystone" for a number of previous results on various special cases of the problem.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Routing and layout*; G.2.2 [**Discrete Mathematics**]: Graph Theory

General Terms: Algorithms, Theory

Additional Key Words and Phrases: planar graphs, edge–disjoint paths

[1]Technische Universität Darmstadt, FB Informatik, 64283 Darmstadt, Germany, hochstein@algo.informatik.tu-darmstadt.de

[2]The University of Newcastle, School of Electrical Engineering and Computer Science, Callaghan, NSW 2308, Australia, weihe@cs.newcastle.edu.au

## 1. INTRODUCTION

It is well known that this problem is polynomially solvable [Becker and Mehlhorn 1986; Frank 1985]. However, various $\mathcal{NP}$–completeness results for generalizations and variations (*e.g.* [Middendorff and Pfeiffer 1993]) suggest that it is quite "close" to the borderline of polynomiality.

### 1.1 Contribution of the paper

The asymptotically fastest algorithm known so far has been proposed by Becker and Mehlhorn and requires $\mathcal{O}(|P| \cdot |V| + T(|V|))$ time, where $T(n)$ denotes the worst-case time required to solve a full instance, i.e. an instance such that all plugs are terminals, on $n$ vertices [Becker and Mehlhorn 1986]. In [Wagner and Weihe 1995], $T(n) \in \mathcal{O}(n)$ was shown, which implies $\mathcal{O}(|P| \cdot |V|)$ for the whole problem. Clearly, $|V| \cdot |P|$ is quadratic in $|V|$ in the worst case.

In this paper, a completely novel approach is introduced, which results in an algorithm with asymptotic complexity $\mathcal{O}(|V|)$ for simple graphs (note that $|E| \in \mathcal{O}(|V|)$ for simple planar graphs [Nishizeki and Chiba 1988]) and $\mathcal{O}(|V| + |E|)$ in case multiple edges are allowed.

This approach is based on a surprising insight: the graph can be partitioned into two non–trivial parts such that the resulting partial problem in one part can be solved without any knowledge about the other part. This gives rise to a certain kind of divide–and–conquer approach. To the best of the authors' knowledge, this is the first non–trivial disjoint–path problem for which such an approach yields an efficient, optimal algorithm.

### 1.2 Background

The distinction between plugs and switch vertices and the parity condition on all switch vertices are quite natural for various kinds of geometric routing problems. As Figures 1 and 2 demonstrate, this includes every graph that is induced by an arrangement of straight lines in the plane such that the endpoints and intersection points are the vertices and the resulting line segments are the edges (provided no endpoint is surrounded by straight lines from all sides).

For example, the restriction to grid graphs like in Figure 1 (often called the *knock–knee routing model*) yields a common generalization of various classes of routing problems that have been extensively investigated in view of combinatorial VLSI design. Moreover, our problem definition is general enough to follow certain more recent trends, in which one deviates from a pure grid structure and, for example, uses another regular topology like those shown in Figure 2. See [Kaufmann and Mehlhorn 1990; Lengauer 1990; Möhring and Wagner 1997; Ripphausen-Lipa et al. 1995] for an overview.

The intimate relation to various kinds of geometrically embedded computation networks (*e.g.* wide–area networks) is also evident: if every edge is a link and every node a processor connected to an even number $\ell$ of links, every processor may maintain $\ell/2$ information channels simultaneously, which basically means edge–disjoint paths connecting pairwise disjoint pairs of terminals.

Neither for the knock–knee case nor for the topology classes in Figure 2 was a subquadratic algorithm known so far. However, subquadratic algorithms have

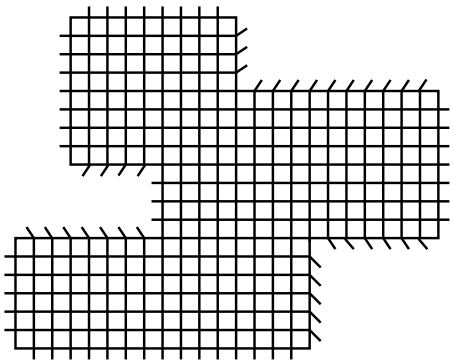

Fig. 1. An example of a switch graph with a grid topology.

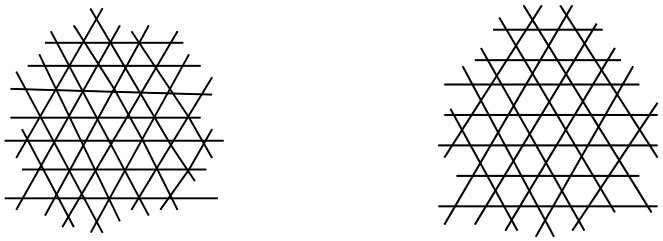

Fig. 2. Further examples of regular structures that fit into the general model.

been proposed for many special cases. Most of these algorithms only apply to full instances [Kaufmann and Klär 1991; Okamura and Seymour 1981; Wagner and Weihe 1995] or even to special cases of the full case. For example, [Kaufmann and Mehlhorn 1986] yields $\mathcal{O}(|V| \log^2 |V|)$ for full instances with arbitrary grid topologies. On the other hand, [Kaufmann 1990] gives a linear-time algorithm for the case that the instance is full and the graph a convex grid. In that, a grid is called *convex* if every horizontal or vertical straight line in the plane crosses the boundary of the outer face at most twice. Note that the grid in Figure 1 is not convex.

In particular, the case that the grid forms a rectangle (often called *switchbox routing*) has attracted a lot of attention. In this special case the restriction to full instances is usually dropped. The first linear–time algorithm was presented in [Frank 1982]. Later work (*e.g.* [Mehlhorn and Preparata 1986]) incorporates further optimization criteria and achieves sublinear complexities through an implicit representation of the output. If the output is to be explicitly represented, each of these algorithms is linear (or slightly superlinear). We refer to [Lengauer 1990; Möhring and Wagner 1997] for further information about this special case.

In general, we refer to [Kaufmann and Mehlhorn 1990; Lengauer 1990; Möhring and Wagner 1997; Ripphausen-Lipa et al. 1995] for a more extensive survey of this broad field and various further extensions, variations, and applications of the

problem, and to [Nishizeki and Chiba 1988] for a more general background on algorithms in planar graphs.

### 1.3 Overview

In Section 2, some basic terminology and facts are reviewed, which will be used throughout the paper. Afterwards, in Section 3, an oracle is presented which finds all cuts of a certain type in a planar graph. An implementation of this oracle is presented and it is shown how this oracle can be efficiently maintained, resulting in linear running time. Section 4 introduces the divide–and–conquer algorithm on a high level depending on the oracle to find the division lines in the graph. Finally, we give a complete proof of the algorithm's correctness in Section 5, relying on the oracle as a "black box."

## 2. PRELIMINARIES

### 2.1 Notation

We use the notation common in graph theory. The edge set and vertex set of a directed or undirected graph $G$ are $E(G)$ and $V(G)$, respectively. An edge $e = (v, w)$ in a digraph has the *head* $w$ and the *tail* $v$. $G[X]$ denotes the subgraph induced by a vertex set $X \subset V(G)$. For a digraph $G$ we define the *reversed graph* $\overleftarrow{G} := (V(G), \{(w, v);\ (v, w) \in E(G)\}$.

In order to easily modify a given graph, we define operations "+" and "−" on graphs and their parts. For graphs $G$ and $H$ and $v \in V(H)$ we set

$$\begin{aligned} G + H &:= (V(G) \cup V(H),\ E(G) \cup E(H)), \\ G + v &:= (V(G) \cup \{v\},\ E(G)). \end{aligned}$$

And for $V(H) \subset V(G)$ and $e \in E(H)$ we define

$$\begin{aligned} G + E(H) &:= (V(G),\ E(G) \cup E(H)), \\ G + e &:= (V(G),\ E(G) \cup \{e\}). \end{aligned}$$

We set

$$G - E(H) := (V(G),\ E(G) \setminus E(H)).$$

If $V(H) \subset V(G)$ then

$$\begin{aligned} G - V(H) &:= G[V(G) \setminus V(H)] \quad \text{and} \\ G - H &:= (G - V(H)) - E(H). \end{aligned}$$

When considering two disjoint vertex sets $X, Y \subset V(G)$, $E(X, Y)$ is the set of edges with one end in $X$ and the other in $Y$. In a digraph this can be divided into the set $E^+(X, Y)$ of edges from $X$ to $Y$, and $E^-(X, Y) = E^+(Y, X)$. We use $E(X)$ and $E^\pm(X)$ as an abbreviation for $E(X, V(G) \setminus X)$ and $E^\pm(X, V(G) \setminus X)$, respectively.

The edges incident to $v \in V(G)$ make up the set $\delta(v) := E(\{v\})$. If in a digraph, then $\delta(v)$ can be divided into the set of outgoing edges $\delta^+(v)$ and the set of incoming edges $\delta^-(v)$.

Finally, we remark that we use $\subset$ always in the non–strict sense, i.e. $X \subset X$, and write $\subsetneq$ otherwise.

2.1.1 *Problem definition.* In the following, $G = (V, E)$ denotes a simple, undirected and planar graph. We assume that $V$ can be partitioned into two vertex sets, $U := U(G)$ and $W := W(G)$, such that all vertices in $U$ have degree one and all vertices in $W$ have even degrees. The vertices in $U$ will be called the *plugs* and the vertices in $W$ the *switches*. Furthermore, we assume that the graph is planar and can be embedded in the plane without crossings of edges such that all plugs are in the outer face. See Figure 3.

Obviously, $|U|$ is even. Let $k \in \{0, \ldots, |U|/2\}$ and let $S = (s_1, \ldots, s_k)$ and $T = (t_1, \ldots, t_k)$ be two ordered sets of plugs such that $s_i \neq s_j$ and $t_i \neq t_j$ for $i \neq j$ and $s_i \neq t_j$ for all $i, j$. The problem that we address in this paper is to find $k$ edge-disjoint paths, $P_1, \ldots, P_k$, for a given instance $(G, S, T)$ such that $P_i$ connects $s_i$ with $t_i$ for every $i \in \{1, \ldots, k\}$. We denote this solution by $P = (P_1, \ldots, P_k)$.

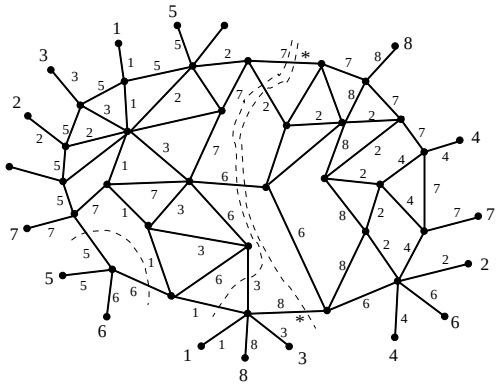

Fig. 3. A plane switch graph $G = (W \dot{\cup} U, E)$, embedded such that all plugs are in the outer face. The instance contains eight pairs of terminals. One feasible solution $(P_1, \ldots, P_8)$ is given by the numbers attached to the edges: number $i$ indicates that an edge belongs to $P_i$. The dashed lines show three saturated $W$–cuts as defined in Section 2.1.

Normally, we represent a solution by an edge label $\lambda : E(G) \to \{0, 1, ..., k\}$ which gives the path index for edges belonging to solution paths, or 0 for unused edges.

The vertices in $S$ and $T$ are the *terminals* and the vertices in $U \setminus (S \cup T)$ the *gaps*. An instance $(G, S, T)$ is called *full* if it does not contain any gaps. The terminal $t_i$ is called the *mate* of $s_i$ and vice versa. The restriction of $E$ to $W$ will be denoted by $E(G[W])$.

We assume a counter-clockwise ordering of all terminals along the boundary of the outer face such that $s_i$ precedes $t_i$ for all $i$ and $t_i$ precedes $t_{i+1}$ for $i = 1, ..., k-1$.

2.1.2 *Boundary.* Without loss of generality, we assume that the subgraph of $G$ induced by $W$ is 2–vertex–connected.[3] The *boundary* of $G$ is the set of edges in $E(G[W])$ incident to the outer face. Note that the edges incident to plugs are not regarded as boundary edges.

2.1.3 *Connectedness.* Without loss of generality, we will assume that the subgraph of $G$ induced by $W$ is *internally* 3*–edge–connected*. By that we mean that, after removing any two edges, every connected component shares at least one edge with the boundary of $G$. We now motivate this assumption, cf. Figure 4:

If $G$ is not 3–edge–connected, let $G'$ be a subgraph of $G$ that does not share edges with the boundary of $G$ and is connected to the rest of $G$ by at most two edges. If it is connected by only one edge, we may safely remove $G'$ and its connecting edge. Otherwise, we may safely replace $G'$ and its two connecting edges by a single edge. Since each edge separator of $G$ is a cycle in the dual graph, all subgraphs $G'$ can be identified in linear time in total: it suffices to identify all dual cycles of length two. To do this we number all primal faces (dual vertices) and collect the pairs of numbers which represent the two faces adjacent to each edge. In these pairs the smaller of the two numbers occupies the first position. We then sort these pairs

[3]Otherwise, each 2–vertex–connected component $C$ of this subgraph could be treated separately by shrinking each connected component of $W \setminus C$ to a single vertex.

lexicographically using bucket sort (cf. [Knuth 1975]). If two successive pairs are equal then we have found a dual cycle of length two.

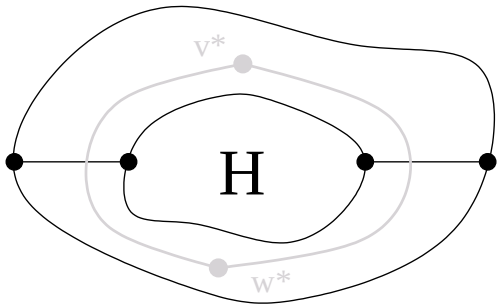


Fig. 4. Two faces (dual vertices) $v^*$ and $w^*$ of $G$ enclose a subgraph $H$. Thus $G$ is not internally 3–edge–connected. Dual edges are drawn in grey showing the dual cycle of length 2.

2.1.4 *Cuts.* By $(X, Y)$ we denote a partition of $V$ into two connected subsets, that is, $X \neq \emptyset$, $Y \neq \emptyset$, $X \cup Y = V$, and $X \cap Y = \emptyset$. We call $(X, Y)$ a *$W$–cut* if no edge incident to a plug belongs to $E(X, Y)$. The *supply* of $W$–cut $(X, Y)$ is simply the number of edges in $E(X, Y)$, $\mathrm{supp}(X, Y) = |E(X, Y)|$. On the other hand, the *demand* of $(X, Y)$, $\mathrm{dem}(X, Y)$, is the number of $i \in \{1, \ldots, k\}$ such that $s_i \in X$ and $t_i \in Y$ or vice versa. We abbreviate $\mathrm{dem}(X) := \mathrm{dem}(X, V \setminus X)$ and $\mathrm{supp}(X) := \mathrm{supp}(X, V \setminus X)$. A $W$–cut $(X, Y)$ is called *saturated*, if $\mathrm{dem}(X, Y) = \mathrm{supp}(X, Y)$. See Figure 3.

An instance is said to fulfill the *cut condition* if $\mathrm{dem}(X, Y) \leq \mathrm{supp}(X, Y)$ for every $W$–cut $(X, Y)$. Obviously, the cut condition is a necessary condition for an instance to be solvable. However, in general it is not sufficient. The seminal work by Okamura and Seymour has shown that the cut condition is sufficient for full instances [Okamura and Seymour 1981]. Frank proved that a certain generalization of the cut condition to families of $W$–cuts is necessary *and* sufficient for non-full instances [Frank 1985]. See Figure 5 for an example of a non–full instance that satisfies the cut condition but is not solvable.

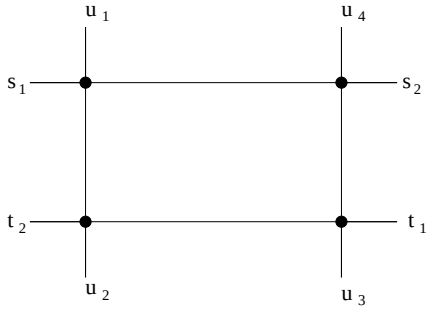


Fig. 5. This instance with gaps $u_1$ through $u_4$ is unsolvable though the cut condition is fulfilled.

To simplify the notation in the following, we will regard a $W$–cut $(X, Y)$ as ordered, that is $(X, Y) \neq (Y, X)$.

2.1.5 *Path crossings.* We make extensive use of the rather graphic concept of crossing paths. We say two paths $P_i$ and $P_j$ from the solution $P$ *cross each other in* $v$, if in any cyclic ordering of $v$'s incidence list there is exactly one edge from $P_i$ between the two edges from $P_j$.

2.1.6 *Algorithmic notation.* We present some algorithms and procedures in this paper. These are written as pseudo-code designed for better readability. Next we define some operations we use in several procedures: Whenever we speak of a stack $A$, we mean – as usual – an ordered set which can be accessed from one side only by operations *push* and *pop*. Thereby, $\text{push}(A, v)$ pushes the object $v$ onto the stack and $v := \text{pop}(A)$ removes the topmost element from $A$ and assigns it to $v$. We usually assume the incidence lists of vertices to be sorted in a cyclic order (clockwise or counter-clockwise) according to the fixed embedding of $G$ into the plane. Thus, for $e \in \delta(v)$ the edge $ccnext(v, e)$, which is the counter-clockwise next edge after $e$ in the incidence list of $v$, is easily available.

## 2.2 Leftmost and rightmost paths and branchings

We present some results that we will later apply to the directed auxilliary graphs in Section 3. In this subsection $G$ will always be a directed plane switch graph with its set of terminals $\{x_0, ..., x_{h-1}\} = S \cup T$ numbered such that $x_i$ precedes $x_{i+1}$ in a counter-clockwise ordering along the boundary of $G$. We allow parallel edges in $G$. Nevertheless, to avoid clumsy notation we write $e = (u, v)$ because it always becomes clear from the context which edge we mean.

Let $P$ be a directed path in $G$ from a terminal $s$ to a terminal $t$. Then $P$ partitions the area of $G$ into two parts, a left part and a right part, where the right part is the one appearing on the right side when we walk along $P$ from $s$ to $t$. We say that $P$ is *leftmost* if the right side is inclusion–maximal among all paths from $s$ to $t$.

We extend the definition: A $v$–$w$–path $P$ is *leftmost* if for no $v$–$w$–path $Q$ the graph $C = P + \overleftarrow{Q}$, defined by going forward along $P$ and then backwards along $Q$ contains a clockwise cycle. Obviously, leftmost paths are unique if they exist.

A further extension of our definition: a $x_k$–$v$–path $P$ is *the leftmost* path to $v$, if $P$ is a leftmost $x_k$–$v$–path and $v$ is not reachable from $x_0, x_1, ..., x_{k-1}$. In accordance with the previous definition, we say a terminal $x_k$ is *to the left* (resp. *to the right*) of a terminal $x_\ell$ if $k < \ell$ (resp. $k > \ell$).

Later we will also need righmost paths. Here we only define "leftmost" and remark that one can define "rightmost" in the same way and prove mirror-symmetric results by replacing "left" with "right" and "counter-clockwise" with "clockwise" in the appropriate places.

Next we analyze the relationship between primal and dual leftmost and rightmost paths. We define the dual graph as $G^* = (W^* \cup P^*, E(G))$. Here, $W^*$ denotes the set of inner faces and $P^* = \{y_0, ..., y_{h-1}\}$ the set of dual terminals where $y_i$ is the unique dual terminal between $x_{i-1}$ and $x_i$. Each dual edge crosses its corresponding primal edge from right to left. See Figure 6 for an example.

LEMMA 2.1. *Let $G$ be a directed plane switch graph without clockwise cycles.*

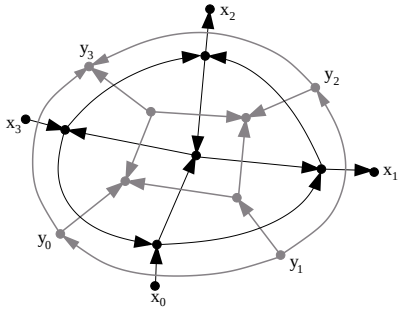


Fig. 6. The dual graph $G^*$ (grey) of an exemplary primal graph $G$ (black).

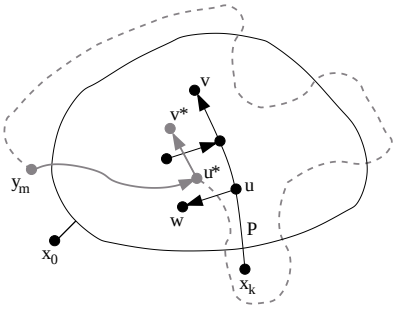


Fig. 7. The situation in the first part of the proof of Lemma 2.1. A possible outline of the vertex set $A$ is indicated in dashed grey.

*Then a $x_k$–$v$–path $P$ is the leftmost path to $v \in V(G)$, if and only if for each dual vertex $v^*$ incident to $P$ from the left there exists a dual $y_m$–$v^*$–path $R^*_{v^*}$ that does not cross $P$, where $y_m \in \{y_{k+1}, ..., y_{h-1}, y_0\}$.*

Proof. We can disregard the trivial case $v \in U(G)$. If $P$ is the leftmost path to $v$ then it contains the leftmost path between any two vertices $u$ and $w$ where $u$ precedes $w$ (not necessarily immediately) on $P$. Therefore we need to consider only the dual vertex $v^*$ that is incident from the left to the last edge on $P$. Let $L^-(P)$ be the set of edges pointing to $P$ from the left, i.e.

$$L^-(P) \;=\; \{e \in E^-(V(P) \setminus \{v\});\; e \text{ incident to } P \text{ from the left}\},$$

and analogously

$$L^+(P) \;=\; \{e \in E^+(V(P) \setminus \{v\});\; e \text{ incident to } P \text{ from the left}\}.$$

We construct an auxilliary graph by removing from $G$ each edge $(w, u) \in L^-(P)$ if there is also an edge $(u, w) \in E(G)$. Consider the set $A$ of all vertices from which $v$ is reachable in $G'$. Let $u$ be the vertex on $P$ nearest to $v$ such that $u$ is incident to an edge $(u, w) \in L^+(P)$; and let $u^*$ be the dual vertex incident to $P$ from the left and incident to $(u, w)$ from the right (cf. Figure 7). If there is no such $u$, then we set $u := x_k$, $w := x_{k-1}$ and $u^* := y_k$. Since $P$ is leftmost and $G$ contains no clockwise cycles, $A$ is disjoint to $\{x_0, x_1, ..., x_{k-1}\}$ as well as to $\{w\}$.

Thus a subset of $E^+(A)$ defines a dual path that begins at some dual terminal $y_m \in \{y_{k+1}, ..., y_{h-1}, y_0\}$, ends at $u^*$ and does not cross $P$. In other words, we have found a path $R^*_{u^*}$. Moreover, by the choice of $u^*$, there is a path from $u^*$ to $v^*$ that uses only edges from $L^-(P)$. Hence we can extend $R^*_{u^*}$ to a path $R^*_{v^*}$. That shows necessity.

Due to the definition of $R^*_{v^*}$, there is no path from the left side of $P$ or from $\{x_0, x_1, ..., x_{k-1}\}$ to the right side of $P$ that does not cross $P$. Therefore, any path $Q$ that was more to the left than $P$ would have to join $P$ from the left at some vertex $w$. That is impossible because of the dual path ending at the face incident from the left to the edge $(u, w) \in E(P)$ where $u$ is the vertex immediately preceding $w$ on $P$. Note that this is also true for $w = v$. □

The dual graph $G^*$ contains no counter-clockwise cycles if and only if every connected set of primal vertices $A$ has an incoming edge, i.e. $E^-(A) \neq \emptyset$. And the latter, in turn, holds if and only if each vertex $v \in V(G)$ is reachable from some terminal. As always there is a mirror–symmetric result for the dual graph. Thus we get

PROPOSITION 2.2.

—*The leftmost (and rightmost) paths to all vertices $v \in V(G)$ exist if and only if the dual graph $G^*$ contains no counter-clockwise cycles.*

—*The leftmost (and rightmost) paths to all dual vertices $v^* \in V(G^*)$ exist if and only if the primal graph $G$ contains no clockwise cycles.*

Let $P$ be a path found by walking forward along arcs in $G$ starting with an arc $e$. At each vertex $w$ which we reach by an arc $(v, w)$, we choose the clockwise next unused outgoing arc after $(v, w)$ to go on. We stop, when there are no unused outgoing arcs. Quite intuitively, we call $P$ the *left-first path of* $e$ in $G$. We define the *reverse left-first path of* $e$ as the left-first path of $e$ in $\overleftarrow{G}$.

We say, a subgraph $F$ in a directed plane switch graph is a *spanning branching*, if it contains exactly one path from the set of plugs $P$ to each $v \in V$ and each edge in $F$ belongs to such a path. A spanning branching is *leftmost*, if it contains the leftmost path to each $v \in V$. $F$ is a branching and unique by the uniqueness of leftmost paths. And by Proposition 2.2 it exists, if and only if $G^*$ has no clockwise cycles.

The following result shows that the primal leftmost branching and the dual rightmost branching form a partition of a planar graph's inner edges.

THEOREM 2.3. *Consider a graph $G$ and its dual $G^*$. If $F$ is the leftmost branching in $G$ and $F^*$ the rightmost branching in $G^*$ then for each primal edge $e \in E(G)$ and its dual edge $e^*$ exactly one of the following statements holds:*

—*$e$ belongs to $F_i$ or*

—*$e^*$ belongs to $F^*_i$.*

PROOF. First, we consider a primal edge $e = (v, w)$ that belongs to $F$ and its corresponding dual edge $(v^*, w^*)$. We choose $k$ and $\ell$ such that the leftmost path $P$ to $v$ begins at $x_k$ and the rightmost path $Q^*$ to $v^*$ begins at $y_\ell$. That situation is shown in Figure 8a. Since $P + (v, w)$ is the leftmost path to $w$, by Lemma 2.1

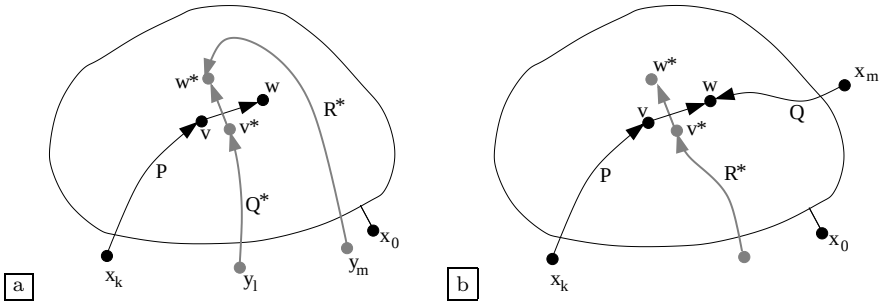


Fig. 8. The proof of Theorem 2.3. As always, the primal graph is black and dual vertices and edges are grey.

we have a semi-cut $R^*$ from a terminal $y_m \in \{y_{k+1}, ..., y_{h-1}, y_0\}$ to $w^*$. If $y_m$ lay to the left of $y_\ell$ then we could walk along $Q^*$ to where the latter crosses $R^*$ and would thus obtain a path which is more to the left than $Q^*$. Hence $y_m$ lies to the right of $y_\ell$. Therefore $R^*$ is more to the right than $Q^*$ and so $e^* = (v^*, w^*) \notin F^*$. Which shows that no edge belongs to both primal and dual branching.

It remains to show that each edge belongs to $F$ or $F^*$. The following argument is depicted in Figure 8b. Consider an inner primal edge $e = (v, w)$ that does not belong to $F$. We choose $k$ and $m$ such that the leftmost path $P$ to $v$ begins at $x_k$ and the leftmost path $Q$ to $w$ begins at $x_m$. Since $P+(v, w)$ is not the leftmost path to $w$, $x_m$ lies to the left of $x_k$. We consider the rightmost path $R^*$ to $v^*$ and apply the mirror–symmetric "righmost version" of Lemma 2.1 to the dual graph $G^*$. In order to show, that $R^* + (v^*, w^*)$ is the rightmost path to $w^*$ we only have to find a semi-cut w.r.t. $w^*$, that begins at a terminal $x_\ell \in \{x_0, x_1, ..., x_{k-1}\}$ and ends at $w$ and does not cross $R^* + (v^*, w^*)$. Obviously, $Q$ is just such a path. We conclude that $e^*$ is an edge on the rightmost dual path to $w^*$ and as such is contained in $F^*$. □

We now show how to construct a leftmost branching in linear time. For each vertex $v \in V(G)$ we only store its unique incoming arc $\phi_i(v)$. If there is no such arc, then $\phi_i(v) = \emptyset$. Procedure 1 uses left-first depth-first search and thus finds the leftmost paths to all reachable vertices. We touch each vertex at most once and probe each arc at most twice. That shows the linear running time.

**Procedure 1** Find the leftmost branching.

**Input:** A directed plane switch graph $G$ with all terminals $x_0, ..., x_{h-1}$ numbered counter-clockwise around the boundary.

**Output:** A function $\phi : V(G) \to E(G)$ giving the incoming arc $\phi(v)$ in the branching for each $v \in V(G)$.

Initialize $\phi(v) := \emptyset$ for all $v \in V(G)$.
Denote the set of unvisited vertices by $B$ and store the path leading from $x_j$ to the current vertex in the stack $Q$ and set $B := V(G)$ in the beginning.
**for** $j = 0, 1, ..., h-1$ **do**
  Let $e$ be the unique arc incident to $x_j$.
  Set $B := B \setminus \{x_j\}$, $Q := \{e\} \cap E(G)$ and $v := x_j$.
  **while** $Q \neq \emptyset$ **do**
    **if** $\phi(v) = \emptyset$ and head$(e) = v$ **then** set $\phi(v) := e$.
    Let $f$ be the clockwise next arc after $e$ which leaves $v$ and leads to $w \in B$.
    **if** there is no such arc **then**
      Set $e :=$ pop$(Q)$ and $v :=$ head$(e)$.
    **else**
      push$(Q, f)$ and set $B := B \setminus \{w\}$, $e := f$ and $v := w$.
    **end if**
  **end while**
**end for**

## 3. THE ORACLE

We describe an oracle that finds a sequence of non-crossing $W$–cuts in our input instance in linear time. "Non-crossing" means that the corresponding dual paths do not cross. These cuts partition the input instance into partial instances that contain no saturated $W$–cuts. Such instances can then be solved using a technique from [Becker and Mehlhorn 1986] as we will show in Section 5.

### 3.1 Definition

3.1.1 *Candidate/relevant cuts.* Consider a boundary edge $\{v, w\}$ where $v$ immediately precedes $w$ when we walk along the boundary of $G$ in counter-clockwise direction. There may or may not be saturated $W$–cuts $(X, Y)$ such that $w \in X$ and $v \in Y$. These $W$–cuts will be called *the candidate cuts of* $\{v, w\}$. Among them, one will be called *the relevant cut of* $\{v, w\}$, namely the $W$–cut $(X, Y)$ of $\{v, w\}$ with $X \subseteq X'$ for all other candidate cuts $(X', Y')$ of $\{v, w\}$.

For example, the saturated $W$–cut $(X, Y)$ used in Figures 15 and 16 is a candidate cut for the upper boundary edge indicated by a '*' in Figure 3, but is not the relevant cut of this edge (the other depicted cut crossing this edge is its relevant cut). On the other hand, $(Y, X)$ is the only candidate cut of the lower boundary edge labeled by a '*' and thus its relevant cut.

LEMMA 3.1. *In a solvable instance* $(G, S, T)$ *the cut* $(X_1 \cap X_2,\ V(G) \setminus (X_1 \cap X_2))$ *defined by the intersection of the two saturated cuts* $(X_1, V(G) \setminus X_1)$ *and* $(X_2, V(G) \setminus X_2)$ *is saturated.*

PROOF. The following two equations hold for any two connected sets of vertices, in particular for $X_1$ and $X_2$:

$$\begin{aligned} \mathrm{supp}(X_1 \cap X_2) &= \mathrm{supp}(X_1) + \mathrm{supp}(X_2) - \mathrm{supp}(X_1 \cup X_2) \quad \text{and} \\ \mathrm{dem}(X_1 \cap X_2) &= \mathrm{dem}(X_1) + \mathrm{dem}(X_2) - \mathrm{dem}(X_1 \cup X_2). \end{aligned}$$

We subtract, and since both cuts are saturated, we get

$$\mathrm{supp}(X_1 \cap X_2) - \mathrm{dem}(X_1 \cap X_2) = \mathrm{dem}(X_1 \cup X_2) - \mathrm{supp}(X_1 \cup X_2).$$

As $(G, S, T)$ is solvable, the lefthand-side is non-negative and the righthand-side is non-positive. Therefore both sides must be zero, which shows $\mathrm{supp}(X_1 \cap X_2) = \mathrm{dem}(X_1 \cap X_2)$. □

COROLLARY 3.2. *In a solvable instance* $(G, S, T)$*, the relevant cut of* $\{v, w\}$ *exists unless* $\{v, w\}$ *does not have candidate cuts at all. In other words, the partial order on all candidate cuts of* $\{v, w\}$ *induced by the relation* $(X, Y) \prec (X', Y') \iff X \subset X'$ *has a unique minimum.*

COROLLARY 3.3. *The relevant cuts of different boundary edges do not cross each other, i.e. for each pair of relevant cuts* $(X_1, V(G) \setminus X_1)$ *and* $(X_2, V(G) \setminus X_2)$ *we have* $X_1 \subset X_2$ *or* $X_2 \subset X_1$*.*

3.1.2 *The Oracle's Specification.* Given an instance $(G, S, T)$ and a boundary edge the oracle finds out whether this edge has a candidate cut. More precisely, for a boundary edge $\{v, w\}$, exactly one of the following answers will be returned:

(1) A candidate cut of $\{v, w\}$.

(2) The information that $\{v, w\}$ does not have candidate cuts.

(3) The information that the input instance is unsolvable.

Answer #1 or #2 may be given even if the input instance is unsolvable. However, if the input instance is solvable and answer #1 is given, the returned $W$–cut will be the relevant cut of $\{v, w\}$. We can only guarantee linear running time in total, if the oracle is consulted once for each edge on the boundary of $G$ in a counter-clockwise order.

By consulting the oracle in that way, we get a sequence of non–crossing $W$–cuts which define a partition of the input instance.

From [Becker and Mehlhorn 1986] we adopt the concept of $U$–minimal cuts: a saturated $W$–cut $(X, Y)$ is called *$U$–minimal* if $X' \subset X$ for no other saturated $W$–cut $(X', Y')$. Note that the relevant cut of a boundary edge $\{v, w\}$ is not necessarily $U$–minimal. But at least one relevant cut is $U$–minimal and every $U$–minimal cut is the relevant cut of some boundary edge.

3.1.3 *Doubled instance.* Let $G' = (V', E')$ denote the graph that is constructed from $G$ by doubling all edges (in other words, every edge in $E$ is replaced by two parallel edges). Each such pair induces an additional face in between, whose boundary thus has length two. Moreover, each terminal is duplicated in $V'$, whereas all gaps and their incident edges are removed. The copies of a terminal $s_i$ are named $s'_i$ and $s''_i$ ($t_i$ analogously). Let $(G', S', T')$ denote the doubled instance. See Figure 9. Obviously, if $(G, S, T)$ is feasible, then $(G', S', T')$ is so (but not necessarily vice versa).

In $G'$ there may be parallel edges. Like in Subsection 2.2, we are rather sloppy and depend on the context to clarify which edge we mean by $\{u, v\}$.

3.1.4 *Cyclic numbering of terminals.* Let $K = 2k$. In the following, $x_0, \ldots, x_{K-1}$ is an enumeration of all terminals in counter-clockwise ordering around $G'$, starting at an arbitrary yet fixed terminal (see Figure 9). Whenever we refer to a vertex $x_i$, expressions such as $x_{i-1}$ and $x_{i+1}$ are to be regarded as "mod $K$." We write "$k \leq i \leq \ell \mod K$" to indicate that $i$ lies in the interval $\{k, k+1, k+2, ..., \ell-1, \ell\}$ where each member is taken mod $K$.

3.1.5 *Dual graph.* Figure 10 shows the *multiple-source dual graph* $G^* = (V^*, E^*)$ of $G'$. This means that we have $V^* = W^* \dot\cup U^*$, where every vertex in $W^*$ represents an internal face of $G'$ and every vertex in $U^*$ a section on the boundary of $G'$ between two successive terminals. As shown in Figure 10, let $y_0, \ldots, y_{K-1}$ denote the vertices in $P^*$ such that $y_i$ represents the section of the boundary between $x_{i-1}$ and $x_i$. Note that $G^*$ is bipartite, with the faces induced by the pairs of parallel edges being one bipartition set.

3.1.6 *Dual interpretation of cuts.* Obviously, a $W$–cut of $G'$ that crosses edges on the boundary may be identified with an undirected simple path in $G^*$ that has both end vertices in $P^*$ (here *simple* means: no repetition of vertices or edges). In the following, a $W$–cut $(X, Y)$ that corresponds to a dual path with end vertices $y_i$ and $y_j$ will be called a $\{y_i, y_j\}$–cut. Obviously, all $\{y_i, y_j\}$–cuts have equal demand, which we denote by $\mathrm{dem}(y_i, y_j)$ $(= \mathrm{dem}(y_j, y_i))$, but may have different supplies.

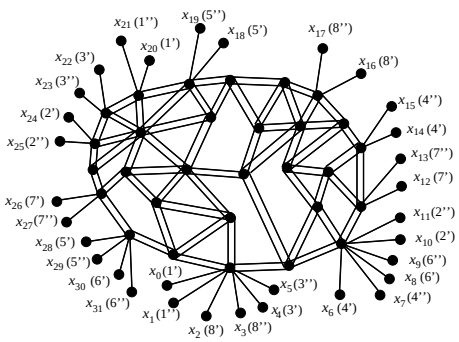


Fig. 9. The doubled instance $(G', S', T')$ for the input instance $(G, S, T)$ from Figure 3 and a possible enumeration $x_0, \ldots, x_{K-1}$ $(K = 2k)$.

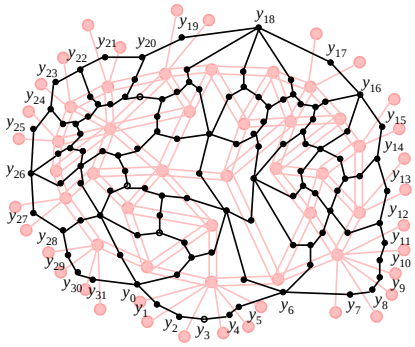


Fig. 10. The graph $G'$ of Figure 9 (grey), the multiple-source dual graph $G^*$ of $G'$ (black), and the dual vertices $y_0, \ldots, y_{K-1}$.

3.1.7 *Distance weightings.* Next we describe a set of weightings $d_0, \ldots, d_{K-1}$ of the vertices in $G^*$. An example is depicted in Figure 11. For $i$, $j \in \{0, \ldots, K-1\}$, we define $d_i(y_j)$ as twice the negative demand of an $\{y_i, y_j\}$–cut: $d_i(y_j) = -2 \cdot \mathrm{dem}(y_i, y_j)$. In particular, the values $d_i(y_j)$ form a metric on $\{0, \ldots, K-1\}$.

Such a weighting $d_i$ of $U^*$ can be extended to all vertices of $G^*$ as follows: for $v^* \in W^*$, $d_i(v^*)$ is defined to be the minimum distance to any vertex $y_j \in U^*$ in $G^*$ plus a correction value of $d_i(y_j)$. To put this definition in more formal terms: for $v^*$, $w^* \in V^*$ let $\Delta(v^*, w^*)$ denote the minimum number of edges on any $v^*$–$w^*$–path in $G^*$. Then $d_i(v^*)$ is defined as

$$d_i(v^*) = \min\{d_i(y_j) + \Delta(y_j, v^*) : 0 \le j < K\}. \tag{1}$$

LEMMA 3.4. *If the input instance $(G, S, T)$ is solvable, every weighting $d_i$ is* consistent *in the sense that Equation* (1) *is also valid for every $v^* \in P^*$.*

PROOF. If Equation (1) is not valid for some $i$ and some $v^* = y_j$, there is $\kappa \in \{0, \ldots, K-1\} \setminus \{j\}$ such that $d_i(y_j) > d_i(y_\kappa) + \Delta(y_j, y_\kappa)$. However, $\Delta(y_j, y_\kappa)$ is the supply of some $\{y_j, y_\kappa\}$–cut. On the other hand, due to the triangle–inequality, $|d_i(y_j) - d_i(y_\kappa)|$ is a lower bound on the demand of every $\{y_j, y_\kappa\}$–cut. Hence, this $W$–cut violates the cut condition, and $(G, S, T)$ is thus unsolvable. □

LEMMA 3.5. *If a weighting $d_i$ of $V^*$ is consistent in the sense of Lemma 3.4, then we have $|d_i(v^*) - d_i(w^*)| = 1$ for any two vertices $v^*$, $w^* \in V^*$ that are adjacent in $G^*$.*

PROOF. Since $G^*$ is bipartite, the $d_i$-values of any two adjacent vertices in $V^*$ have different parities. Hence, the assumption $|d_i(v^*) - d_i(w^*)| \neq 1$ implies $d_i(w^*) > d_i(v^*) + 1$ (or vice versa). Let $j \in \{0, \ldots, K-1\}$ such that $d_i(v^*) = d_i(y_j) + \Delta(y_j, v^*)$. The shortest dual $(y_j, v^*)$-path extended by $\{v^*, w^*\} \in E^*$ yields the contradiction $d_i(w^*) > d_i(y_j) + \Delta(y_j, w^*)$. □

The distance weighting $d_e$ of all dual vertices for a boundary edge $e \in E$ is defined to be the weighting $d_i$ for the dual vertex $y_i$ incident to $e$.

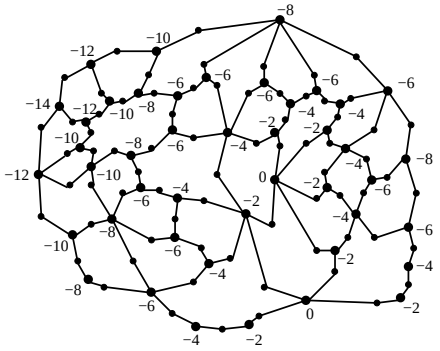

Fig. 11. The weighting $d_i$ for the dual graph shown in Figure 6 with $i = 6$. More specifically, only the values of one partition set of this bipartite graph are depicted. According to Lemma 3.5, the $d_i$–value of any other vertex $v^*$ is computed as follows: if its two adjacent vertices have equal $d_i$–values, $d_i(v^*)$ is one higher, otherwise $d_i(v^*)$ is the mean of both incident $d_i$–values.

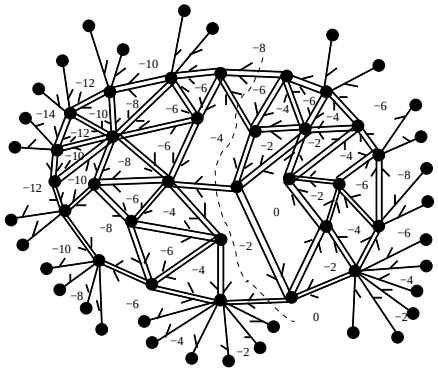

Fig. 12. The orientations of the edges of $G'$ as induced by the weighting $d_6$ from Figure 11. For an easier overview, the orientation of an edge is only indicated by a "half-arrow." The depicted $W$–cut visualizes the statement of Theorem 3.6 for the lowest crossed edge.

3.1.8 *Induced orientation.* For any two vertices $v^*$, $w^* \in V^*$ that are adjacent in $G^*$, we orient the edge $e \in E'$ between the faces of $v^*$ and $w^*$ such that the one out of $v^*$ and $w^*$ with smaller $d_i$-value (which exists due to Lemma 3.5) is on the right side, see Figure 12. The orientation induced by $d_i$ is denoted by $o_i : E(G') \to V(G') \times V(G')$ where $o_i(e)$ is the orientation of $e \in E'$ induced by $d_i$. By $G'_i = (V(G'),\ o_i(E(G'))$ we denote the doubled graph oriented according to $o_i$. Each primal orientation induces a dual orientation by the following rule: All dual edges $e^*$ are oriented such that they cross their corresponding primal edge from right to left, i.e. they are oriented from smaller to larger $d_i$-values. The dual graph directed according to $o_i$ will be called $G^*_i$.
The following theorem states that the orientations $o_i$ are indeed an oracle for the candidate cuts of a boundary edge.

THEOREM 3.6. *Let $e \in E(G)$ be a boundary edge and $y_i$ the incident external dual vertex. Each candidate cut of $e$ uniquely corresponds to a directed $y_i$–$y_j$–path in $G'_i$ and vice versa.*

PROOF. Let $(X, Y)$ be a $\{y_i, y_j\}$-cut and $P^*$ the corresponding undirected dual $y_i$–$y_j$–path such that $X$ is on the right side of $P^*$, when walking from $y_i$ to $y_j$. Recall that for each dual edge $(v^*, w^*)$ we have $d_i(w^*) = d_i(v^*) + 1$. Now, when we go along $P^*$ from $y_i$ to $y_j$, $d_i(\cdot)$ is

—incremented once for each primal edge we pass "from right to left" and
—decremented once for each primal edge we pass "from left to right."

That means $d_i(y_j) = d_i(y_i) - |E_{G'_i}(X, Y)| + |E_{G'_i}(Y, X)|$. By definition we have $d_i(y_j) = -2 \operatorname{dem}(X, Y)$, $d_i(y_i) = 0$ and $|E_{G'_i}(X, Y)| + |E_{G'_i}(Y, X)| = 2 \operatorname{supp}(X, Y)$. Thus $(X, Y)$ is saturated if and only if $P^*$ is directed. □

COROLLARY 3.7. *Again, let $e \in E(G)$ be a boundary edge and $y_i$ the incident external dual vertex. The relevant cut of $e$ exists, if and only if the dual right-first path $P^*$ of $e^*$ in $G_i^*$ ends at a dual terminal. In that case $P^*$ is directed and rightmost. Therefore the corresponding primal cut is relevant.*

In order to find all these right-first paths in linear time in total, we need to maintain a dual rightmost spanning branching (cf. Subsection 2.2). Back-tracking in that branching then yields the paths we are looking for. Since, later on, we also need a primal leftmost branching, we only maintain the latter. By Theorem 2.3, the dual rightmost branching is then implicitly given by the edges that do not belong to the primal leftmost branching.

The orientation $o_0$, the dual graph $G^*$ and the primal leftmost spanning branching in the graph $G'$ oriented according to $o_0$ can be constructed in linear time.

### 3.2 Maintenance

Clearly, if we want to achieve a linear run time in total, we cannot afford to construct the oracle from scratch for every boundary edge. Hence, we need efficient procedures to maintain the oracle, when the focus of the calling algorithm shifts to the counter-clockwise next edge on the boundary of $G$. The solution to this task will be based on Lemma 3.8 below.

To formulate this lemma, let $i \in \{0, \ldots, K-1\}$. For every pair $(s'_j, t'_j)$ (resp. $(s''_j, t''_j)$) of terminals to be connected in the doubled instance $(G', S', T')$, $\alpha'(i,j)$ ($\alpha''(i,j)$) is the one of them which appears first when we walk around the boundary of $G$ in counterclockwise direction, starting with $y_i$. The other one is denoted by $\beta'(i,j)$ ($\beta''(i,j)$).

LEMMA 3.8. *The orientation $o_i$ defined above has the following properties:*

*(O1) For each $v \in W$ the number of incident edges oriented by $o_i$ towards $v$ equals the number of incident edges oriented away from $v$.*

*(O2) All directed cycles in $G'_i$ are counter-clockwise.*

*(O3) Each edge incident to an $\alpha$-terminal is directed by $o_i$ away from this terminal.*

*(O4) Each edge incident to a $\beta$-terminal is directed by $o_i$ towards this terminal.*

PROOF. (O3)+(O4): By definition $d_i(y_{j+1}) = d_i(y_j) + 1$, if an $\alpha$–terminal lies between $y_j$ and $y_{j+1}$. Analogously for $\beta$–terminals.
(O1): Let $v_0^*, ..., v_{k-1}^*$ be the faces in $G'$ incident to $v$ in counterclockwise order and set $v_k^* := v_0^*$. In the dual graph $G^*$ these vertices form a cycle $C$. By Lemma 3.5 we have $|d_i(v_{j+1}^*) - d_i(v_j^*)| = 1$ for all $j$. On the other hand the cyclic sum

$$\sum_{j=0}^{k-1} (d_i(v_{j+1}^*) - d_i(v_j^*)) = 0.$$

Therefore, the number of increments equals the number of decrements on the edges of $C$.
(O2): Since $d_i$ is a shortest path labeling on $G^*$, each inner vertex $v^*$ is connected to some $y_j$ by a directed path which runs from $y_j$ to $v$. Hence there can be no clockwise cycle around the face $v^*$ in $G$. □

LEMMA 3.9. *Two orientations $o$ and $o'$ which have the properties (O1) and (O2) differ exactly on the edges of certain edge-disjoint paths, each of which connects two terminals and is a directed path both in $o$ and $o'$.*
*In particular, for $i \in \{0, ..., K-1\}$ the orientation $o$ is uniquely defined by the properties (O1)–(O4).*

PROOF. $o(E(G'))$ and $o'(E(G'))$ differ on a set of edges, say $D \subset E(G')$. Property (O1) implies that for each vertex $v \in W$ the number of edges in $D$ oriented towards $v$ by $o$ equals the number of edges in $D$ oriented away from $v$ by $o_i$. Therefore $D$ can be partitioned into paths and cycles. More specifically the paths connect terminals and the paths and cycles are directed both in $o_i$ and in $o'_i$. $D$ cannot contain such cycles, however, because switching the orientation of all edges on a counter-clockwise cycle would result in a clockwise cycle, contradicting property (O2). That shows the first part of the assertion.
If, in addition, $o$ and $o'$ have properties (O3) and (O4) then $D$ cannot contain directed terminal connecting paths either, because the orientation of all edges incident to terminals is given by these properties. Thus $D$ is empty and $o(E(G')) = o'(E(G'))$. □

Property (O1) states a kind of flow conservation rule. From that we deduce

LEMMA 3.10. *Let $X$ be a connected vertex set with $E^+(X) = \emptyset$. The number $|E^-(X)|$ of edges entering $X$ is equal to the number of pairs $(x_k, mate(x_k))$ such that $x_k \in X$ is a $\beta$-terminal and its mate (an $\alpha$-terminal) lies outside $X$.*

THEOREM 3.11. *Let $x_j$ be the mate of $x_i$. If $(G, S, T)$ is solvable, the leftmost path from $x_i$ to $x_j$ with respect to $o_i$ exists, and the edges of this path are exactly the edges on which $o_i$ and $o_{i+1}$ differ.*

PROOF. If there is no $x_i$–$x_j$–path in $G'_i$, then there exists a $x_i$-$x_j$-separating cut $(X, V \setminus X)$ such that $x_i \in X$ and $E^+(X) = \emptyset$. Thus the dual path corresponding to $(X, V \setminus X)$ is directed and the cut is saturated by Theorem 3.6. Lemma 3.10 shows: For each edge $e \in E^-(X)$ we have a $\beta$-terminal $x_k$ in $X$ and its mate in $V \setminus X$. Denote the set of all such pairs $(x_k, mate(x_k))$ by $M$. Since $x_i$ is an $\alpha$-terminal, the pair $(x_i, mate(x_i))$ is not contained in $M$. Hence, we have a demand of at least $|M|+1$ nets on the cut $(X, V \setminus X)$ which then is over-saturated in both $(G', S', T')$ and $(G, S, T)$.
By Lemma 3.9, $o_i$ and $o_{i+1}$ differ on a set of terminal connecting paths. Note that $x_i$ is an $\alpha$–terminal with respect to $o_i$ and a $\beta$–terminal with respect to $o_{i+1}$ (and vice versa for the mate of $x_i$). The other terminals do not change their $\alpha/\beta$–status. Thus the difference between $o_i$ and $o_{i+1}$ exactly consists of the edges on one $x_i$-$x_j$-path, say $P$.
Now assume the existence of a $x_i$–$x_j$–path $Q$ that does not run completely on the right side of $P$ or on $P$ itself. Since $Q$ cannot cross itself, there is a minimal subpath $Q_{[v,w]}$ of $Q$ which lies completely on the left side of $P$ such that $v$ lies before $w$ on $P$. Then $Q_{[v,w]}$ together with $P_{[v,w]}$ forms a clockwise cycle in $G'_{i+1}$, contradicting property (O2). □

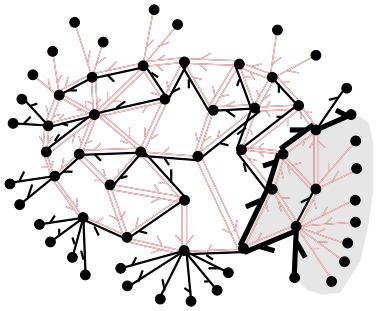

Fig. 13. The edges of the primal forest $F_i$ (black) for the weighting $d_i$ of Figure 11 and the corresponding orientation $O_i$ as shown in Figure 12. In this example, $F_i$ consists of one non–trivial forest and isolated terminals. The thick black lines form the path from $x_i$ to its mate as predicted by Theorem 3.11. The grey area indicates the right part of this path.

### 3.3 Complexity

Let $Q_i$ be the path characterized in Theorem 3.11 for $i \in \{0, \ldots, K-1\}$. We denote by $\Delta_{ij} = E(G'_i)\Delta E(G'_j)$ the set of edges on which $o_i$ and $o_j$ differ.
Now, it remains to show how to construct all paths $Q_i$ in linear time in total. To that end, we maintain a leftmost branching $F_j$ in the sense of Subsection 2.2. Procedure 1 shows how to construct such a branching from scratch. This is done only for $F_0$. The branchings $F_j$ for $j = 1, ..., K-1$ are obtained by modifying $F_0$, as we will see later.

By Properties (O1) and (O2) and Proposition 2.2 we have

COROLLARY 3.12. *$G'_i$ contains no clockwise cycles and $G^*_i$ contains no cycles at all. Thus, the leftmost spanning branching in $G'_i$ and the rightmost spanning branching in $G^*_i$ exist.*

Note that in Subsection 2.2 we always consider the case $i = 0$. When applying that subsection's results here we have to modify them to allow for variable $i$.

LEMMA 3.13. *If for a vertex $v \in W(G)$ the incoming edge in $F_i$ is changed by the transition from $F_i$ to $F_{i+1}$, i.e. $\phi_i(v) \neq \phi_{i+1}(v)$, then the leftmost path to $v$ in $G_i$ begins at $x_i$.*

PROOF. If the leftmost path $P$ to $v$ in $G_i$ begins at $x_k$ and $k > i \ (mod\ K)$ then $v$ is not reachable from $x_i, x_{i+1}, ..., x_{k-1}$ in $G'_i$ and $V(P) \cap V(Q_i) = \emptyset$. By Theorem 3.11, $E(G'_i)$ and $E(G'_{i+1})$ differ exactly on $Q_i$. Hence $v$ is not reachable from $x_{i+1}, x_{i+2}, ..., x_{k-1}$ in $G'_{i+1}$. Therefore the leftmost path to $v$ in $G'_{i+1}$ originates at $x_k$. Since $P$ and $Q_i$ are disjoint, $P$ remains unchanged and is still the leftmost $x_k$–$v$–path in $G'_{i+1}$. Thus the branching's incoming edge at $v$ is not changed and we have $\phi_i(v) = \phi_{i+1}(v)$. □

We denote by $F_i^v$ the arborescence belonging to $F_i$ that is rooted at $v \in V(F_i)$. If $v = x_k$ then we write $F_{i,k} := F_i^{x_k}$. Due to Lemma 3.13, the set of vertices for which the incoming edge in $F_{i+1}$ differs from that in $F_i$ is a subset of $F_{i,i}$. In particular:

COROLLARY 3.14. *If $v \in V(F_{i,k})$ then $v \in V(F_{m,k})$ for all $m = i, i+1, ..., k$.*

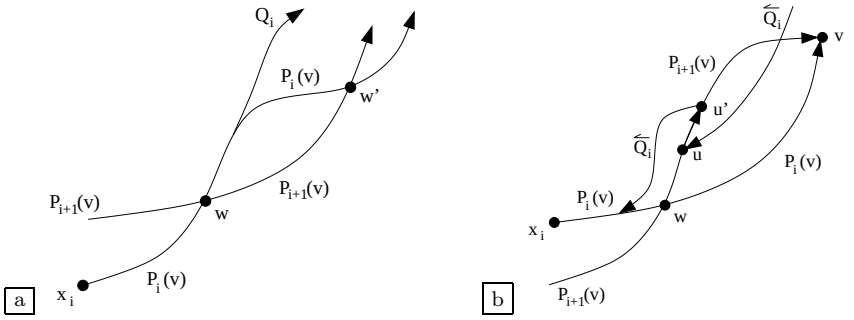


Fig. 14. Examples for the proof of Lemma 3.16.

Now follow some technical results which describe properties of the transition from branching $F_i$ to $F_{i+1}$.

LEMMA 3.15. *For no vertex $v \in V(Q_i)$ the edge $\phi_{i+1}(v)$ is incident to $Q_i$ from the left.*

PROOF. Assume that for a vertex $v \in V(Q_i)$ the edge $\phi_{i+1}(v)$ is incident to $Q_i$ from the left. Since $v$ is reachable from $mate(x_i)$ in $G'_{i+1}$, the start vertex of $P_{i+1}(v)$ lies on $Q_i$ or on its right side and thus $P_{i+1}(v) - v$ is not vertex disjoint to $Q_i$. Let $w$ be the vertex where $P_{i+1}(v)$ first meets $Q_i$. If $v$ lies in front of $w$ on $Q_i$, then going along $\overleftarrow{Q_i}$ is more to the left than using $\phi_{i+1}(v)$. On the other hand, if $v$ lies behind $w$ on $Q_i$, then $Q_i$ cannot be leftmost in $G'_i$, since a subpath of $P_{i+1}(v)$ bypasses it on its left side. □

LEMMA 3.16. *The path $P_i(v)$ does not cross $P_{i+1}(v)$.*

PROOF. If $v \notin V(F_{i,i})$ then $P_{i+1}(v) = P_i(v)$. Otherwise assume first that $P_i(v)$ crosses $P_{i+1}(v)$ from right to left. For the following argument refer to Figure 14a. Then there is a maximal subpath $P_i(v)|_{[w,w']}$ vertex disjoint to the right side of $P_{i+1}(v)$ such that $w, w' \in V(P_i(v)) \cap V(P_{i+1}(v))$. Since $P_{i+1}(v)$ is leftmost in $G'_{i+1}$, $P_i(v)$ cannot exist in $G'_{i+1}$ and we have $\emptyset \neq E(Q_i) \cap E(P_i(v)|_{[w,w']}) \subset E(Q_i) \cap E(P_i(v))$. Now, $E(Q_i) \cap E(P_i(v))$ are the edges of a (connected) path that begins at $x_i$. Therefore, $w \in V(Q_i) \cap V(P_i(v))$ and thus the predecessor of $w$ on $P_{i+1}(v)$ lies on the left side of $Q_i$, contradicting Lemma 3.15.

If $P_i(v)$ does cross $P_{i+1}(v)$ from left to right, then $P_{i+1}(v)$ cannot exist in $G'_i$, since $P_i(v)$ is leftmost. Hence we have $E(P_{i+1}(v)) \cap E(\overleftarrow{Q_i}) \neq \emptyset$. Let $(u, u')$ be an edge on $P_{i+1}(v)$ that lies on $\overleftarrow{Q_i}$. Since $(Q_i)_{[x_i,u']}$ and $(Q_i)_{[u,x_j]}$ must be vertex disjoint for the simple path $Q_i$, at least one vertex of $P_{i+1}(v)$ must lie on the left side of $Q_i$, contradicting Lemma 3.15. (For an example see Figure 14b.) □

Can there be $i < j$ with $|i - j| > 1$ such that $P_i(v)$ crosses $P_j(v)$? If there were, then all paths $P_k(v)$ for $i < k < j$ would have to cross either $P_i(v)$ or $P_j(v)$. In particular, $P_{i+1}(v)$ and $P_{j-1}(v)$ would have to do so, contradicting Lemma 3.16. Hence we have

COROLLARY 3.17. *$P_i(v)$ does not cross $P_j(v)$ for any $i$ and $j$.*

LEMMA 3.18. *If $v \in V(F_{i,i}) \setminus V(Q_i)$ and $P_{i+1}(v)$ is not vertex disjoint to $Q_i$ then $\phi_{i+1}(v) = \phi_i(v)$.*

PROOF. Let $w \in V(Q_i)$ be the last vertex on $P_i(v)$ that lies on $Q_i$ and $w' \in V(Q_i)$ be the last vertex on $P_{i+1}(v)$ that lies on $Q_i$. Since $P_{i+1}(v)$ is leftmost and $Q_i$ has been reversed, $w'$ cannot lie after $w$ on $Q_i$. Nor can $w'$ lie before $w$ on $Q_i$, because $P_i(v)$ is leftmost in $G'_i$. Hence, $w = w'$ and $P_{i+1}(v)$ leaves $Q_i$ at the same vertex where $P_i(v)$ leaves $Q_i$. Thus, the leftmost path from $w$ to $v$ is $(P_i(v))_{[w,v]}$ in both $G'_i$ and $G'_{i+1}$ since it is edge disjoint to $Q_i$. □

From now on let $j$ be the index such that $x_j = \text{mate}(x_i)$. Our considerations are much simplified by

LEMMA 3.19. *For all vertices $v$ on the left side of $Q_i$ we have $\phi_{i+1}(v) = \phi_i(v)$.*

PROOF. By Lemma 3.13 we need only consider vertices $v$ whose leftmost path $P_i(v)$ in $G'_i$ begins at $x_i$. After the reversal of $Q_i$, $v$ is reachable from $x_j := \text{mate}(x_i)$. Therefore, the leftmost path $P_{i+1}(v)$ to $v$ in $G'_{i+1}$ originates at $x_j$ or at a terminal on the right side of $Q_i$, and so is not vertex disjoint to $Q_i$. Then Lemma 3.18 completes the proof. □

The following result then gives a first idea how the branching $F_{i+1}$ might be obtained from $F_i$.

THEOREM 3.20. *If $\phi_{i+1}(v) \neq \phi_i(v)$ for a vertex $v \in V(G)$, then $\phi_{i+1}(v)$ is the counter-clockwise next incoming edge after $\phi_i(v)$ such that its tail is reachable from $\{x_{i+1}, ..., x_{j-2}, x_{j-1}\}$ in $G'_{i+1}$. If no such edge exists, then $\phi_{i+1}(v) = \phi_i(v)$.*

PROOF. Let $(w, v)$ be the counter-clockwise next incident edge pointing to $v$ after $\phi_i(v)$, such that $w$ is reachable from $\{x_{i+1}, ..., x_{j-2}, x_{j-1}\}$ in $G'_{i+1}$. Consider the leftmost terminal $x_k$ from which $w$ is reachable, and the leftmost $x_k$–$w$–path $P$ in $G'_{i+1}$. By construction, $P$ is leftmost among all paths to $v$ that contain $(w, v)$ as their last edge. If there were a path entering $v$ by an edge in the counter-clockwise interval $\{\phi_i(v), ..., \phi_{i+1}(v)\}$ of $v$'s incidence list then it could not begin at a terminal to the left of $x_k$. Any path entering $v$ by another edge would have to cross $P$. It cannot cross $P_i(v)$ due to Lemma 3.16. Hence, $P$ is the leftmost path to $v$

If $v$ is not reachable from $\{x_{i+1}, ..., x_{j-2}, x_{j-1}\}$ in $G'_{i+1}$, then $P_{i+1}(v)$ begins at $x_j$ and thus is not vertex disjoint to $Q_i$. In that case Lemma 3.18 shows that $\phi_{i+1}(v) = \phi_i(v)$. □

The following Procedure 2 determines $F_{i+1}$ from $F_i$ and at the same time reverses the path $Q_i$. We use a modified right-first depth-first search beginning at $x_i$. At each vertex we branch in counter-clockwise order over all outgoing edges but only up to the first incoming edge. The latter becomes the new branching edge of the examined vertex. By $ccnext_{i+1}(v, e)$ we mean the edge $ccnext(v, e)$ oriented according to $o_{i+1}$. This makes a difference for $e \in E(Q_i)$, since these edges are reversed by the procedure.

Each vertex may be *labeled* either temporarily or permanently. We say a vertex is labeled, if it is labeled in either way. At the start of the oracle's execution all labels are removed. Temporary labels are removed after the end of Procedure 2. Whereas permanent labels stand till the end of the calling algorithm's execution.

**Procedure 2** Construct $F_{i+1}$ from $F_i$

**Input:** The graph $G'_i$ and the branching $F_i$ given by $\phi_i$.
**Output:** The orientation $o_{i+1}$ and the branching $F_{i+1}$ given by $\phi_{i+1}$.

1: Set $\phi_{i+1} := \phi_i$, $o_{i+1} = o_i$ and the stack $A := \emptyset$.
2: Determine $Q_i$ by backtracking in $F_i$ beginning at $x_j := \text{mate}(x_i)$.
3: Let $v$ be the unique successor of $x_i$ and $e = (x_i, v) \in E(Q_i)$.
4: Reverse the edge $e$, i.e. $o_{i+1}(\{x_i, v\}) := (v, x_i)$.
5: **while** $v \neq x_j$ **do**
6: Label $v$ temporarily and let $w$ be the immediate successor of $v$ on $Q_i$.
7: Reverse the edge $(v, w) \in E(Q_i)$, i.e. $o_{i+1}(\{v, w\}) := (w, v)$.
8: Set $e := \text{ccnext}_{i+1}(v, e)$ and label $v$ temporarily.
9: Do push$(A, e)$.
10: **while** $A$ not empty **do**
11: **if** $e \in \delta^-(v)$ **then**
12: **if** $e = \phi_i(v)$ **then** label $v$ permanently.
13: Set $\phi_{i+1}(v) := e$.
14: Set $e := \text{pop}(A)$ and $v := \text{tail}(e)$.
15: **else**
16: **if** head$(e)$ is not labeled **then**
17: Do push$(A, e)$ and set $v := \text{head}(e)$ and $e := \text{ccnext}_{i+1}(v, e)$ and label $v$ temporarily.
18: **end if**
19: **end if**
20: Set $e := \text{ccnext}_{i+1}(v, e)$.
21: **end while**
22: Set $e := (w, v) \in E(Q_i)$ and $v := w$.
23: **end while**

The proof that Procedure 2 is correct and efficient takes up the rest of this subsection. It also completes the proof of the oracle's linear complexity, since we are now able to maintain the orientations $o_i$ efficiently.

THEOREM 3.21. *Procedure 2 correctly determines $F_{i+1}$ from $F_i$ and the aggregate running time of all iterations is linear.*

We begin with some definitions and lemmata that lead up to the final proof of the previous theorem.

For each labeled vertex $v$ denote by $\rho(v)$ the edge $e \in E(G'_i)$ by which $v$ is entered in line 6 or 17 of Procedure 2. Let $R(v)$ be the path with vertices $(v, \text{tail}(\rho(v)), \text{tail}(\rho(\text{tail}(\rho(v)))), ..., x_i)$, i.e. the path found by backtracking along the edges $\rho(\cdot)$ in $G'_i$.

3.3.1 *Intervals of incidence lists.* The interval of edges in $v$'s incidence list that lie counter-clockwise after an edge $e$ and before an edge $f$ (both included) will be denoted by $[e, f]_v$. Analogously, we write $[v, w]_{G'}$ for the set of all terminals that lie counter-clockwise after the terminal $v$ and before the terminal $w$ on the boundary of $G'$. As usual we use parentheses instead of brackets to indicate open or half-open intervals. We define $[e, e]_v := \{e\}$ and $[v, v]_{G'} := \{v\}$.

LEMMA 3.22. *The start terminal $b_i(v)$ of the leftmost path $P_i(v)$ to $v$ in iteration $i$ revolves once in counter-clockwise direction around the boundary of $G'$ during the execution of the oracle, i.e. $b_{i+1}(v) \in [b_i(v), b_K(v)]_{G'}$ for all $0 < i < K$.*

PROOF. By Lemma 3.13, $b_i(v)$ advances in an iteration $i$ only if $b_i(v) = x_i$. In particular, if it advanced further than $b_0(v)$ in any but the "zeroth" iteration, then it would stay there at least until iteration $K$. Which is impossible, since $b_K(v) = b_0(v)$. Therefore, we have $b_{i+1}(v) \in [b_i(v), b_K(v)]_{G'}$ for all $0 < i < K$. In particular, $b_i(v)$ advances in counter-clockwise direction. □

The previous lemma together with Corollary 3.17 then shows

LEMMA 3.23. *For each $v \in V(G')$ the incoming branching edge revolves about $v$ at most once in counter-clockwise direction.*

LEMMA 3.24. *Consider iterations $k$ and $\ell$ and a vertex $v \in V(G')$. Then $b_i(v) \in [b_k(v), b_\ell(v)]_{G'}$ for all iterations $k \le i \le \ell \mod K$.*

PROOF. If $b_i(v)$ is constant for $i$ in the given interval of iterations, i.e. $b_k(v) = b_{k+1}(v) = ... = b_\ell(v)$ then the assertion is trivial. Otherwise, we have $x_\ell \in [b_k(v), b_\ell(v)]_{G'}$. (Recall that Lemma 3.13 shows $b_{i+1}(v) \neq b_i(v) \Rightarrow b_i(v) = x_i$.) By the proof of Lemma 3.22, $b_{i+1}(v)$ lies in $[b_i(v), b_K(v)]_{G'}$ for all $k \le i < K \mod K$. If $b_i(v)$ lay outside the interval $[b_k(v), x_\ell]_{G'}$ for an iteration $i$ with $k \le i \le \ell \mod K$, then $b_\ell(v) = b_i(v)$, by Lemma 3.13. Thus, $b_i(v)$ lies in $[b_k(v), b_\ell(v)]_{G'}$ for iterations $k \le i \le \ell \mod K$. □

The previous result has shown that the function $i \mapsto b_i(v)$ is monotonous, in a sense. Corollary 3.17 then yields

COROLLARY 3.25. *Consider iterations $k$ and $\ell$ and a vertex $v \in V(G')$. Then $\phi_i(v) \in [\phi_k(v), \phi_\ell(v)]_v$ for all iterations $i$ with $b_i(v) \in (b_k(v), b_\ell(v))_{G'}$.*

LEMMA 3.26. *Consider a vertex $v$ that is being labeled in line 6 or 17 of Procedure 2.*

*(a)* $\forall\ (v,u) \in [\phi_{i+1}(v),\ \phi_i(v)]_v :\ v \in V(P_i(u)) \implies F^u_{i+1} = F^u_i$.

*(b)* $\forall\ (v,u) \in [\rho(v),\ \phi_{i+1}(v)]_v :$
$\phi_{i+1}(v) \neq \phi_i(v)$ *and* $u$ *not yet labeled* $\implies \phi_{i+1}(u) \neq \phi_i(u)$.

*(c)* $\forall\ u \in V(R(v)) \setminus \{v\} :$ *no incoming edge is incident to $u$ from the right side of $R(v)$.*

*(d)* $\forall\ (u,v) \in [\phi_i(v),\ \rho(v)]_v :\ u \notin P_i(v) \implies u \notin P_{i+1}(v)$.

*(e)* $\forall\ (v,u) \in [\phi_i(v),\ \rho(v)]_v :\ F^u_{i+1} = F^u_i$.

PROOF. (a): Consider $w \in V(F^u_i)$. Then $v \in V(P_i(u))$ implies $V(P_i(v)) \subset V(P_i(w))$. Assume that $P_{i+1}(w)$ is more to the left than $P := P_{i+1}(v) + (P_i(w) - P_i(v))$. Then, the former has at least one vertex in the region enclosed in counter-clockwise direction by $P_{i+1}(v)$, $\overleftarrow{P_i(v)}$ and the boundary of $G'$. Since $P_{i+1}(w)$ cannot cross $P_{i+1}(v)$, it must cross $P_i(v)$. But that contradicts Lemma 3.16, because $V(P_i(v)) \subset V(P_i(w))$.
(b): As $u$ is not labeled, $R(v) + (v,u)$ is the rightmost $x_i$–$u$–path in $G'_{i+1}$. Since $G$ is 3–edge–connected, there has to be a path from $x_\ell \in \{x_{i+1}, ..., x_{k-1}, x_k\}$ to $u$ that

does not contain $v$. Here, $x_k$ is the terminal at which $P_{i+1}(v)$ starts. Moreover, $\phi_{i+1}(v) \neq \phi_i(v)$ implies $k \neq j$. Since $R(v)$ is rightmost and $P_{i+1}(v)$ is leftmost, we have $v \in P_i(u)$ and thus $\phi_{i+1}(u) \neq \phi_i(u)$.
(c): Otherwise, let $e$ be such an edge and $u'$ the immediate successor of $u$ on $R(v)$. Then we have $e \in [\rho(u), (u, u')]$. Therefore, Procedure 2 would have stopped scanning the incidence list of $u$ at the edge $e$ in line 11 and $(u, u') \notin E(R(v))$.
(d): Assume that $u \in P_{i+1}(v)$. Now, $u$ lies in a region enclosed by $P_i(v)$ and $R(v)$. By (c), $P_{i+1}(v)$ cannot cross $R(v)$ from right to left. Lemma 3.16 then yields the contradiction, because $P_{i+1}(v)$ crosses $P_i(v)$.
(e): Let $w \in V(F_i^u)$. Again, $u$ lies in a region enclosed by $P_i(v)$ and $R(v)$. By (c), $P_{i+1}(w)$ cannot cross $R(v)$ from right to left. By Lemma 3.16, it cannot cross $P_i(v)$, because $V(P_i(v)) \subset V(P_i(w))$. □

LEMMA 3.27. *If $\phi_{i+1}(v) = \phi_i(v)$ for a vertex $v \in V(F_{i,i})$ that is labeled by Procedure 2, then $\phi_k(v) = \phi_i(v)$ for all iterations $k$ after $i$.*

PROOF. Let $u$ be the first vertex on $P_i(v)$ that lies on $P_{i+1}(w)$. Since $Q_i$ is reversed, $u$ is no terminal. Thus, we have $\phi_{i+1}(u) \neq \phi_i(u)$. Corollary 3.25 then yields $\phi_k(v) = \phi_i(v)$ for all $k$ with $b_k(v) \in (b_{i+1}(v), b_i(v))_{G'}$. That is, $\phi_i(v)$ is not changed during the rest of the oracle's execution. □

PROOF OF THEOREM 3.21. Lemma 3.26(a) and (e) imply that we traverse all vertices which need to be changed. Assertion (d) of that lemma shows that we make no mistake by not scanning the interval $[\phi_i(v), \rho(v)]_v$ of $v$'s incidence list. Thus, the procedure works correctly.

All edges tested in line 16 of the procedure lie in the subgraph induced by the set of labeled vertices. This graph is planar and therefore, the number of tested edges is, asymptotically, as large as the number of vertices in the labeled subgraph. Hence, testing already labeled vertices does not increase the asymptotical running time.

This running time is linear, because in each iteration and for each vertex $v$ labeled in this iteration we advance $\phi_i(v)$ or label $v$ permanently. And the number of advances of $\phi_i(v)$ is smaller or equal to $|\delta(v)|$, by Lemma 3.23. □

## 4. THE MAIN ALGORITHM

### 4.1 Auxilliary instances

We first introduce two auxiliary instances for a saturated $W$–cut $(X, Y)$. The first one, $(G'(X), S'(X), T'(X))$ is shown in Figure 15. The vertex set of $G'(X)$ is constructed from $X$ by adding a new vertex $r$ and all terminals in $Y$ that have their mates in $X$. The edge set of $G'(X)$ is constructed from $E \cap (X \times X)$ as follows: for every edge $(v, w) \in E(X, Y)$, $v \in X$, an edge $(v, r)$ is inserted (multiple edges allowed), and every terminal taken from $Y$ is connected with $r$ by a new edge. The relative order of these terminals around $r$ is strictly taken over from $(G, S, T)$.

The graph $G''(X)$ depicted in Figure 16 is defined on $Y$ instead of $X$ and quite mirror-symmetrically to $G'(X)$. The only exception is that no "bundle vertex" $r$ is introduced, but the edges inserted for $E(X, Y)$ are immediately connected with new vertices of degree one. We will consider an instance $(G''(X), S''(X, P'), T''(X, P'))$, which not only depends on $G$, $X$, $S$, and $T$, but also on a solution $P'$ to $(G'(X), S'(X), T'(X))$.

More specifically, for an edge $e = \{v, w\} \in E(X, Y)$ in the original graph $G$, $v \in X$, let $e_1 = \{v, r\}$ be the corresponding edge in $G'(X)$ and $e_2$ the corresponding edge in $G''(X)$. Then the path label of $e_1$ with respect to $P'$ is taken over as the number of the terminal incident to $e_2$ in $G''(X)$.[4] The terminals in $Y$ are taken over as they are. This constitutes $S''(X, P')$ and $T''(X, P')$.

We call $(G''(X), S''(X, P'), T''(X, P'))$ an *R–instance*, because in our figures it is (almost) always to the right of the other auxilliary instance $(G'(X), S'(X), T'(X))$ which we refer to as an *L–instance*, accordingly. These names do not refer to the notation used for the instance but to the way this instance was derived. For example, if we consider a vertex of the decomposition tree that is not the root then the partial instance $(G, S, T)$ corresponding to that vertex can be an L–instance. Finally, we say $(G, S, T)$ is an *$\mathcal{L}$–instance* if at least one predecessor in the decomposition tree is an L–instance. In other words, $(G, S, T)$ is an $\mathcal{L}$–instance if and only if it has a bundle vertex $r$. (Note, that no partial instance can have more than one bundle vertex.)

Obviously, if the instance $(G, S, T)$ is solvable, then $(G'(X), S'(X), T'(X))$ is also solvable. Moreover, a solution $P'$ to $(G'(X), S'(X), T'(X))$ and a solution $P''$ to $(G''(X), S''(X, P'), T''(X, P'))$ can be trivially combined into a solution to the original instance $(G, S, T)$.

The following are corollaries of Lemma 3.9. They show that the oracle gives the correct results if it is applied to partial instances.

COROLLARY 4.1. *Let $(G, S, T)$ be an instance, $e \in E$ a boundary edge, $y_i$ the dual vertex in $P^*$ incident to $e$, and $(X, Y)$ a candidate cut of $e$. When the orientation $o_i$ of all edges of $G$ are transferred to the corresponding edges in $(G'(X), S'(X), T'(X))$ and the edges incident to the terminals taken from $Y$ are oriented towards these terminals, then the result is the orientation of $(G'(X), S'(X), T'(X))$ with respect to $y_i$.*

[4]Since $(X, Y)$ is saturated in $(G, S, T)$, the corresponding cut in $(G'(X), S'(X), T'(X))$ is also saturated. Hence, there are as many as $E(X, Y)$ terminals in $X$ that have their mates in $Y$.

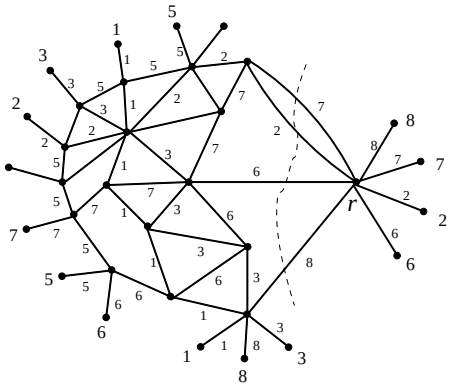


Fig. 15. The auxiliary instance $(G'(X), S'(X), T'(X))$ for the instance $(G, S, T)$ and the rightmost saturated $W$–cut $(X, Y)$ in Figure 3. The edge labels indicate the restriction of the solution in Figure 3 to $(G'(X), S'(X), T'(X))$, which obviously is a solution to $(G'(X), S'(X), T'(X))$.

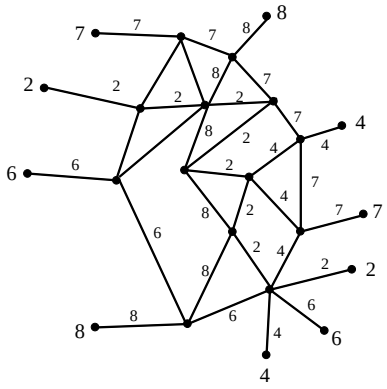

Fig. 16. The instance $(G''(X), S''(X, \bar{P}), T''(X, \bar{P}))$ for the original instance $(G, S, T)$ from Figure 3, for the same saturated $W$–cut $(X, Y)$ as in Figure 15, and for the solution $\bar{p}$ to $(G'(X), S'(X), T'(X))$ shown in Figure 15. This figure demonstrates that the restriction to $(G''(X), S''(X, P), T''(X, P))$ of a solution to $(G, S, T)$ is indeed a solution to $(G''(X), S''(X, P), T''(X, P))$ (cf. Figure 3).

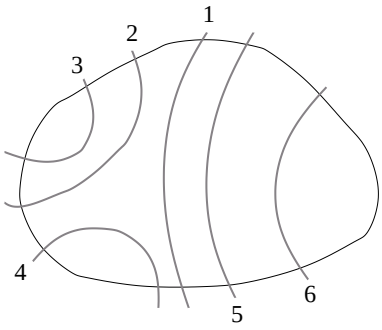


Fig. 17. Outline of an instance. The numbers indicate the edges for which a relevant $W$–cut (grey) exists in the order in which theses cuts have been found. Here, the algorithm started at the edge immediately counter-clockwise before the edge labeled "1".

COROLLARY 4.2. *Let $(G, S, T)$, $e$, $y_i$, and $(X, Y)$ be defined as in Corollary 4.1 and let $P'$ be an arbitrary solution to $(G'(X), S'(X), T'(X))$. When the orientations of all edges of $G$ are transferred to the corresponding edges of $G''(X)$, then the result is the orientation of $(G''(X), S''(X, P'), T''(X, P'))$ with respect to $y_i$.*

## 4.2 Outline of the algorithm

In a given instance $(G, S, T)$ the algorithm tries to find a saturated $W$-cut by asking the oracle (ll. 1-10). If such a cut has been found (l. 11), the partial instance $(G'(X), S'(X), T'(X))$ is constructed and the algorithm solves this part of the orig-

inal instance recursively. Using the returned solution $P'$ to $(G'(X), S'(X), T'(X))$, the partial instance $(G''(X), S''(X, p'), T''(X, p'))$ is constructed and again solved recursively. Otherwise, if no saturated $W$-cut has been found (l. 29), the instance is extended to a full instance using a technique from [Becker and Mehlhorn 1986], and then solved by the algorithm from [Wagner and Weihe 1995].

The start and stop edge $e^*$ resp. $e^{**}$ for the original instance can be set arbitrarily, provided that $e^* = \mathrm{ccnext}(G, e^{**})$. The start and stop edge for each partial instance are chosen by the algorithm such that they delimit the section of the original graph's boundary that belongs to the partial instance. (Corollary 3.3 shows that relevant cuts cannot begin at an boundary edge of the partial instance that does not belong to the original instance's boundary.) Thus, the oracle is consulted exactly once for each boundary edge and the algorithm does this in counter-clockwise order beginning with the given start edge (cf. Figure 17). That way, the oracle guarantees linear running time in total.

**Algorithm 1** The Main Algorithm

**Input:** An instance $(G, S, T)$ a start edge $e^*$ and a stop edge $e^{**}$ both on the boundary.

**Output:** A solution $P$ for $(G, S, T)$ or the message "unsolvable."

Set $found\_a\_cut := false$.
**for all** $e$ on the boundary in counter-clockwise order from $e^*$ to $e^{**}$ **do**
  Call the oracle with parameter $e$.
  **if** the oracle returned "unsolvable" **then**
    **return** "unsolvable."
  **else if** the oracle returned a $W$-cut $(X, V \setminus X)$ **then**
    $found\_a\_cut := true$.
    **break loop**.
  **end if**
**end for**
**if** $found\_a\_cut = true$ **then**
  Let $e'$ be the cut edge on the boundary opposite from $e$.
  Construct the partial instance $(G'(X), S'(X), T'(X))$ and
    let $x$ be the counter-clockwise first terminal after ccnext$(G, e)$.
  Apply the algorithm recursively to $(G'(X), S'(X), T'(X))$ with
    start edge ccnext$(G, e)$ and stop edge $e'$.
  **if** the algorithm returned "unsolvable" **then**
    **return** "unsolvable."
  **else**
    Denote the returned solution by $P'$.
  **end if**
  Construct the other partial instance $(G''(X), S''(X, P'), T''(X, P'))$ and
    let $x$ be the counter-clockwise first terminal after $e$ in the partial instance.
  Apply the algorithm recursively to $(G''(X), S''(X, P'), T''(X, P'))$ with
    the start edge ccnext$(G, e')$ and stop edge $e$.
  **if** the algorithm returned "unsolvable" **then**
    **return** "unsolvable."
  **else**
    Denote the returned solution by $P''$.
  **end if**
  Combine the two partial solutions $P'$ and $P''$ into a solution $P$ to $(G, S, T)$.
**else**
  Construct the canonical extension $\overline{(G, S, T)}$ for $(G, S, T)$.
  Apply the algorithm from [Wagner and Weihe 1995] to $\overline{(G, S, T)}$ with start terminal $x$.
  **if** the algorithm returned "unsolvable" **then return** "unsolvable."
**end if**
**return** the solution $P$.

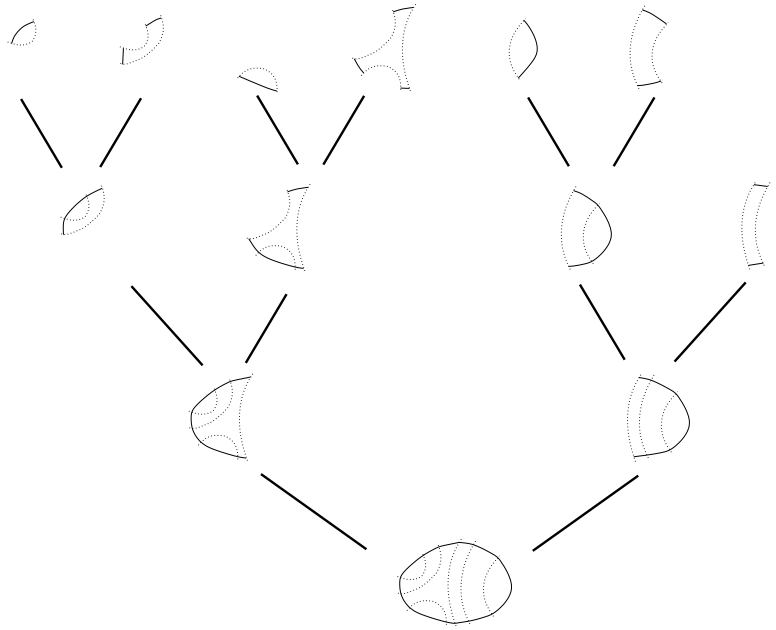

Fig. 18. The decomposition tree of the instance in Figure 17.

## 5. CORRECTNESS

We first state the main result. The rest of this section constitutes its proof.

THEOREM 5.1. *In linear time, the Main Algorithm (Algorithm 1) decides whether a given instance is solvable and finds a solution if it is.*

As outlined in Section 4 the algorithm partitions a given instance along a saturated $W$-cut and applies itself recursively to both parts. This defines a decomposition tree with the original instance at its root and partial instances without saturated $W$-cuts at its leaves (see Figure 18).

For this proof we use a top-down strategy, moving from the leaves of the decomposition tree down to the root.

### 5.1 Instances without saturated $W$-cuts

We partition the original instance along each relevant cut found by the oracle. So, in step 29 of the algorithm an instance $(G, S, T)$ without such cuts remains. We have then reached a leaf of our decomposition tree and do not divide the instance any further. Steps 29 through 31 are executed during this recursion of the algorithm. Note that, although the oracle only finds $W$–cuts w.r.t. the original instance, the leaves of the decomposition tree contain no gap separating saturated $W$–cuts whatsoever. That is, for each saturated $W$–cut $(X, Y)$ we have $P \cap X = \emptyset$ or $P \cap Y = \emptyset$.

We use the following technique from [Becker and Mehlhorn 1986] to create a full instance which can then be processed as shown in Section 5.2.

*Definition* 5.2. Let $(G,S,T)$ be an instance and $U \setminus (S \cup T) = \{u_1, ..., u_b\}$ its set of gaps where $u_i$ precedes $u_{i+1}$ in a counter-clockwise order. We enhance $(G,S,T)$ by nets $(u_{2i-1}, u_{2i})$ to obtain a full instance, the *canonical extension* $(G, S \oplus \{u_1, u_3, ..., u_{b-1}\},$
$T \oplus \{u_2, u_4, ..., u_b\})$ of $(G,S,T)$.

Since $b = |U \setminus (S \cup T)|$ is even in all legal instances such a pairing of gaps can always be found. Trivially, the construction takes time $O(b)$. Also from [Becker and Mehlhorn 1986], we adapt

LEMMA 5.3. *For a solvable instance $(G,S,T)$ without gap separating saturated $W$-cuts the canonical extension is solvable.*

PROOF. Let $P$ be an arbitrary solution to $(G,S,T)$ and delete all edges on paths $P_i$ and all terminals from $(G,S,T)$. The remaining instance is still legal and solvable, and a pairing of its gaps is also a pairing for the original instance. So, from now on, we assume $S = T = \emptyset$.

Let $(X, V \setminus X)$ be a $W$-cut in $G$. By definition of the term "$W$-cut", $E(X, V \setminus X)$ contains exactly two edges on the boundary of $G$. The construction of the canonical extension shows that in this case $(X, V \setminus X)$ separates at most two pairs of plugs, i.e. $\mathrm{dem}(X, V \setminus X) \leq 2$. Our prerequisite was that $\mathrm{supp}(X, V \setminus X) \geq 1$. Hence it suffices to show that $\mathrm{supp}(X, V \setminus X)$ is even, if $\mathrm{dem}(X, V \setminus X)$ is even.

Let $N_X$ be the number of nets $(u_{2i-1}, u_{2i})$ such that both $u_{2i-1}$ and $u_{2i}$ are in $X$. The number of edges in the subgraph induced by $X \cap W$ is

$$|E(X \cap W)| = \frac{1}{2}\Big( \sum_{v \in X \cap W} |\delta(v)| - \mathrm{dem}(X, V \setminus X) - \mathrm{supp}(X, V \setminus X) - 2N_X \Big),$$

and thus,

$$\mathrm{dem}(X, V \setminus X) + \mathrm{supp}(X, V \setminus X) = \sum_{v \in X \cap W} |\delta(v)| - 2\,|E(X \cap W)| - 2N_X$$

which is even, since all non-plug vertices have even degrees. Therefore $\mathrm{dem}(X, V \setminus X)$ and $\mathrm{supp}(X, V \setminus X)$ have the same parity. □

## 5.2 Full partial instances

In line 30 of our main algorithm (Procedure 1) we have a full instance $(G,S,T)$. This is in general a partial instance located between saturated cuts. We want to prove that given an ordering of the $S$-terminals, Wagner's and Weihe's algorithm finds an ordering of the $T$-terminals which is optimal in a sense. In order to do so we need the following properties of the algorithm from [Wagner and Weihe 1995]. For details see Section 3 in that paper.

5.2.1 *Wagner's and Weihe's algorithm.* Let $(G,S,T)$ be a solvable full instance and $x \in S \cup T$ a terminal. In a preprocessing step the algorithm determines a digraph $A(G,S,T,x)$ which contains all vertices of $G$. And for each edge $\{u,v\}$ in $G$ at most one of the arcs $(u,v)$ or $(v,u)$ is contained. The algorithm then uses right-first depth-first search in $A(G,S,T,x)$ to produce a solution $P$ consisting of one directed path $P_i$ from $s_i$ to $t_i$ in $A(G,S,T,x)$ for each net $(s_i,t_i)$ in counter-clockwise order beginning with the start vertex $x$. Obviously, this induces a solution for $(G,S,T)$.

In this section, whenever we use directed cuts or paths we mean the orientation induced by the paths of a solution by the algorithm from [Wagner and Weihe 1995].

In Section 3.3.1 we defined the intervals $[e,f]_v$ for $e,f \in \delta(v)$. In digraphs we distinguish $[e,f]_v^+ := [e,f]_v \cap \delta^+(v)$ and $[e,f]_v^- := [e,f]_v \cap \delta^-(v)$. In order to formulate some properties we need the following notation from [Wagner and Weihe 1995]: We denote by $\mathcal{D}_v[e,f]$ the number of edges in $[e,f]_v \cap E(A(G,S,T,x))$ entering $v$ minus the number of edges leaving $v$, i.e.

$$\mathcal{D}_v[e,f] = |[e,f]_v^- \cap E(A(G,S,T,x))| - |[e,f]_v^+ \cap E(A(G,S,T,x))|.$$

We define a path label $\lambda : E(A(G,S,T,x)) \to \{1,...,|S|\}$. For each edge $e$ the label $\lambda(e)$ gives the index of the path that uses $e$.

PROPERTIES 5.4. *The relevant properties of the algorithm from [Wagner and Weihe 1995] are these:*

*(i) Each time $P_i$ and $P_j$ with $i<j$ cross each other, $P_j$ crosses $P_i$ from right to left. In particular, solution paths cross each other at most once.*

*(ii) In linear time, either a solution is constructed or a certificate for unsolvability is returned.*

*(iii) If we remove solution paths and their corresponding nets from an instance, then the solution to the reduced instance is the same as the restriction of the original solution to the thus reduced instance. If the start vertex $x$ is among the removed terminals, then the start vertex of the restricted instance must be the counter-clockwise next terminal after $x$ that has not been removed.*

*(iv) In the digraph $A(G,S,T,x)$ we have $|\delta^-(v)| = |\delta^+(v)|$ at each vertex $v \in W$.*

*(v) The next is a consequence of the algorithm's right-first search strategy: Consider edges $e,f \in E(P_i)$ and a vertex $v$ such that $e \in \delta^-(v)$ and $f \in \delta^+(v)$. Then $\lambda(g) \leq i$ holds for all $g \in [e,f]_v^+$.*

*(vi) For a vertex $v$ and $e,f \in \delta(v)$ such that $f \notin E(A(G,S,T,x))$ we have $\mathcal{D}_v[e,f] \leq 0$.*

*(vii) $E(A(G,S,T,x)) = E(P_1) \cup E(P_2) \cup ... \cup E(P_{|S|})$.*

These properties have the following implications:

LEMMA 5.5. *Consider $v \in V(G)$.*

*(a) Let $e,f \in \delta(v) \setminus E(A(G,S,T,x))$ such that all edges in $(e,f)_v$ belong to the digraph $A(G,S,T,x)$. Then $\mathcal{D}_v[e,f] = 0$ and $|[e,f]_v|$ is even.*

*(b) For each $e \in \delta^+(v) \cap E(A(G,S,T,x))$ there exists an $f \in \delta^-(v) \cap E(A(G,S,T,x))$ such that $[f,e]_v \subset E(A(G,S,T,x))$.*

PROOF. (a): The case $\delta(v) \subset E(A(G,S,T,x))$ is trivial. In all other cases, we break $\delta(v)$ into intervals of edges $(e_1,e_2)_v, ..., (e_{2k-1},e_{2k})_v$ such that each interval is completely contained in $A(G,S,T,x)$ but the delimiting edges are not. Then, by property (iv),

$$\begin{aligned} 0 &= |\delta^-(v) \cap E(A(G,S,T,x))| - |\delta^+(v) \cap E(A(G,S,T,x))| \\ &= \sum_{i=1}^{k} \mathcal{D}_v[e_{2i-1}, e_{2i}] \end{aligned}$$

Property (vi) then implies $\mathcal{D}_v[e_{2i-1}, e_{2i}] = 0$ for all $i$. That is, the number of incoming edges equals the number of outgoing edges in the given interval. Thus $|[e, f]_v|$ is even.
(b): Again, we can disregard the case $\delta(v) \subset E(A(G, S, T, x))$. We show that for each interval of edges in $\delta(v)$ not belonging to $A(G, S, T, x)$ the edge immediately counter-clockwise before the interval enters $v$. This edge can serve as the edge $f$ from the assertion for any edge in that interval. Let $e_1 \in \delta(v) \setminus E(A(G, S, T, x))$ such that the counter-clockwise previous edge $e_0$ in the incidence list of $v$ belongs to $A(G, S, T, x)$. That implies $[e_0, e_1]_v = \delta(v)$ and thus $\mathcal{D}_v[e_0, e_1] = 0$. Now, we consider the interval $(e_0, e_1]_v = [e_0, e_1]_v \setminus \{e_0\}$. Obviously, $\mathcal{D}_v(e_0, e_1] = \pm 1$. Property (vi) then yields $\mathcal{D}_v[e_0, e_1] = -1$. Hence, we have $e_0 \in \delta^-(v)$. □

5.2.2 *Net mappings.* To facilitate working with reordered terminals we introduce an extended notation of partial instances $(G, \tilde{S}, T, \sigma, \tau)$. For this let the terminals $\tilde{s}_i$ be numbered clockwise around the boundary of $G$ with $t_1$ preceding $\tilde{s}_1$ immediately and let $S_0$ (resp. $T_0$) be the set $S$ (resp. $T$) in the original instance. Recall that the terminals $t_i$ are ordered counter-clockwise around the boundary (Section 2.1.1). We define bijective *net mappings* $\sigma : M \to \{1, ..., |S|\}$ and $\tau : M \to \{1, ..., |T|\}$ with $M \subset \{1, ..., |S_0|\}$ such that $(\tilde{s}_{\sigma(i)}, t_{\tau(i)})$ is a net in the partial instance. Since the $t_i$ are ordered counter-clockwise in the original instance, $\tau$ always increases monotonously.

*Note:* When applying the algorithm from [Wagner and Weihe 1995] to such an extended instance, we have to renumber the terminals because that algorithm does not know about our net mappings and we do not want to modify it here. In detail, we apply the algorithm to the instance $(G, \overline{S}, \overline{T})$ with $\overline{S} = \{\bar{s}_1, ..., \bar{s}_{|S|}\}$, $\overline{T} = \{\bar{t}_1, ..., \bar{t}_{|T|}\}$ and for all $j$ we set $\bar{t}_j := t_j$ and $\bar{s}_j := \tilde{s}_{\sigma(\tau^{-1}(j))}$.

5.2.3 *Induced net mappings.* In an $\mathcal{L}$–instance $(G, S, T)$ (defined in Section 4.1) the edges which are incident to the bundle vertex $r$ but to no $t'_i$ we call $e_i = \{u_i, r\}$, and number these edges clockwise around $r$ with $e_1$ succeeding $\{r, t'_1\}$ immediately. These are the edges crossed by the cut in Figure 15. The *induced net mapping* $\theta$ of a solution $P$ is such that the path $P_i$ contains $e_{\theta(i)}$.

5.2.4 *Unbundled $\mathcal{L}$–instances.* Whether an $\mathcal{L}$–instance is solvable or not, does not depend on the induced net mapping. Therefore, we introduce an "unbundled" $\mathcal{L}$–instance $(G_u, S, T_u)$ which arises from $(G, S, T)$ by removing the bundle vertex $r$ and its incident edges and replacing each edge $e_i = \{u_i, r\}$ by $\{u_i, t'_i\}$. With this new instance type it makes sense to differentiate between solvable and unsolvable.

5.2.5 *Intersections of mappings.* Let $\alpha, \beta : M \to N$ be two bijective mappings on the same set $M \subset \mathbf{Z}_{>0}$ onto $N \subset \mathbf{Z}_{>0}$. We denote by $\mathrm{dom}(\alpha)$ the domain of $\alpha$ and by $\mathrm{im}(\alpha)$ the range of $\alpha$, i.e. $\mathrm{im}(\alpha) = \alpha(\mathrm{dom}(\alpha))$. We then define the $(k, \ell)$-*intersection* of the pair of functions $(\beta, \alpha)$ as

$$\begin{aligned} C^k_\ell(\beta, \alpha) &:= \{i \in M : \beta(i) \le k \text{ and } \alpha(i) \le \ell\} \\ &= \beta^{-1}(N \cap \{1, ..., k\}) \cap \alpha^{-1}(N \cap \{1, ..., \ell\}). \end{aligned}$$

Here, we are only interested in the case in which $\beta = id$. We write $C^k_\ell(\alpha) := C^k_\ell(id, \alpha) = \{i \in M : i \le k \text{ and } \alpha(i) \le \ell\} = \{1, ..., k\} \cap \alpha^{-1}(N \cap \{1, ..., \ell\})$.

For a solution $Q$ to $(G, S, T, \sigma, \tau)$ with an induced net mapping $\theta$, the numbers $|C_\ell^k(\theta)|$ measure the "similarity" between the achieved net mapping $\theta$ and the instance's target net mapping $\tau$. This will become clearer with Theorem 5.8.

A cut $(Z, V \setminus Z)$ with $\tilde{S} \cap Z = \{\tilde{s}_1, \tilde{s}_2, ..., \tilde{s}_\ell\}$ and $T \cap Z = \{t_1, t_2, ..., t_m\}$ will be called an *$\ell$–$m$–cut*. Imagine an instance with $\tilde{S}$ on the left and $T$ on the right. The set of (labels of) paths that start above $\tilde{s}_\ell$ and end below $t_{m+1}$ cross any $\ell$-$m$-cut downwards. The paths that cross these cuts upwards can be described analogously. This motivates the following definitions for $0 \le k \le |S_0|$ and $0 \le \ell, m \le |S|$:

$$U_{\ell,m}^k(\sigma,\theta) := \{j \in \mathrm{dom}(\sigma) : \ j \le k \text{ and } \sigma(j) \le \ell \text{ and } \theta(j) > m\} \quad \text{and}$$
$$D_{\ell,m}^k(\sigma,\theta) := \{j \in \mathrm{dom}(\sigma) : \ j \le k \text{ and } \sigma(j) > \ell \text{ and } \theta(j) \le m\}.$$

We observe that for all $k$, $\ell$, $m$, $\sigma$ and $\theta$ the equation

$$C_m^k(\theta) \,\dot\cup\, U_{\ell,m}^k(\sigma,\theta) = C_\ell^k(\sigma) \,\dot\cup\, D_{\ell,m}^k(\sigma,\theta) \tag{2}$$

holds and implies

$$|C_m^k(\theta)| + |U_{\ell,m}^k(\sigma,\theta)| = |C_\ell^k(\sigma)| + |D_{\ell,m}^k(\sigma,\theta)|. \tag{3}$$

An instance $(G, S, T)$ such that $S$ and $T$ are contained in disjoint intervals of the boundary of $G$ we call *$S$-$T$-disjoint*. We write $h := |S| = |T|$.

LEMMA 5.6. *If $(G'(X), \tilde{S}'(X), T'(X), \sigma, \tau)$ is a full $S$-$T$-disjoint $\mathcal{L}$–instance and $\theta$ is the induced net mapping of a solution $P$ that has been obtained by the algorithm from [Wagner and Weihe 1995], then* $\max\ D_{\ell,m}^h(\sigma,\theta) \le \min\ U_{\ell,m}^h(\sigma,\theta)$ *for all $l$, $m$.*

PROOF. Each path $P_i$ with $i \in D_{\ell,m}^h(\sigma,\theta)$ crosses each path $P_j$ with $j \in U_{\ell,m}^h(\sigma,\theta)$ from left to right. Thus, all path indices in $D_{\ell,m}^h(\sigma,\theta)$ are smaller than those in $U_{\ell,m}^h(\sigma,\theta)$, by Property 5.4(i). □

5.2.6 *A partial order on mappings.* When comparing two functions $\alpha, \beta : M \to N$ we call $\alpha$ *more orderly* than $\beta$, iff $\forall\ k, \ell :\ |C_\ell^k(\alpha)| \ge |C_\ell^k(\beta)|$; or $\alpha \succcurlyeq \beta$, for short. If $\alpha \succcurlyeq \beta$ and $|C_\ell^k(\alpha)| > |C_\ell^k(\beta)|$ for at least one pair $(k, \ell)$, then we write $\alpha \succ \beta$. This kind of comparison is useful for solutions as well: If $\theta$ and $\theta'$ are the induced net mappings of solutions $P$ and $P'$, respectively, then $P$ is said to be *more orderly* than $P'$, if and only if $\theta \succcurlyeq \theta'$. The following Lemma 5.7 shows that this partial order on mappings is not changed by translating, extending or restricting the mappings.

For integers $m \notin \mathrm{dom}(\alpha)$ and $n \notin \mathrm{im}(\alpha)$ we define $\alpha \cup (m, n)$ as the *augmentation* of $\alpha$ by $(m, n)$. In other words, we add to $\alpha$ the pair $\alpha(m) = n$. For integers $m$ and $n$ such that $\alpha(m) = n$, we can reduce $\alpha$ by $(m, n)$. We call $\alpha \setminus (m, n)$ the *reduction* of $\alpha$ by $(m, n)$. For an integer $c$ we define $a + c$ by $(\alpha + c)(i) := \alpha(i) + c$ for $i \in \mathrm{dom}(\alpha)$.

LEMMA 5.7. *Let $\alpha$ and $\beta$ be two bijective functions with* $\mathrm{dom}(\alpha) = \mathrm{dom}(\beta)$, $\mathrm{im}(\alpha) = \mathrm{im}(\beta)$ *and $\alpha \succcurlyeq \beta$. We then have*

*(a)* $\alpha + c \succcurlyeq \beta + c \qquad (\forall\ c \in \mathbf{Z})$,
*(b)* $\alpha \cup (m, n) \succcurlyeq \beta \cup (m, n)$ *and*

*(c)* $\alpha \setminus (m,n) \succcurlyeq \beta \setminus (m,n)$.

PROOF. (a): $|C^k_\ell(\alpha + c)| = |C^k_{\ell-c}(\alpha)| \geq |C^k_{\ell-c}(\beta)| = |C^k_\ell(\beta + c)|$.

(b): $|C^k_\ell(\alpha \cup (m,n))| = |C^k_\ell(\alpha)| + \left\{ \begin{array}{l} 1 \text{ if } m \leq k \text{ and } n \leq \ell, \\ 0 \text{ otherwise} \end{array} \right\} \geq |C^k_\ell(\beta \cup (m,n))|$.

(c): $|C^k_\ell(\alpha \setminus (m,n))| = |C^k_\ell(\alpha)| - \left\{ \begin{array}{l} 1 \text{ if } m \leq k \text{ and } n \leq \ell, \\ 0 \text{ otherwise} \end{array} \right\} \geq |C^k_\ell(\beta \setminus (m,n))|$. □

THEOREM 5.8. *Given a full S-T-disjoint $\mathcal{L}$–instance $(G'(X), \tilde{S}'(X), T'(X), \sigma_0, \tau)$, the algorithm from [Wagner and Weihe 1995] finds an induced net mapping $\theta_0$ such that for all solvable instances $(G'_u(X), \tilde{S}'(X), T'_u(X), \sigma, \theta)$ $\sigma \preccurlyeq \sigma_0$ implies $\theta \preccurlyeq \theta_0$. We say, $\theta_0$ is* most orderly *for $(G'(X), \tilde{S}'(X), T'(X), \sigma_0, \tau)$.*

We split the proof into several lemmata.

LEMMA 5.9. *Let $(G'(X), \tilde{S}'(X), T'(X), \sigma_0, \tau)$ be a full S-T-disjoint L–instance, $\theta_0$ an induced net mapping found by the algorithm from [Wagner and Weihe 1995], $\sigma \preccurlyeq \sigma_0$ and $\theta$ such that $(G'_u(X), \tilde{S}'(X), T'_u(X), \sigma, \theta)$ is solvable. If there exists a $\ell$-$m$-cut that is saturated in $(G'_u(X), \tilde{S}'(X), T'_u(X), \sigma_0, \theta_0)$, then for all $k$ with* $\max\ D^h_{\ell,m}(\sigma_0, \theta_0) \leq k \leq \min\ U^h_{\ell,m}(\sigma_0, \theta_0)$ *we have $|C^k_m(\theta)| \leq |C^k_m(\theta_0)|$.*

PROOF. Let $(Z, V \setminus Z)$ be a saturated $\ell$-$m$-cut. We denote by $\mathrm{dem}(Z)$ the demand of this cut w.r.t. $(G'_u(X), \tilde{S}'(X), T'_u(X), \sigma, \theta)$ and by $\mathrm{dem}_0(Z)$ the demand w.r.t. $(G'_u(X), \tilde{S}'(X), T'_u(X), \sigma_0, \theta_0)$. The paths with indices in $D^h_{\ell,m}(\sigma_0, \theta_0) \cup U^h_{\ell,m}(\sigma_0, \theta_0)$ cross any $\ell$-$m$-cut, in particular they cross $(Z, V \setminus Z)$. Since the latter is saturated and $(G'_u(X), \tilde{S}'(X), T'_u(X), \sigma, \theta)$ is solvable we have

$$|D^h_{\ell,m}(\sigma,\theta)| + |U^h_{\ell,m}(\sigma,\theta)| = \mathrm{dem}(Z) \leq \mathrm{dem}_0(Z) = |D^h_{\ell,m}(\sigma_0,\theta_0)| + |U^h_{\ell,m}(\sigma_0,\theta_0)|,$$

and

$$|D^h_{\ell,m}(\sigma,\theta)| - |U^h_{\ell,m}(\sigma,\theta)| = |Z \cap T| - |Z \cap S| = |D^h_{\ell,m}(\sigma_0,\theta_0)| - |U^h_{\ell,m}(\sigma_0,\theta_0)|.$$

Then subtraction shows $|U^h_{\ell,m}(\sigma,\theta)| \leq |U^h_{\ell,m}(\sigma_0,\theta_0)|$ and $|D^h_{\ell,m}(\sigma,\theta)| \leq |D^h_{\ell,m}(\sigma_0,\theta_0)|$. The latter inequality together with Lemma 5.6 implies

$$\begin{aligned} |D^k_{\ell,m}(\sigma,\theta)| &\leq |D^h_{\ell,m}(\sigma,\theta)| \leq |D^h_{\ell,m}(\sigma_0,\theta_0)| = |D^k_{\ell,m}(\sigma_0,\theta_0)|, \quad \text{and} \\ |U^k_{\ell,m}(\sigma,\theta)| &\geq 0 = |U^k_{\ell,m}(\sigma_0,\theta_0)| \end{aligned}$$

for all $\max\ D^h_{\ell,m}(\sigma_0,\theta_0) \leq k \leq \min\ U^h_{\ell,m}(\sigma_0,\theta_0)$. Equation (3) then yields the assertion. □

The following Procedure 3 then finds a cut $(Z, V \setminus Z)$ which will be seen to fulfill the prerequisites of Lemma 5.9. The procedure uses reverse left-first search on edges belonging to $P_1, P_2, ..., P_k$. It starts at $t_{m+1}$ and finishes at an $S$–terminal.

LEMMA 5.10. *Given integers $k$ and $m$, Procedure 3 finds an integer $l$ and a $l$-$m$-cut $(Z, V \setminus Z)$ such that*

*(i)* $\max\ \lambda(E^-(Z)) \leq k \leq \min\ \lambda(E^+(Z))$ *and*

*(ii) $(Z, V \setminus Z)$ is saturated and*

*(iii)* $\max\ D^h_{\ell,m}(\sigma_0,\theta_0) \leq k \leq \min\ U^h_{\ell,m}(\sigma_0,\theta_0)$.

**Procedure 3** Find a saturated $l$-$m$-cut

**Input:** A solvable instance $(G, S, T)$, a solution $P$ by the algorithm from [Wagner and Weihe 1995] and two integers $1 \leq m \leq |S|$ and $1 \leq k \leq |S_0|$.

**Output:** An integer $l$ and a set $Z \subset V(G)$.

1: Remove all edges from $E(G)$ that do not belong to any path $P_i$.
2: Transform each edge in $E(G)$ to an arc using the orientation induced by $P$.
3: Let $e = (v, t_{m+1})$ be the unique arc incident to $t_{m+1}$. Add $e$ to the empty path $Q$.
4: **while** $v \notin S$ **do**
5: Let $f = (w, v)$ be the clockwise next arc after $e$ which enters $v$ and belongs to a path $P_j$ with $j \leq k$.
6: Set $e := f$ and $v := w$.
7: Add $e$ to $Q$.
8: **end while**
9: Let $\ell$ be such that $v = \tilde{s}_{\ell+1}$.
10: **return** $l$ and all vertices on the right side of $Q$.

PROOF. Let $Q$ be the path constructed by Procedure 3. Consider an edge $g \in E(Z)$ and the vertex $v$ in $g \cap E(Q)$. Let $e = (u, v)$ and $f = (v, w)$ be the arcs entering resp. leaving $v$ on $Q$. Lemma 5.5(b) together with the procedure's reverse left-first search strategy implies $(e, f)_v \subset E_0$. Hence we have $E(Z) \subset E(A(G, S, T, x))$. The construction of $Q$ ensures $E(Q) \subset E(A(G, S, T, x))$. Therefore we can define $i := \lambda(g)$ and $j := \lambda(f)$.
If $g \in E^+(Z)$ then our reverse left-first search yields $i > k$. If, on the other hand, $g \in E^-(Z)$ then let $f'$ be the edge immediately preceding $f$ on $P_j$. Since the algorithm from [Wagner and Weihe 1995] uses right-first search to find the solution paths, we have $f' \notin [e, f]_v$. Hence, $g \in [e, f]_v \subset [f', f]_v$. Property 5.4(v) then shows $i \leq j$. And, by construction of $Q$, we have $j \leq k$. That completes the proof for assertion (i).
Assertions (ii) and (iii) are consequences of (i) and $E(Z) \subset E(A(G, S, T, x))$. □

PROOF OF THEOREM 5.8. We use Procedure 3 to find a saturated cut for each $k$ and $m$. Lemma 5.10 then yields the prerequisites of Lemma 5.9. Thus, we have $|C_m^k(\theta)| \leq |C_m^k(\theta_0)|$ for all $k$ and $m$. □

5.2.7 *Local nets.* In an $\mathcal{L}$–instance we define *local nets* to be those whose $T$–terminal is not adjacent to the bundle vertex $r$. If an $\mathcal{L}$–instance is solvable, then paths for all local nets can be found. Therefore, the instance is still solvable after removing all local nets and their corresponding paths. Moreover, this removal does not affect our partial order on the induced net mappings, because all solution paths for local nets are vertex disjoint to $r$. The nets constructed by the canonical extension are all local. Hence:

PROPOSITION 5.11. *A most orderly solution $\overline{P}$ to the canonical extension $\overline{(G, S, T)}$ of a possibly non-full $\mathcal{L}$–instance $(G, S, T)$ induces a solution $P$ to $(G, S, T)$ which is again most orderly.*

From now on we assume w.l.o.g. that no $\mathcal{L}$–instance has local nets. This assumption has useful consequences, because $\mathcal{L}$-instances without local nets are $S$-$T$-

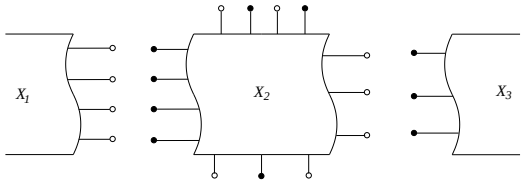


Fig. 19. A full partial instance $X_2$ and its neighbours $X_1$ and $X_3$. $S$-terminals are drawn as discs and $T$-terminals as circles.

disjoint.

### 5.3 Division and Recombination

At a non–leaf vertex of our decomposition tree we divide an instance $(G, S, T)$ along a $W$-cut found by the oracle. And obtain first a solution $P'$ to $(G'(X), S'(X), T'(X))$ and then a solution $P''$ to $(G''(X), S''(X, P'), T''(X, P'))$ as seen in the previous sections. We then combine the two partial solutions to form a solution $P$ to the instance $(G, S, T)$.

We say, a solution $P$ to an $\mathcal{L}$–instance $(G, S, T)$ is *most orderly* if in our extended notation $(G, \tilde{S}, T, \sigma_0, \tau) = (G, S, T)$ and the net mapping induced by $P$ is most orderly for $(G, \tilde{S}, T, \sigma_0, \tau)$.

LEMMA 5.12. *Consider an $\mathcal{L}$–instance $(G, S, T)$. Let $S^* \subset S$ and $T^* \subset T$ be intervals of terminals on the boundary of $G$. Then a solution $P^*$ to $(G, S^*, T^*)$ is most orderly if and only if a solution $P$ to $(G, S, T)$ that induces the same net mapping on $T^*$ is most orderly.*

PROOF. We write $(G, \tilde{S}, T, \sigma_0, \tau) = (G, S, T)$ and $(G, \tilde{S}^*, T^*, \sigma_0^*, \tau^*) = (G, S^*, T^*)$. Let $\theta_0$ and $\theta_0^*$ be the net mappings induced by $P$ and $P^*$, respectively.

We inspect how $\sigma_0^*$ arises from $\sigma_0$. Since $S^* \subset S$ is an interval of terminals, we can define $\tilde{S}^* = \{\tilde{s}_\ell, \tilde{s}_{\ell+1}, ..., \tilde{s}_k\}$. Now we remove from $\sigma_0$ all pairs $(m, n)$ such that $\sigma_0(m) = n$ and $m < \ell$ or $m > k$. We then arrive at $\sigma_0^*$ by subtracting $\ell$ from all remaining values of $\sigma_0$. By Lemma 5.7, these operations do not change the partial order on net mappings. For the transition from $\sigma_0^*$ to $\sigma_0$ we add $\ell$ to all values of $\sigma_0^*$ and then augment it. We proceed in the same way for the transitions between the starred and non–starred versions of $\sigma$, $\theta_0$ and $\theta$. In consequence we have $\sigma \preccurlyeq \sigma_0 \Longleftrightarrow \sigma^* \preccurlyeq \sigma_0^*$ and $\theta \preccurlyeq \theta_0 \Longleftrightarrow \theta^* \preccurlyeq \theta_0^*$.

If $P$ is most orderly then $\sigma \preccurlyeq \sigma_0$ implies $\theta \preccurlyeq \theta_0$ for all solvable unbundled $\mathcal{L}$–instances $(G_u, \tilde{S}, T_u, \sigma, \theta)$. And by the two previous equivalences this implicationis equivalent to $\sigma^* \preccurlyeq \sigma_0^* \Longrightarrow \theta^* \preccurlyeq \theta_0^*$ which means that $P^*$ is most orderly. □

LEMMA 5.13. *Let $(G, S, T)$ be solvable, $(X, V \setminus X)$ a saturated $W$-cut and $P'$ a most orderly solution to $(G'(X), S'(X), T'(X))$. Then $(G''(X), S''(X, P'), T''(X, P'))$ is solvable.*

PROOF. In order to define net mappings, we write $(G, S, T)$ as $(G, \tilde{S}, T, \sigma, \tau)$ and the partial instance $(G'(X), S'(X), T'(X))$ as $(G'(X), \tilde{S}'(X), T'(X), \sigma', \tau')$. Since $(G, \tilde{S}, T, \sigma, \tau)$ is solvable there exists a solution $Q'$ to $(G'(X), \tilde{S}'(X), T'(X), \sigma', \tau')$ with the induced net mapping $\theta'$ such that $(G''(X), \tilde{S}''(X, Q'), T''(X, Q'), \theta', \tau'')$ is

solvable. By the Theorem of Okamura and Seymour it suffices to show that no cut in the partial instance $(G''(X), \tilde{S}''(X,P'), T''(X,P'), \theta'_0, \tau'')$ has a demand higher than it has in $(G''(X), \tilde{S}''(X,Q'), T''(X,Q'), \theta', \tau'')$.

Now, w.l.o.g. consider cuts $(Y_1, Y_2)$ such that $Y_1$ and $Y_2 = V \setminus Y_1$ are connected and $\{\tilde{s}''_1, ..., \tilde{s}''_\ell, t''_1, ..., t''_m\} \subset Y_1$ and $\{\tilde{s}''_{\ell+1}, ..., \tilde{s}''_h, t''_{m+1}, ..., t''_h\} \subset Y_2$. Then

$$\begin{aligned} \mathrm{dem}_{P'}(Y_1) &= |U^h_{\ell,m}(\sigma,\theta_0)| + |D^h_{\ell,m}(\sigma,\theta_0)| \\ &= \ell + m - 2\ |C^m_\ell(\theta'_0)| \end{aligned}$$

which, by Theorem 5.8 is

$$\leq \ell + m - 2\ |C^m_\ell(\theta')| = \mathrm{dem}_{Q'}(Y_1).$$

□

Recursive application of the following result then shows the correctness of the Main Algorithm: If an $\mathcal{L}$–instance $(G,S,T)$ is solvable we find a most orderly solution to $(G'(X), S'(X), T'(X))$. Then $(G''(X), S''(X,P'), T''(X,P'))$ is solvable by Lemma 5.13. Once more we find a most orderly solution and the combination of both partial solutions is again most orderly. And so on, down to the root of the decomposition tree. That proves that the algorithm finds a solution if the instance is solvable.

LEMMA 5.14. *Let $(G,S,T)$ be an $S$-$T$-disjoint $\mathcal{L}$–instance, $P'$ a most orderly solution to the partial instance $(G'(X), S'(X), T'(X))$ and $P''$ a most orderly solution to $(G''(X), S''(X,P'), T''(X,P'))$. Then the combined solution $P$ to $(G,S,T)$ is most orderly.*

PROOF. We write $(G,S,T)$ as $(G,\tilde{S},T,\sigma_0,\tau)$. Let $\theta'_0$ be the net mapping induced by $P'$ and $\theta''_0$ the net mapping induced by $P''$. Since $P'$ is most orderly, $\sigma \preccurlyeq \sigma_0$ implies $\theta' \preccurlyeq \theta'_0$ for all solvable instances $(G'_u(X), \tilde{S}'(X), T'_u(X), \sigma, \theta')$. Therefore, the start net mapping for $(G''(X), S''(X,P'), T''(X,P'))$ cannot be more orderly than $\theta'_0$. As $P''$ is also most orderly, $\sigma'' \preccurlyeq \theta'_0$ implies $\theta'' \preccurlyeq \theta''_0$ for all solvable instances $(G''(X), \tilde{S}''(X,P'), T''(X,P'), \sigma'', \theta'')$.

Now we have shown the assertion under the assumption that all nets have their $S$–terminal in $G'(X)$ and their $T$–terminal in $G''(X)$. The general case ensues from Lemma 5.12. □

To complete the proof of Theorem 5.1, we remark that the linear complexity of the algorithm is obvious, given the linear complexity of the oracle (cf. Section 3) and the algorithm from [Wagner and Weihe 1995].

## REFERENCES


Becker, M. and Mehlhorn, K. 1986. Algorithms for routing in planar graphs. *Acta Informatica 23,* 2 (May), 163–176.

Frank, A. 1982. Disjoint paths in a rectilinear grid. *Combinatorica 2*, 361–371.

Frank, A. 1985. Edge-disjoint paths in planar graphs. *J. Comb. Theory B 39*, 164–178.

Kaufmann, M. 1990. A linear time algorithm for routing in a convex grid. *IEEE Trans. CAD-ICAS 9*, 180–184.

Kaufmann, M. and Klär, G. 1991. A faster algorithm for edge-disjoint paths in planar graphs. In *Proc. ISA*. Lecture Notes in Computer Science, vol. 557. Springer-Verlag Inc., New York, NY, USA, 336–348.

Kaufmann, M. and Mehlhorn, K. 1986. Generalized switchbox routing. *Journal of Algorithms 7*, 510–531.

Kaufmann, M. and Mehlhorn, K. 1990. *Routing problems in grid graphs*. Springer-Verlag Inc., New York, NY, USA, 165–184.

Knuth, D. 1975. *Art of Computer Programming, Vol. 3: Sorting and Searching, 2nd Print.* Addison-Wesley, Reading, MA, 168–177.

Lengauer, T. 1990. *Combinatorial Algorithms for Integrated Circuit Layout*. John Wiley & Sons Ltd., New York.

Mehlhorn, K. and Preparata, F. 1986. Routing through a rectangle. *J. ACM 33*, 60–85.

Middendorff, M. and Pfeiffer, F. 1993. On the complexity of the disjoint path problem. *Combinatorica 13*, 97–107.

Möhring, R. and Wagner, D. 1997. *Combinatorial topics in VLSI design (annotated bibliography)*. John Wiley & Sons Ltd., New York, 429–444.

Nishizeki, T. and Chiba, N. 1988. Planar Graphs: Theory and Algorithms. In *Annals of Discrete Mathematics*. Vol. 32. North-Holland, Amsterdam.

Okamura, H. and Seymour, P. 1981. Multicommodity flows in planar graphs. *J. Comb. Theory B 31*, 75–81.

Ripphausen-Lipa, H., Wagner, D., and Weihe, K. 1995. *Efficient algorithms for disjoint paths in planar graphs*. Vol. 20. American Mathematical Society, Providence, RI, 295–354.

Wagner, D. and Weihe, K. 1995. A linear-time algorithm for edge-disjoint paths in planar graphs. *Combinatorica 15*, 135–150.

Weihe, K. 1999. Edge-disjoint routing in plane switch graphs in linear time. In *40th Annual Symposium on Foundations of Computer Science: October 17–19, 1999, New York City, New York,*, IEEE, Ed. IEEE Computer Society Press, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 330–339. IEEE Catalog Number 99CB37039.