Report

An $O(n^4)$ time algorithm to compute the bisection width of solid grid graphs

Author(s):
Feldmann, Andreas Emil; Widmayer, Peter

Publication Date:
2011

Permanent Link:
https://doi.org/10.3929/ethz-a-006935587

Rights / License:
In Copyright - Non-Commercial Use Permitted
An $O(n^4)$ Time Algorithm to Compute the Bisection Width of Solid Grid Graphs

Andreas Emil Feldmann and Peter Widmayer

Institute of Theoretical Computer Science, ETH Zürich, Switzerland
{feldmann,widmayer}@inf.ethz.ch

Abstract. The bisection problem asks for a partition of the $n$ vertices of a graph into two sets of size at most $\lceil n/2 \rceil$, so that the number of edges connecting the two sets is minimised. A grid graph is a finite connected subgraph of the infinite two-dimensional grid. It is called solid if it has no holes. Papadimitriou and Sideri [7] gave an $O(n^5)$ time algorithm to solve the bisection problem on solid grid graphs. We propose a novel approach that exploits structural properties of optimal cuts within a dynamic program. We show that our new technique leads to an $O(n^4)$ time algorithm.

1 The Problem and Our New Approach

The problem of partitioning a graph into pieces of roughly equal sizes by cutting edges continues for decades to be of genuine theoretical interest and practical relevance, since it is usable in many divide-and-conquer algorithms. In our case the application stems from load distribution in parallel finite element simulations, where the input graph is a huge 3D-grid. The aim is to cut it into near equal sized pieces that each is scheduled onto a different machine. At the same time the interprocessor communication, modelled by the number of edges connecting the pieces, is to be minimised since this constitutes a bottleneck in parallel computations.

Of the above problem class we study a variant in which the graph is to be partitioned into two equal sized pieces. Our goal is to understand the nature of this problem from a theoretical viewpoint so as to lay the ground for further investigations of other problems of this type. A partition of the $n$ vertices of a graph into two sets of size at most $\lceil n/2 \rceil$ each is called a bisection, and the number of edges connecting the two sets is its cut-size. The bisection problem asks for a bisection of minimum cut-size, the bisection width. For general graphs, it is NP-hard [5], and it can be approximated with ratio $O(\log n)$ [9]. Even though the problem is weakly NP-hard [8] for planar graphs with vertex weights, the complexity is unknown for unweighted planar graphs. However a PTAS has been proposed [2]. For other graph classes such as trees and hypercubes the problem can be solved to optimality in polynomial time [1, 6]. We are interested in

* We gratefully acknowledge discussions with Peter Arbenz, and the support of this work through the Swiss National Science Foundation under Grant No. 200021_125201/1
grid graphs, defined as finite, connected subgraphs of the infinite two-dimensional grid. We assume that a grid graph is given together with its natural embedding in the plane, where each vertex is a coordinate in $\mathbb{N}^2$. We call a grid graph solid if it has no holes, i.e. it has no interior face surrounded by more than four edges (Figure 1). There is a polynomial time reduction [7] from planar graphs to grid graphs with holes, and to determine the complexity for planar graphs has been an open problem for many years. Hence restricting ourselves to solid grid graphs is a natural way to gain insights about the more general problem. Motivated by a VLSI layout problem, Papadimitriou and Sideri [7] presented an algorithm that computes the bisection width for a solid grid graph in time $O(n^5)$. More than 15 years ago, they asked whether this runtime can be improved. This problem appeared to be difficult ever since, because the approach by Papadimitriou and Sideri [7] was the only known one, and appeared not to be open to modifications that might lead to faster runtimes. In this paper we show that with a novel approach that makes use of some structural properties of a solution, a better runtime is indeed possible, and we present an $O(n^4)$ time algorithm.

A Bird’s Eye View of Our Technique. Our approach is based on two key findings. The first one limits the shapes of the pieces that are cut off from the solid grid, and it bounds the number of cut edges in an optimum cut. The second one uses these limits in a dynamic program.

The shape of a piece cut off from a solid grid is determined by the edges being cut, or alternatively by the sequence of faces (grid cells) leading from the exterior face through the grid and back to the exterior face (it is easy to see that the exterior face needs to be present for the cut to be optimum; see also [7]). It will be convenient to view this sequence of faces (including the exterior face) of the grid graph $G$ as a simple cycle in the dual graph of $G$. The dual graph is defined as the (multi-)graph whose vertices are faces of $G$ and whose edges are between vertices of adjacent faces. The edge set in $G$ corresponding to such a simple cycle is called a segment. The optimum cut will in general cut the graph into more than two pieces (more than two connected components). A cut in a grid graph therefore corresponds to a set of simple cycles in its dual graph. Hence the segments of a grid graph can be seen as building blocks of which a cut consists.

We recall [7] that it is enough for an optimum cut to limit the segments to the shape of a straight line, a corner, a stair, a clamp, or a square (Figure 2), with one small variation: for the sake of being able to cut off exactly the desired number of vertices, a side-step by one grid cell can be present that we call a break and define precisely later. Furthermore, we will show that a single stair, clamp, or square segment (with or without break) is enough in an optimum cut. That is, all others can be straight and corner segments. In addition, we can bound the
number of edges of any segment in an optimum cut to $O(\sqrt{n})$, by recalling [3]
that the bisection width of a solid grid on $n$ vertices is $O(\sqrt{n})$.

We make use of these limitations on segments by explicitly considering all
possible stairs, clamps, and squares, without and with a break. For each of both
parts into which such a segment cuts the grid, we only consider straight
and corner segments that cut away exactly the desired number of vertices. We are able
to compute the optimal way to cut out any desired number of vertices in each part
using only straight and corner segments inductively in a dynamic program. The
efficiency of our approach rests on the fact that there are only $O(n^2)$ segments
to be considered, since each segment is defined by three parameters: first, one of
the corners in its shape (at one of at most $n$ positions); second, the distance to
a (suitable) neighbouring corner of its shape (at most $O(\sqrt{n})$); and third, the
potential position of a break (at most $O(\sqrt{n})$ possible ones). We will show that
only an additional multiplicative term of $O(n^2)$ is needed to compute the optimal
bisection. This proves the claimed runtime of $O(n^4)$.

2 Properties of Optimal $k$-Cuts

For our dynamic program we need to generalise the bisection problem to consider-
ing $k$-cuts. These are sets of segments that, when removed from the graph, leave a
spanning subgraph that contains a set of connected components including exactly
$k$ vertices. We say that a $k$-cut $S$ cuts out the $k$ vertices of these connected
components, and we call the two sets of vertices of size $k$ and $n - k$ the parts
cut out by $S$. Given a $k$-cut $S$ we call the number of edges $\sum_{s \in S} |s|$ in $S$ its cut-size,
and we call a $k$-cut that minimises the cut-size over all $k$-cuts optimal. Notice
that some edges may be counted several times in the sum. However, edges that
appear more than once in different segments can be removed from a $k$-cut. This
is why this generalisation does not change the optimal solution.

In order to prove our claimed results we need to analyse the types of segments
that may occur in an optimal $k$-cut. We first recall [7] the result that these only
include so called straight, corner, stair, clamp, and square segments. Thereafter
we will prove that at most one of the segments in an optimal $k$-cut is not a
straight or corner segment. We begin by formally defining the above types of
segments (Figure 2) with the help of the dual (multi-)graph $D$ of a solid grid
graph $G$. In the following any face refers to a face of $G$, i.e. a vertex of $D$. Let $s$
be a segment in $G$ such that the simple cycle $p$ corresponding to $s$ in the dual $D$
includes the exterior face. An interior face \( f \) in \( D \) lying on \( p \) is called a bend of \( s \) if \( f \) touches two edges \( e_1, e_2 \in s \) such that \( e_1 \) and \( e_2 \) share a vertex. We say that the bend \( f \) points in two directions: the directions are up and right if \( e_1 \) and \( e_2 \) lie above and to the right of \( f \) respectively, and analogously they can be down or left if the edges lie appropriately. Two bends of \( s \) are said to point in opposing directions if they do not share any direction in which they point. If they share at least one direction they are said to point in a common direction. A break of \( s \) is an edge \( e \in s \) such that \( e \) touches two bends of \( s \) that point in opposing directions. Let \( q \) be a sub-path of \( p \) such that \( q \) starts and ends in two faces \( f_1, f_2 \) of \( D \) and \( f_1 \) and \( f_2 \) are bends of \( s \) or equal the exterior face \( f_\infty \). The subset \( b \) of \( s \) corresponding to \( q \) is called a bar of \( s \) if no face on \( q \) between \( f_1 \) and \( f_2 \) is a bend of \( s \) or equal to \( f_\infty \). The bar \( b \) is said to end at the two faces \( f_1 \) and \( f_2 \). The subset \( b \) is called a broken bar of \( s \) if \( b \) can be partitioned into three bars of \( s \) and the one ending neither at \( f_1 \) nor at \( f_2 \), i.e. the middle one, is a break of \( s \). Also the broken bar \( b \) is said to end at the two faces \( f_1 \) and \( f_2 \). Two bends of \( s \) are called consecutive if there is a bar of \( s \) ending at them.

**Definition 1** The segment \( s \) is called (Figure 2)

- a straight segment if it has no bend.
- a corner segment if it has exactly one bend.
- a stair segment if any consecutive bends of \( s \) point in opposing directions. Additionally, if \( s \) has no break then it has exactly two bends. Otherwise it contains a broken bar \( b \) such that \( s \setminus b \) constitutes at most two bars each of which ends at \( f_\infty \).
- a clamp segment if there are two bends \( f_1, f_2 \) of \( s \) pointing in a common direction. Additionally \( s \) can be partitioned into (1) a bar or broken bar \( b \) ending at \( f_1 \) and \( f_2 \), and (2) two bars that both end at \( f_\infty \).
- a square segment if there are three bends \( f_1, f_2, f_3 \) of \( s \) such that \( s \) can be partitioned into (1) a bar or broken bar \( b \) ending at \( f_1 \) and \( f_2 \), (2) a bar \( b' \) ending at \( f_2 \) and \( f_3 \), and (3) two bars ending at \( f_\infty \), and \( f_1 \) or \( f_3 \) respectively. Additionally each of the pairs \( f_1, f_2 \) and \( f_2, f_3 \) point in a common direction, and if \( b \) is a broken bar then \( f_1 \) and its consecutive bend of \( s \) touching the break, point in a common direction. Moreover \(|b'| \leq \beta' \leq |b'| + 1 \), where \( \beta \) is the length of \( b \) without counting its breaks. That is, \( \beta = \beta \) if \( b \) is a bar, and \( \beta = |b| - 1 \) if \( b \) is a broken bar.

If any of the above segments contains a broken bar we say that \( s \) has a break, and we refer to the corresponding break of the broken bar as the break of \( s \). For a clamp or square segment \( s \) let \( v \in V \) be the vertex of the grid graph that is shared by the edges touching the bend \( f_1 \) of \( s \). The part cut out by \( s \) including \( v \) is referred to as convex, and the other part as concave.

The next lemma, which states that there exists an optimal \( k \)-cut containing only the types of segments in the above definition, follows from Lemmas 3 and 4 in [7]: Lemma 3 therein can be used to convert the segments described by Lemma 4 in [7] in order to derive the particular shape required by the above definition.
Lemma 2 (follows from [7])  There is an optimal \(k\)-cut containing only straight, corner, stair, clamp, and square segments.

Furthermore at most one segment in an optimal \(k\)-cut is not a straight or corner segment. We prove this by shifting pieces of cut out areas from one segment to another. This is done so that the overall cut out area stays the same while the cut-size does not increase.

Theorem 3  There is an optimal \(k\)-cut that contains only straight and corner segments except at most one which is either a stair, clamp, or square segment.

Proof. According to Lemma 2 we can assume that an optimal \(k\)-cut \(S\) contains only straight, corner, stair, clamp, and square segments. Let \(A\) and \(B\) be the parts cut out by \(S\). It is easy to see that in an optimal \(k\)-cut there are no two segments sharing an edge, since such an edge could be removed yielding a \(k\)-cut of smaller cut-size. Hence all vertices incident to an edge in some segment \(s \in S\) and belonging to one of the cut out parts of \(s\), also belong to either one of \(A\) or \(B\). We assume that there are at least two segments in \(S\) that are not straight or corner segments and show how to convert \(S\) into another optimal \(k\)-cut containing at most one of these types of segments. For this we will identify two vertices \(v_A \in A\) and \(v_B \in B\) for any stair, clamp, and square segment \(s\). These are incident to some edge from \(s\) each and can be “exchanged” between \(A\) and \(B\) by only slightly changing the shape of \(s\). More concretely, of the incident edges to the vertex \(v_X\), where \(X \in \{A, B\}\), some belong to \(s\) and others do not. The vertex \(v_X\) will be chosen such that the segment \(s'\) including all edges from \(s\) except those incident to \(v_X\), but additionally including those incident to \(v_X\) that are not contained in \(s\), is a straight, corner, stair, clamp, or square segment. Hence if \(s\) is exchanged with \(s'\) the set \(X\) loses the vertex \(v_X\) in the cut. Notice that the segment \(s'\) may not exist since some of the edges incident to \(v_X\) may touch the exterior face of the grid graph. In this case the claimed segment \(s'\) is split into two segments. It is however possible to use Lemma 3 in [7] in order to convert the two resulting segments into straight, corner, stair, clamp, or square segments analogous to the way it is done for Lemma 2 above. These segments can then just as well be used in the following arguments and hence this special case will be ignored from now on.

Since we assume that \(S\) contains at least two segments that are not straight or corner segments, we can find another segment \(t \in S\) such that we can give back the lost vertex to \(X\). We do this by accordingly exchanging \(t\) with a segment \(t'\) such that a vertex \(v_Y\) from the other set \(Y \in \{A, B\} \setminus \{X\}\) is lost from \(Y\). Hence after exchanging both \(s\) and \(t\) with \(s'\) and \(t'\) respectively, the new set of segments is again a \(k\)-cut containing only straight, corner, stair, clamp, and square segments. We need however to make sure that the cut-size of the resulting \(k\)-cuts is non-increasing during these exchange steps, i.e. a step in which two segments are exchanged, in order to preserve optimality. This is not always possible for single exchange steps but we will show that there always exists a series of exchange steps for the considered segments \(s\) and \(t\) that lead to non-increasing cut-sizes.
We now list the vertices $v_A$ and $v_B$ for each stair, clamp, and square segment. For any segment $s$ with a break the two vertices are those incident to the break of $s$. If $s$ is a stair segment without break then let $f_1$ and $f_2$ be the two bends of $s$. In this case one of the two vertices $v_A$ and $v_B$ is the one shared by the two edges from $s$ touching the bend $f_1$, and the other vertex is the one shared by the two edges from $s$ touching $f_2$. If $s$ is a clamp or square segment without break then let $f_1$ and $f_2$ be the bends and $b$ the bar of $s$ as in Definition 1. In this case one of the two vertices $v_A$ and $v_B$ is the one incident to the edge from $b$ touching the bend $f_1$ and lying in the concave part cut out by $s$. The other vertex is the one incident to the edge from $b$ touching $f_2$ and lying in the convex part cut out by $s$. It is easy to check that in all of these cases each of the vertices $v_A$ and $v_B$ has the property that the corresponding segment $s'$, as described above, is again a straight, corner, stair, clamp, or square segment. In particular, also for square segments the properties on the lengths of the two (broken) bars $b$ and $b'$ are preserved.

If a vertex $v_X$, where $X \in \{A, B\}$, has more incident edges that are not contained in the respective segment $s$ than edges that are, then the corresponding segment $s'$ contains more edges than $s$ and thus the cut-size may increase if $s$ is exchanged with $s'$. Note however that this can only happen if $s$ is a clamp or corner segment without break and $v_X$ is contained in the concave part cut out by $s$. Therefore if $S$ contains two stair segments $s$ and $t$, we can use the vertex $v_A$ for $s$ and $v_B$ for $t$ to exchange $s$ and $t$ with $s'$ and $t'$ respectively without increasing the cut-size. It cannot be that $s'$ or $t'$ shares an edge with some other segment from the resulting $k$-cut, since such a shared edge could be removed which would result in a $k$-cut of smaller cut-size. Hence this exchange step can be repeated until one of the two segments is replaced with a straight or corner segment. Thus the resulting $k$-cut is optimal and has one stair segment less than $S$. Similarly if $s$ is a clamp or square segment and $t$ is a stair segment we fix $X \in \{A, B\}$ to be the set for which $v_X$ for $s$ lies in the convex part cut out by $s$. We can then exchange $s$ and $t$ with corresponding segments $s'$ and $t'$ repeatedly without increasing the cut-size. Clearly also in this case one of the segments will at some point be exchanged by a straight or corner segment.

If both $s$ and $t$ are clamp or corner segments let the respective (broken) bars referred to as $b$ in Definition 1 be $b_s$ and $b_t$. We assume w.l.o.g. that $|b_s| \leq |b_t|$. Fix $X \in \{A, B\}$ such that $v_X$ for $s$ lies in the convex part cut out by $s$. If $v_Y$ for $t$ lies in the convex part of $t$ then clearly we can find a sequence of exchange steps as above for which the cut-size does not increase. Also in the resulting $k$-cut one of the segments was subsequently replaced by a straight or corner segment. However if $v_Y$ lies in the concave part, as noted above the cut-size is increasing if $t$ has no break. Therefore in this case we need to consider a sequence of exchange steps until $s$ is subsequently replaced with a segment $s'$ that does not have a break. The number of these steps is at most $|b_s|$ and the segment $s'$ contains two edges less than $s$. Since $|b_s| \leq |b_t|$, there can be at most one segment subsequently replacing $t$ that does not have a break. Hence the corresponding resulting segment $t'$ can only have at most two edges more than $t$. All vertices
that are exchanged between $A$ and $B$ during this sequence of steps are incident to edges in $s$ and $t$ in the respective convex and concave cut out parts. Thus the subsequent vertices $v_A$ and $v_B$ always exist for the corresponding segments in the exchange sequence. As above it cannot happen that $s'$ and $t'$ share edges with other segments in the resulting $k$-cut since this would be a contradiction to optimality. Hence also for this case we can find a sequence of exchange steps for which the cut-size is non-increasing and the resulting $k$-cut contains one clamp or square segment less than $S$.

The proof is concluded by noticing that we can repeat this procedure for all pairs of segments that are neither straight nor corner segments until only one such segment remains. 

As a consequence of the above theorem the obvious way to proceed at this point would be to consider each stair, clamp, and square segment explicitly in the algorithm, as described in the introduction. However the runtime would in this case be larger than claimed since, for instance, there are $O(n^3)$ stair segments in the worst case. However not all of these will appear in an optimal $k$-cut since some of them are too large. This follows from the fact that the maximum degree of a grid graph is 4, and a result from [3] where it was shown that the bisection width of any planar graph of maximum degree $\Delta$ is $O(\sqrt{\Delta n})$. Hence by further restricting some of the segments to such ones that contain at most $O(\sqrt{n})$ edges we are able to reduce the runtime.

**Theorem 4 (follows from [3])** The bisection width of a grid graph is $O(\sqrt{n})$.

3 Computing Optimal $k$-Cuts

In this section we will present an algorithm to compute optimal $k$-cuts in solid grid graphs. We do this by assuming that we are given a solid grid graph $G$ and the set $\mathcal{S}$ of straight, corner, stair, clamp, and square segments in $G$. Some of the segments in $\mathcal{S}$ will have a length of at most $O(\sqrt{n})$. We will specify exactly which ones will be short in the next section where the runtime of our algorithm will be determined. According to Lemma 2 and Theorem 4 it suffices to compute an optimal $k$-cut that only uses segments from $\mathcal{S}$. More formally, for any set of segments $\mathcal{S}$ we say that any $k$-cut $S$ is $\mathcal{S}$-restricted if $S \subseteq \mathcal{S}$, and our goal is to compute an optimal non-crossing $\mathcal{S}$-restricted $k$-cut. Additionally we assume that we are given the set $\mathcal{K} \subseteq \mathcal{S}$ of straight and corner segments in $G$. According to Theorem 3 we know that any optimal $\mathcal{S}$-restricted $k$-cut contains at most one segment that is not from $\mathcal{K}$.

One crucial observation needed to construct the dynamic program is that we can assume that no segments cross in the optimal $k$-cut: observe that a simple cycle in the dual of a planar graph corresponds to a closed curve in the embedding of the dual graph in the plane. Hence the cycle divides the plane into an interior and an exterior area. We say that a pair of cycles cross if the corresponding closed curve of one of them both contains points belonging to
the interior and the exterior area into which the other cycle divides the plane. Note that any pair of simple cycles that cross can be seen as a (different) pair of simple cycles that do not cross (Figure 3). Hence we may limit ourselves to cuts in which no segments cross and we call these non-crossing.

The idea of the algorithm is to guess a stair, clamp, or square segment \( s \in S \setminus K \) from which we know that it is contained in the optimal solution and all other segments are straight and corner segments from \( K \). The case when the optimum only contains segments from \( K \) is dealt with separately. We split the graph into the two parts \( V_1^s \) and \( V_2^s \) cut out by \( s \). If the optimum \( K \)-restricted cuts in these two parts are known, then these can be used to compute the optimum containing \( s \). That is, we can compute the cut-size \( C_s(k) \) of an optimal \( k \)-cut that contains \( s \) and only segments from \( K \) that do not cross \( s \).

Let for \( i \in \{1, 2\} \) the set \( K_i^s \) include every segment \( t \in K \) that cuts out a part \( V_t \subseteq V_i^s \). Let also \( C_i^s(k) \) denote the cut-size of an optimal \( K_i^s \)-restricted \( k \)-cut. We define the value of \( C_i^s(k) \) to be infinite if no such cut exists. Using \( C_1^s(\cdot) \) and \( C_2^s(\cdot) \) we compute \( C_s(k) \) as follows. The corresponding \( k \)-cut cuts out some number \( k' \) of the vertices from \( V_1^s \). Since the computed solution should contain \( s \) as a segment the number of vertices cut out from \( V_2^s \) is \( |V_2^s| - (k - k') \). Thus the optimal cut-size is

\[
C_s(k) = \min \{ |s| + C_1^s(k') + C_2^s(|V_2^s| - (k - k')) \mid k' \in \{0, ..., k\} \}. \tag{1}
\]

Since the optimal solution contains at most one segment from \( S \setminus K \) and all others from \( K \), taking the minimum over all \( s \in S \setminus K \) of all computed values \( C_s(k) \) correctly computes the optimal \( k \)-cut if it contains a segment from \( S \setminus K \). To handle the case when the optimum only contains segments from \( K \), we define \( C_1^s(\cdot) \) and \( K_s^i \) accordingly for any \( s \in K \). Notice that then \( s \in K_s^i \). We treat this special case by also taking the cut-size \( C_s^i(\cdot) \) of a segment \( s \in K \) for which \( K_s^i = K \) into account in the final step. It is easy to see that such a segment always exists since \( G \) is a solid grid graph and \( K \) contains all straight and corner segments. Hence also taking the corresponding value \( C_s^i(\cdot) \) into account will correctly find the optimal solution for the given solid grid graph. Note that given the functions \( C_i^s(\cdot) \), for a fixed \( k \) the algorithm will take \( O(\sum_{s \in S \setminus K} n) \) time according to Equation (1) to compute the optimum \( k \)-cut. This is because for each segment \( s \in S \setminus K \) it needs to consider all possible values for \( k' \).

It remains to be shown how the \( K \)-restricted optima \( C_1^s(\cdot) \) and \( C_2^s(\cdot) \) are computed. The main inspiration for this part of our algorithm is taken from the corresponding algorithm for trees [6]. In a tree the segments correspond to single edges and a dynamic program is used to compute an optimal \( k \)-cut bottom-up from the leaves to the root. For each edge \( e \) of the tree the algorithm computes
the optimal solution for the subtree at $e$. It will decide whether to include $e$ in the solution by considering the optimal cuts in the subtrees immediately below $e$. Combining the cuts in the subtrees in order to compute the optimum up to $e$ is easy since they do not interfere with one another.

For our case we proceed in a similar way as for trees by inductively computing the cut-size $C_e^i(k)$ of an optimal $\mathcal{K}_e^i$-restricted $k$-cut for every $k \in \{0, \ldots, n\}$ in each part $V_e^i$, $i \in \{1, 2\}$, cut out by a segment $s \in S \setminus \mathcal{K}$. To compute the $\mathcal{K}_e^i$-restricted solutions in the two parts we will also need the solutions for any part cut out in $V_e^1$ and $V_e^2$ by a segment from $\mathcal{K}$. Hence we show how to compute the optima for the parts cut out by any segment from $S$, i.e. not excluding the straight and corner segments. Fix one of the parts cut out by $s \in S$ and call it $V_s$. We will decide whether to include $s$ into the solution for part $V_s$ by considering the cuts computed for segments cutting out parts from $V_s$. However these solutions do not interfere with one another since the parts can overlap. In order to circumvent this problem the idea is to guess where $V_s$ has to be split so that each segment of the non-crossing optimum is contained in one of the resulting pieces of $V_s$. We will use segments cutting out parts in $V_s$ for splitting. To find the correct way to split a part we give the following definition.

**Definition 5** Let $V_s$ be a part cut out by a segment $s$ and let $S$ denote a set of segments such that each $t \in S$ cuts out a part $V_t \subseteq V_s$. The set $S$ is called an interference-free set (IFS) in $V_s$ if $V_t \cap V_{t'} = \emptyset$ for each pair $t \neq t'$ from $S$. Let $\mathcal{K}_S$ contain all segments $u \in \mathcal{K}$ that cut out a part $V_u \subseteq V_t$ for some $t \in S$.

Note that $s$ itself cannot be contained in an IFS in $V_s$. In order to find the cut-size of the optimal $\mathcal{K}$-restricted $k$-cut in $V_s$ we will split $V_s$ according to each IFS from a small predefined set of IFSs in $V_s$. We will need one such predefined set for each part cut out by a segment $s \in S$ and hence call them $\mathcal{I}_s^1$ and $\mathcal{I}_s^2$. In the next section we will show that for each part we can find such a set that is small enough in order to guarantee the claimed runtime. The IFSs in the set include segments from $S$ and together have the property that they cover all IFSs including segments from $\mathcal{K}$ in $V_s^i$, in the following sense.

**Definition 6** Let $s \in S$ cut out a part $V_s$. An IFS covering set $\mathcal{I} \subseteq 2^S$ (w.r.t. $\mathcal{K}$) includes IFSs in $V_s$ such that for any IFS $S \subseteq \mathcal{K}$ also in $V_s$ there is a set $S^* \in \mathcal{I}$ for which $S \subseteq \mathcal{K}_S$.

Fix an IFS covering set $\mathcal{I}$ for the cut-out part $V_s$ we are considering. For any IFS $S$ in $\mathcal{I}$ let $C_S^i(k)$ denote the cut-size of the optimal $\mathcal{K}_S^i$-restricted $k$-cut. To compute the cut-size of the optimal $\mathcal{K}$-restricted $k$-cut in $V_s$ we can split $V_s$ according to each IFS $S^* \in \mathcal{I}$ and make use of the functions $C_{S^*}^i(\cdot)$. To see this we show that any non-crossing $\mathcal{K}$-restricted $k$-cut $S$ in $V_s$ is $\mathcal{K}_S$-restricted for some $S^* \in \mathcal{I}$. Consider the set $S'$ of segments containing any $t \in S \setminus \{s\}$ that cuts out a part $V_t \subseteq V_s$ such that there is no other segment $t' \in S \setminus \{s\}$ that cuts out a superset $V_{t'} \subseteq V_s$ of $V_t$. Since $S$ is non-crossing the set $S'$ is an IFS. Since also $S' \subseteq \mathcal{K}$, by the definition of an IFS covering set this means that $S \setminus \{s\}$ is $\mathcal{K}_S$-restricted for some $S^* \in \mathcal{I}$. Hence also the optimal non-crossing $\mathcal{K}$-restricted
Consider the case when \( s \) is included in the solution. In this case the number of cut-out vertices from \( V_s \) is \(|V_s| - k\).

Hence for any \( i \in \{1, 2\} \) and \( s \in S \), if the IFS covering set \( I^i_s \) for the cut out part \( V^i_s \) is non-empty then \( C^i_s(k) \) can be computed by

\[
C^i_s(k) = \begin{cases} 
\min\{C_S(k) \mid S^* \in I^i_s\} & \text{if } s \in S \setminus K, \\
\min\{C_S(k), |s| + C_S(|V^i_s| - k) \mid S^* \in I^i_s\} & \text{if } s \in K.
\end{cases}
\]

(2)

In case \( I^i_s \) is empty there are no segments that cut out a subset of \( V^i_s \). Hence then \( C^i_s(0) = 0 \) and if \( s \in K \) also \( C^i_s(|V^i_s|) = |s| \). All other values of \( C^i_s(\cdot) \) are infinite. Computing the table containing all values of the function \( C^i_s(\cdot) \) clearly takes \( \mathcal{O}(n \cdot \sum_{s \in S} |I^1_s \cup I^2_s|) \) steps if all values of the functions \( C_S(\cdot) \) of corresponding IFSs \( S^* \) are given.

The last missing part of this section is to show how a function \( C_S(\cdot) \) for a non-empty IFS \( S \) in a cut out part \( V_s \) of a segment \( s \in S \) can be computed. In order to find the cut size \( C_S(k) \) of an optimal \( K_S \)-restricted \( k \)-cut, the algorithm will combine the solutions computed for the segments in \( S \) in the same way the solutions for subtrees were combined in the algorithm for trees in [6]. If \( S \) contains only a single segment \( t \) then obviously \( C_S(k) = C^i_t(k) \), where \( i \in \{1, 2\} \) such that \( V^i_t \) is the part cut out by \( t \) from \( V_s \). In case \( S \) contains more than one segment, the value of \( C_S(k) \) can, for any fixed \( t \in S \), be recursively computed using the solutions to the IFS \( S \setminus \{t\} \) and the solution for the part \( V^i_t \subset V_s \). An optimal \( K_S \)-restricted \( k \)-cut must cut out some number \( k' \) of the \( k \) vertices from \( V^i_t \). The remaining \( k - k' \) vertices are taken from the parts cut out by the segments in \( S \setminus \{t\} \). Thus finding the minimum cut-size among all possible values of \( k' \) will find the optimal solution. Hence the following equation is correct:

\[
C_S(k) = \min\{C^i_t(k') + C_S(t)(k - k') \mid k' \in \{0, \ldots, k\}\}.
\]

(3)

To evaluate the right hand side of Equation (3) we need only consider values of \( k' \) that are at most the number \( n_t \) of vertices in \( V^i_t \) since all other values of \( C^i_t(\cdot) \) are infinite. This means that we can amortise the runtime needed to compute all values of \( C_S(\cdot) \) for a particular IFS \( S \) to \( \mathcal{O}(\sum_{t \in S} n \cdot n_t) = \mathcal{O}(n^2) \). This is true because we need to consider at most all the \( n + 1 \) possible values of \( k \) for each \( t \in S \) while the parts cut out by the segments in \( S \) are disjoint. To
compute all values of $C_s^i(\cdot)$ for all $s \in S$ we need to compute all values of $C_{S^*}(\cdot)$ for all $S^* \in \mathcal{I}_s$ and all $s \in S$. Hence computing the whole table for all values of $C_{S^*}(\cdot)$ takes $O(n^2 \cdot \sum_{s \in S} |\mathcal{I}_s^1 \cup \mathcal{I}_s^2|)$ time. Therefore the runtime of the algorithm (Figure 4) is dominated by the time needed to compute the table containing the values of the functions $C_{S^*}(\cdot)$.

4 Counting Segments and IFS Covering Sets

In Section 3 we have seen that we can efficiently compute optimal $k$-cuts for solid grids if the number of considered segments and IFS covering sets is small. Hence we need to identify a small IFS covering set $\mathcal{I}_s^1$ for each considered segment $s$ and each cut out part $V^i_s$, where $i \in \{1, 2\}$. In this section we will prove that the runtime of the given algorithm is $O(n^4)$ as claimed, by counting the number of segments and the sizes of the IFS covering sets. In order for the involved sets not to be too large, the set $S$ includes only straight, and corner segments, together with the stair, clamp, and square segments without breaks. It also contains all stair segments that consist of only a single broken bar. Additionally $S$ contains all stair, clamp, and square segments with breaks that have at most $c\sqrt{n}$ edges, for some constant $c$ according to Theorem 4. The latter theorem together with Lemma 2 guarantees that these sets $\mathcal{K}$ and $S$ suffice in order to compute an optimal $k$-cut using the algorithm in Section 3.

According to the results in Section 3 we need to show that $\sum_{s \in S} |\mathcal{I}_s^1 \cup \mathcal{I}_s^2| \in O(n^2)$ and that all required segments and IFS covering sets can be found efficiently, in order to guarantee a total runtime of $O(n^4)$. It was already shown in [4] that the number of straight and corner segments is $O(n)$, and the number of IFSs in each set $\mathcal{I}_s^1$ and $\mathcal{I}_s^2$ for any such segment $s$ also is $O(n)$. Additionally these segments and their IFS covering sets can also be listed [4] in time $O(n^2)$. Hence we only need to prove similar results for the stair, square, and clamp segments. We start by counting the number of segments for each such type.

Lemma 7 There are $O(n^2)$ stair, clamp, and square segments without breaks, and $O(n^2)$ of these segments with breaks having a length of at most $c\sqrt{n}$. Also there are $O(n)$ stair segments that consist of only a single broken bar. Furthermore all of these segments can be listed in time $O(n^2)$.

Proof. There are $O(n^2)$ stair, clamp, and square segments without breaks since according to Definition 1 each such segment can be identified with the respective bend $f_2$, the directions in which $f_2$ points (the directions in which the other bends point is determined by this), and a distance to the consecutive bends of $f_2$ (which in the case of a square segment amounts to choosing a length for the bar $b$). Since there are $O(n)$ faces in the grid graph that can be used for $f_2$, four directions in which to point, and the distance can be at most $n$, the result follows.

For segments with breaks each choice of the above three parameters also leaves the choice of a position of the break along the length of the respective broken bar. If the segment is a stair segment consisting of only a single broken bar then the only choice is its break and the direction in which the bends point. Thus
there are only $O(n)$ many such segments. For all other segments with breaks the length of the segment is assumed to be at most $c\sqrt{n}$. Therefore there are only $O(\sqrt{n})$ choices for the distance between the bends. This also means that there are $O(\sqrt{n})$ possible positions for the break, and hence the total number of segments with breaks is again $O(n^2)$.

The above counting arguments clearly give a straightforward way of listing all these segments too. Hence they can be found in time $O(n^2)$.

Finally we need to identify the IFS covering sets used for each of the segments $s \in S \setminus K$ and each $i \in \{1, 2\}$. We will prove that there is an IFS covering set of constant size for each such segment. The given construction of the sets can be used to compute them in a preprocessing step of the algorithm in $O(n^2)$ total time.

In all but the case of clamp or square segments with breaks we will ignore the fact that some of the segments in the claimed IFS covering set may partially lie outside of the grid graph. The reason why we can ignore these special cases is that if any segment, except for a clamp or square segment with a break, is split at any point (by the border of the grid graph) then the resulting segments that lie inside the grid graph are all segments of types again included in $S$. Hence these can be used for the IFS. On the other hand, splitting a clamp or square segment $s$ with a break may not result in a set of segments from $S$. We will therefore need to handle these two cases separately.

**Theorem 8** For any stair, clamp, or square segment $s \in S \setminus K$ in a solid grid graph $G$ and any $i \in \{1, 2\}$ there is an IFS covering set $I_i^s$ (w.r.t. $K$) containing at most three IFSs. Moreover all sets $I_i^s$ for segments in $s \in S \setminus K$ can be listed in time $O(n^2)$.

*Proof.* We will show the statement for every type of segment separately by giving a construction of an IFS covering set $I_i^s$. According to Definition 6, for each
The vertices in $V_s$ w.l.o.g. that any bend of $s$ we define horizontal we need to give the respective segments an orientation in the grid $G$. Therefore we define horizontal respectively vertical edges of $G$ to be those for which the incident vertices have the same $y$- respectively $x$-coordinates. A horizontal bar of a segment contains only vertical edges, and a vertical bar only horizontal edges. A horizontal respectively vertical broken bar of a segment contains two horizontal respectively vertical bars.

Consider the case when $s$ is a stair segment. If $s$ has no break let $b$ be the bar of $s$ ending at its two bends. Otherwise $b$ is the broken bar of $s$. We assume w.l.o.g. that any bend of $s$ points down and left, or up and right. Furthermore $b$ is assumed to be vertical, and the vertices that are the upper and right incident vertices of the vertical and horizontal edges of $s$, respectively, are those belonging to $V_s^i$.

If $s$ has no break (Figure 5(a)) it has two bends and hence any straight segment including edges from $b$ would in this case cut out vertices not contained in $V_s^i$. Thus any segment from $K_s^i$ that includes edges from $b$ must be a corner segment. Let $T \subseteq K_s^i$ contain all these corner segments. There is a corner or stair segment without break $t \in S$ that cuts out all vertices from $V_s^i$ except those that are incident to the edges in $b$. For the IFS $S_1^i = \{t\}$ in $V_s^i$ it holds that $K_{S_1^i} = K_s^i \setminus T$ since the only edges that can not be used to form a segment from $K_{S_1^i}$ are those in $b$. Hence for any IFS $S \subseteq K_s^i$ which does not contain a segment from $T$ it holds that $S \subseteq K_{S_1^i}$. Let $u \in S$ be the corner segment that cuts out a part $V_u$ from $V_s^i$ and has the same bend as $s$, say $f_1$, pointing up and right. The vertices in $V_s^i \setminus V_u$ are cut out by the corner segment $u' \in S$ having the other bend $f_2$ of $s$ as its bend which for $u'$ however points up and left. The only segments from $K_s^i$ that are not included in $K_{S_2^i}$, where $S_2^i = \{u, u'\}$, are those that have a horizontal bar crossing the vertical bars of $u$ and $u'$ above the bend $f_2$ of $s$. The vertical bar of any $r \in T$ is included in the vertical bar of $u$. Hence if an IFS $S \subseteq K_s^i$ contains a corner segment $r \in T$ then $S \subseteq K_{S_2^i}$ since $S$ is non-crossing. In conclusion the set $I_{S_1^i}$ for any $i \in \{1, 2\}$, is an IFS covering set if it contains the IFSs $S_1^i$ and $S_2^i$ in case $s$ is a stair segment without break.

If $s$ is a stair segment with break (Figure 5(b)) let $v$ be the vertex in the cut out part $V_s^i$ incident to the break of $s$. Two of the four incident edges to $v$, say $e_1$ and $e_2$, belong to $s$, while the other two, say $e'_1$ and $e'_2$, are not included in $s$. Let $t$ be the straight, corner, or stair segment that contains the edges of $s$ but where $e_1$ and $e_2$ are exchanged with $e'_1$ and $e'_2$. Note that $t$ may or may not have a break if it is a stair segment, but $t$ is always included in $S$ since it contains the same number of edges as $s$. The only segment in $K_s^i$ that cuts out the vertex $v$ from $V_s^i$ is the corner segment $r$ having the face touching $e_1$ and $e_2$ as its bend and pointing up and right. Hence if an IFS $S \subseteq K_s^i$ does not contain $r$ then $S \subseteq K_{S_1^i}$, where $S_1^i = \{t\}$. The vertices from $V_s^i$ that are not contained in the cut out part $V_r \subset V_s^i$ by $r$ are cut out by the following segments. Those belonging to the connected component induced by $V_s^i \setminus V_r$ and containing vertices
incident to the edges of the horizontal bar of r, are cut out by a corner or clamp segment u ∈ S without break. All remaining vertices in V_s^i, if any, are cut out by a corner segment u^i ∈ S. Let the IFS S_1^i contain r and u, and also u^i if it exists. Obviously any IFS S containing r must be contained in the set K_{S_2^i}. Therefore in case s is a stair segment with break the set I_s^i, for any i ∈ {1, 2}, containing the IFSs S_1^i and S_2^i is an IFS covering set.

Consider the case when s is a clamp or square segment. We use the same names for the bends (f_1, f_2, f_3) and (broken) bars (b, b') of s as in Definition 1. Assume w.l.o.g. that the bend f_1 points down and right.

We first regard the case when V_s^i is the convex part cut out by s (Figure 5(c)). Let b'' ⊆ b be the horizontal bar of s ending at f_1. Any straight or corner segment including edges from b'' cuts out vertices that are not included in V_s^i since such a segment has at most one bend. Hence for the segment t cutting out all vertices from V_s^i except those that are incident to the edges in b'' it holds that K_{S''} = K_{S_1^i}, where S'' = {\{t\}}. Note that t is either a straight, corner, or clamp segment if s is a clamp segment, and that t is a clamp or square segment if s is a square segment. Furthermore if t is a clamp or square segment it does not have a break, regardless of whether s has one. If t is a square segment it is also important to check that the horizontal and vertical bars of t not ending at the exterior face of G have the correct lengths. This is true due to the assumptions on the sizes of s’s bars b and b’ according to Definition 1, except in the case when s is a square segment without break and b’ is smaller than b. However in the latter case we may consider the analogous case where the identities of b and b’ are swapped. Hence t ∈ S which means that in this case I_s^i = \{S''\} is an IFS covering set.

Finally we turn to the case when V_s^i is the concave part cut out by a clamp or square segment s. If s has no break (Figure 5(d)) let T ⊆ K_s^i contain all straight and corner segments that include some edge from the bar b. There is a clamp or square segment t (t is of the same type as s) that cuts out all vertices in V_s^i except those that are incident to the edges in b. For the IFS S_1^i = \{t\} in V_s^i it holds that K_{S_1^i} = K_{S_1^i} \ T since the only edges that can not be used to form a segment from K_{S_1^i} are those in b. Hence for any IFS S ⊆ K_s^i which does not contain a segment from T it holds that S ⊆ K_{S_1^i}. Let t_1 be the corner segment that has the bend f_1 of s as its bend which for t_1 points left and down. Also let also t_1', the corner segment with a bend at f_1 that for t_1' however points right and down. In case s is a clamp segment let t_2 be the corner segment that has the bend f_2 of s as its bend which for t_2 points left and down. Also let t_2', be the corner segment with a bend at f_2 which for t_2' points right and down. If s is a square segment we define t_2 to be the clamp segment without break that has the bends f_2 and f_3 of s as bends such that f_2 points left and down for t_2. In addition we define t_2'' to be a stair segment that has f_2 and f_3 as its only bends such that f_2 points right and down for t_2'. Depending on the length of b' the stair segment t_2'' either has no break or consists of only a single broken bar. Thus all of the defined segments above are contained in S. For any of the cases the only segments from K_{S_1^i}, that are not included in K_{S_2^i}, where S_2^i = \{t_1, t_2\}, are those that have a vertical bar crossing the horizontal bars of t_1 and t_2 to the
left of the bend $f_1$ of $s$. Analogously the only segments from $K^i_s$ that are not included in $K_{S_2}$, where $S_2^i = \{t'_1, t'_2\}$, are those that have a vertical bar crossing the horizontal bars of $t_1$ and $t_2$ to the right of the bend $f_2$ of $s$. Any straight or corner segment $r \in T$ that contains an edge from $b$ has a horizontal bar that extends to the left from $f_1$ or to the right from $f_2$, since $r$ has at most one bend. Hence if an IFS $S \subseteq K^i_s$ contains a segment $r \in T$ for which the horizontal bar is included in the horizontal bar of $t_2$, since $S$ is non-crossing it must be that $S \subseteq K_{S_2}$. Analogously, if the horizontal bar if $r$ is included in the horizontal bar of $t'_1$ then $S \subseteq K_{S_2}$. Therefore in this case the set $I'_s = \{S_1^i, S_2^i, S_3^i\}$ is an IFS covering set.

If $s$ is a clamp or square segment with break (Figure 5(e)) let $v$ be the vertex in the concave cut out part $V^i_s$ incident to the break of $s$. Two of the four incident edges to $v$, say $e_1$ and $e_2$, belong to $s$, while the other two, say $e'_1$ and $e'_2$, are not included in $s$. Let $t$ be the clamp or square segment ($t$ is of the same type as $s$) that contains the edges of $s$ but where $e_1$ and $e_2$ are exchanged with $e'_1$ and $e'_2$. Note that $t$ may or may not have a break, but $t$ is always included in $S$ since it contains the same number of edges as $s$. The only segment in $K^i_s$ that cuts out the vertex $v$ from $V^i_s$ is the corner segment $r$ having the face touching $e_1$ and $e_2$ as its bend which points up and right. Hence if an IFS $S \subseteq K^i_s$ does not contain $r$ then $S \subseteq K_{S_1}$, where $S_1^i = \{t\}$. The vertices from $V^i_s$ that are not contained in the cut out part $V_r \subset V^i_s$ by $r$ are cut out by the following segments. Those belonging to the connected component induced by $V^i_s \setminus V_r$ and containing vertices incident to the edges of the horizontal bar of $r$ are cut out by a corner or stair segment $u \in S$, if $s$ is a clamp or square segment respectively. If $u$ is a stair segment it either has no break or it consists of only a single broken bar, depending on the size of $b'$. All other vertices in $V^i_s$ are cut out by a stair segment $u' \in S$. Also $u'$ either has no break or consists of only a single broken bar, depending on the size of $b$. Obviously any IFS $S$ containing $r$ must be contained in the set $K_{S_2}$, where $S_2^i = \{r, u, u'\}$. Therefore in this case the set $I'_s = \{S_1^i, S_2^i\}$ is an IFS covering set.

Notice that in the last case, if $s$ is a square segment with break the edges $e'_1$ and $e'_2$ may touch the exterior face. In this case the segment $t$ is split at the corresponding face (Figure 6). If $t$ is a square segment with break, one of the resulting segments however is of none of the types that are contained in $S$. This means that for this special case we need to find a different IFS covering set than proposed above. The set $S_3^i$ can still be used and thus for any IFS $S \subseteq K^i_s$ containing $r$ it holds that $S \subseteq K_{S_2}$. However for the case when $r \notin S$, instead of
$S_i^*$ as above we construct the following two IFSs for $I_i$. Let $e'_1$ be horizontal and $e'_2$ vertical. Let $v' \in V_i^*$ be the vertex not equal to $v$ incident to $e'_1$. Apart from $r$ there is only one other segment $r' \in K_i^*$ that cuts out $v'$ from $V_i^*$, which as $r$ is also a corner segment. Except for $v$, the vertices from $V_i^*$ that are not contained in the cut out part $V_{r'} \subset V_i^*$ by $r'$ are cut out by the following segments. Those belonging to the connected component induced by $V_i^* \setminus (V_{r'} \cup \{v\})$ and containing vertices incident to the edges of the horizontal bar of $r'$ are cut out by a stair segment $u \in S$. This stair segment either has no break or consists of only a single broken bar, depending on the size of $b'$. All remaining vertices in $V_i^* \setminus \{v\}$ are cut out by a corner segment $u'$. Obviously any IFS $S \subset K_i^*$ not containing $r$ but $r'$ must be contained in the set $K_{S_i^*}$, where $S_i^* = \{r', u, u'\}$. All vertices in $V_i^*$ except $v$ and $v'$ that are not contained in the cut out part $V_{u'} \subset V_i^*$ by $u'$ are cut out by a clamp segment $u''$ without break. Hence for an IFS $S \subset K_i^*$ neither including $r$ nor $r'$ it must be that $S \subset K_{S_i^*}$, where $S_i^* = \{u', u''\}$. Therefore in this special case the set $I_i = \{S_i^*, S_i^*, S_i^*\}$ is an IFS covering set.

All the above constructions give a straightforward way to list the IFS covering sets. Since by Lemma 7 there are $O(n^2)$ many segments in $S \setminus K$ and each such segment needs at most three IFS covering sets, all sets can be found in time $O(n^2)$. □

5 Discussion

We presented a novel approach for solid grid bisection that runs in time $O(n^4)$, in contrast to the best earlier algorithm whose runtime was $O(n^5)$. The new algorithm is based on structural properties of the cut, and on the use of these properties within a dynamic program. Just as in the original algorithm by Papadimitriou and Sideri [7] we not only compute the optimal bisection but the optimum $k$-cut for every $k \in \{0, \ldots, n\}$. This means that also objective functions depending on the cut size and the balance of the cut can be optimised in time $O(n^4)$. For instance in load distribution applications, such as ours, the cut size may be more important than the balance.

One might ask whether the structural properties could also have been used with the original algorithm [7]. The answer is negative. The reason is that the original algorithm recursively computes an optimal $k$-cut for each pair of boundary edges of the grid by splitting the boundary in clockwise direction from one edge of the pair to the other. A limitation like ours on the useful segments does not positively affect the runtime of this approach. It remains to be seen whether these structural properties of the cut, cast into an algorithm, have the potential to lead to improved solutions for more general classes of graphs (such as planar graphs) and for more general cuts (such as partitions into any given number of parts). We remark that for some immediate generalisations from solid grid graphs the presented observations on the structure of the segments do not hold. For instance giving weights to the vertices or edges makes the shapes of the segments in a bisection much more complex. Also it follows from [7] that grid graphs with holes are essentially as hard to bisect as planar graphs, and determining the
complexity of the latter is a long standing open problem. However we believe that
genralisations to graphs with a very regular structure are possible. For instance
if the interior faces of the considered (unweighted) graph constitute a tessellation
of the plane, then similar observations on the structure of segments and their
IFS covering sets as the ones given for solid grid graphs should be achievable.
One motivation for such instances is that many 2D finite element models use
triangles as tessellations.

References

2. J. Díaz, M. J. Serna, and J. Torán. Parallel approximation schemes for problems on
3. K. Diks, H. N. Djidjev, O. Sykora, and I. Vrto. Edge separators of planar and
    right-angled cuts in solid grids. In Proceedings of the 4th Annual International
7. C. Papadimitriou and M. Sideri. The bisection width of grid graphs. Theory of
    In Proceedings of the twenty-fifth annual ACM symposium on Theory of computing,
    STOC ’93, pages 766–775, New York, NY, USA, 1993. ACM.
9. H. Räcke. Optimal hierarchical decompositions for congestion minimization in net-
    works. In Proceedings of the 40th Annual ACM Symposium on Theory of Computing,
    2008.