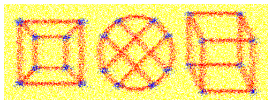

Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 13, no. 2, pp. 119–131 (2009)

# Vertex Bisection is Hard, too

*Ulrik Brandes  Daniel Fleischer*

Department of Computer & Information Science
University of Konstanz

**Abstract**

We settle an open problem mentioned in Diaz, Petit, and Serna: A survey of graph layout problems (*ACM Computing Surveys* 34:313–356, 2002). Of eight objectives considered in that survey, only the complexity status of minimum vertex bisection is listed as unknown. We show that both minimum and maximum vertex bisection are $\mathcal{NP}$-hard, but polynomially solvable on special graph classes such as hypercubes and trees.

| Submitted: | Reviewed: | Revised: | Accepted: |
|---|---|---|---|
| December 2005 | April 2006 | April 2007 | February 2009 |
| | Final: | Published: | |
| | February 2009 | April 2009 | |
| Article type: | | Communicated by: | |
| Concise paper | | S. Albers | |

*E-mail addresses:* Ulrik.Brandes@uni-konstanz.de (Ulrik Brandes) Daniel.Fleischer@uni-konstanz.de (Daniel Fleischer)

# 1 Introduction

We consider simple undirected graphs $G = (V, E)$. A partition $(S, V \setminus S)$ induced by a proper subset $\emptyset \neq S \subsetneq V$ of the vertices is called a *cut*. A cut $(B, V \setminus B)$ that divides the vertex set evenly, i.e. $|B| \in \{\lfloor |V|/2 \rfloor, \lceil |V|/2 \rceil\}$, is called a *bisection*.

The *width* of a bisection is measured either in terms of the number of edges or the number of vertices connecting the two subsets. A bisection $(B, V \setminus B)$ (or simply called $B$) has *edge width* $\kappa(B) = |\{\{v, w\} \in E : v \in B, w \in V \setminus B\}|$ and *vertex width* $\lambda(B) = |\{v \in B : \exists w \in V \setminus B, \{v, w\} \in E\}|$. Thus, $\lambda(B)$ counts the number of vertices in $B$ that have neighbors outside $B$.

Minimum edge bisections are of interest, e.g., for divide and conquer approaches in VLSI design [3, 1]. Minimum vertex bisection is relevant, e.g., for certain forms of gossiping in communication networks [11].

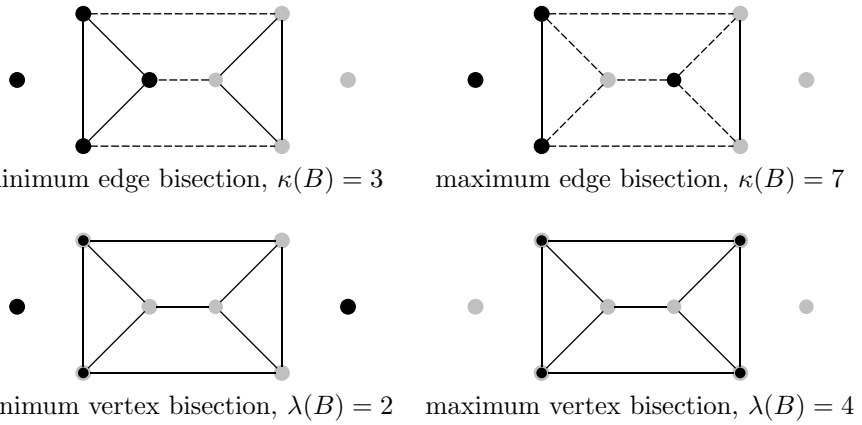


Figure 1: Optimum bisections of an example graph. Vertices in $B$ are filled black (and surrounded grey if and only if there is a neighbor in $V \setminus B$). Cut edges are dashed.

Figure 1 shows bisections of optimum edge and vertex width of an example graph. While cuts of minimum edge width (and arbitrary size) can be determined in polynomial time using network-flow techniques [18], the problem of finding a bisection of minimum edge width is known to be $\mathcal{NP}$-complete [6] and hence (by using the complementary graph $\overline{G}$) the problem of finding a bisection of maximum edge width is also $\mathcal{NP}$-complete. Algorithms with good average case performance for bisections with small edge width can be found in e.g. [3]. A polylogarithmic approximation is given in [5]. Determining cuts of maximum edge width and arbitrary size (i.e. MAXCUT) is also $\mathcal{NP}$-complete [9], but there exists a 0.878 approximation [7].

The complexity status of the remaining variant of determining a bisection of minimum vertex width is stated as open in [4]. We show that both the minimum and maximum versions are $\mathcal{NP}$-complete as well, but polynomially solvable for some graph classes.

## 2 Hardness Results

We show that the decision problems associated with minimum and maximum vertex bisection are $\mathcal{NP}$-complete.

**Definition 1 (MIN VERTEX BISECTION)** *Given an undirected graph $G = (V, E)$ and $k \in \mathbb{N}$, is there a subset $B \subseteq V$ with $|B| \in \{\lfloor |V|/2 \rfloor, \lceil |V|/2 \rceil\}$ and $\lambda(B) \leq k$?*

**Theorem 1** MIN VERTEX BISECTION *is $\mathcal{NP}$-complete.*

**Proof:** Clearly, MIN VERTEX BISECTION is in $\mathcal{NP}$, since we can *guess* a subset $B$ with $|B| \in \{\lfloor |V|/2 \rfloor, \lceil |V|/2 \rceil\}$ and verify whether $\lambda(B) \leq k$.

We show that 3SAT can be polynomially transformed into MIN VERTEX BISECTION. Let $I = (X, C)$ be an instance of 3SAT with variables $X = \{x_1, \ldots, x_\ell\}$ and clauses $C = \{c_0, \ldots, c_{m-1}\}$. W.l.o.g. each clause contains only pairwise different variables. We construct the following graph $G = (V, E)$ with $256m^2+14m$ vertices that are partitioned into 3 sets $S, T, U$, where $S$ is a $128m^2$-clique, $T$ is a $(128m^2 + 6m)$-clique and $U$ consists of $8m$ vertices. One half of the vertices of $S$ are called $s_{i,j}$ and the other half $\tilde{s}_{i,j}$, where $0 \leq i, j < 8m$. The vertices of $U = \{u_i : 0 \leq i < 8m\}$ correspond to a truth assignment of clause $c_j$, where $j = \lfloor i/8 \rfloor$ and the last three binary digits of $i$ define the truth assignment (e.g. $(0, 1, 1)$ for $i = 19$ means the second and third variable of $c_2$ are set true, etc., see Figure 2). Furthermore, we define the following adjacencies.

$$\{u, t\} \in E, \forall u \in U, t \in T, \tag{1}$$

$$\{u_i, s\} \in E, \forall s \in S \text{ and } u_i \textit{ yields a false clause} \tag{2}$$

$$\{u_i, s_{i,j}\} \in E, \forall i, j \tag{3}$$

$$\{u_i, s_{j,i}\} \in E, \forall i, j \text{ and the assignments of the clauses corresponding to } u_i, u_j \text{ do not } \textit{contradict}, \tag{4}$$

$$\{u_i, \tilde{s}_{i,j}\} \in E, \forall i \text{ and } 0 \leq j < \eta_i \text{ and } \eta_i \text{ is the number of clause assignments in } \textit{contradiction} \text{ to } u_i. \tag{5}$$

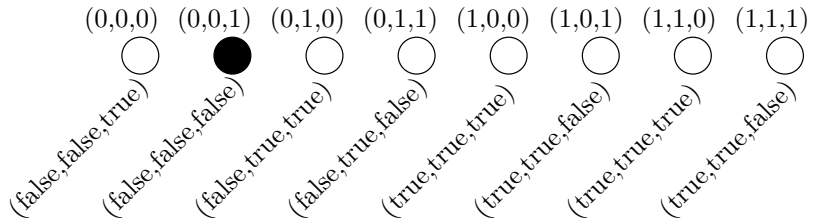


Figure 2: All truth assignments corresponding to $2^3$ vertices for a clause $c_2 = (x_1, x_2, \overline{x}_3)$, i.e. vertices $u_8$ to $u_{15}$. The black vertex corresponds to a false clause.

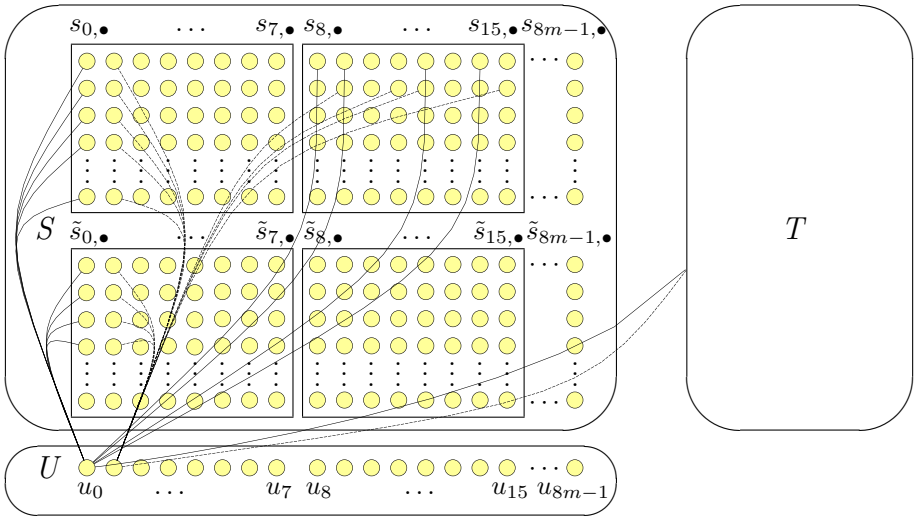


Figure 3: Example for the adjacencies of type (1)–(5) of vertices $u_0$ and $u_1$ that do not yield a false clause.

The basic idea is that there will be $m$ vertices $u \in U$ that are not in the optimum $B$ and they correspond to an assignment of the clauses of $I$. The adjacencies in (1) assure that $T \subseteq V \setminus B$. In (2) we assure that assignments that *yield a false clause* (e.g. $i = 17$ for clause $c_2$, see also Figure 2) are in $B$. We say that two clause assignments *contradict* if they assign different values, i.e. true and false, to a variable (e.g. clause assignments $i = 19$, i.e. $(0, 1, 1)$ for $c_2 = (x_1, x_2, \overline{x}_3)$ and $i = 39$, i.e. $(1, 1, 1)$ for $c_4 = (x_1, x_4, x_5)$ contradict). Note that the eight clause assignments of a clause $c_j$ pairwise contradict. As a consequence only one of the eight vertices corresponding to these assignments will not be in the optimum $B$. The adjacencies in (4) and (5) assure that every vertex $u \in U$ that does not yield a false clause is connected to exactly $16m$ vertices of $S$. See Figure 3.

Assume that $I$ is satisfiable and let $Y$ be a solution consisting of literals $y_1, \ldots, y_\ell$ with each $y_i$ of the form $x_i$ or $\overline{x}_i$ and consider the following partition $B_Y \cup (V \setminus B_Y)$, where

$$B_Y = S \cup \{u_i \in U \,:\, \text{the truth assignment corresponding to } u_i \text{ differs from } Y\}\,.$$

The $7m$ vertices of $U \cap B_Y$ contribute $7m$ to $\lambda(B_Y)$. Every vertex of the $m$ vertices in $U \setminus B_Y$ has exactly $16m$ neighbors in $B$, but since the clause assignments corresponding to these $m$ vertices pairwise do not contradict, they pairwise share exactly one of the $16m$ neighbors. Hence, this partition induces

$$\lambda(B_Y) = 7m + \sum_{j=0}^{m-1} 16m - j = 15.5m^2 + 7.5m\,.$$

We show that this is the minimum that can only be achieved if $I$ is satisfiable. Thus, 3SAT can be solved by the decision problem of finding a bisection $B$ with $\lambda(B) \leq 15.5m^2 + 7.5m$. Clearly, the transformation can be done in polynomial time and hence MIN VERTEX BISECTION is $\mathcal{NP}$-complete.

It can easily be verified that for the minimum $S \subseteq B$ and $T \subseteq V \setminus B$ is required. Assume now that $B = S \cup U$. This partition induces $\lambda(B) = 8m$. Then we have to remove $m$ vertices of $U$ from $B$. Every vertex of $U$ corresponding to a false clause cannot be removed since this partition (after $m$ vertices have been removed from $B$) yields a value of greater than $128m^2 > \lambda(B_Y)$. The removal of every other vertex $u \in U$ contributes (at every step of removing) $16m - 1$ minus the number of vertices already removed and not in contradiction to $u$. Only if every removed vertex does not contradict to any already removed vertex the value of $15.5m^2 + 7.5m$ will be achieved and otherwise exceeded. □

**Definition 2 (MAX VERTEX BISECTION)** *Given an undirected graph $G = (V, E)$ and $k \in \mathbb{N}$, is there a subset $B \subseteq V$ with $|B| \in \{\lfloor |V|/2 \rfloor, \lceil |V|/2 \rceil\}$ and $\lambda(B) \geq k$?*

**Theorem 2** MAX VERTEX BISECTION *is $\mathcal{NP}$-complete.*

**Proof:** Clearly, MAX VERTEX BISECTION is in $\mathcal{NP}$. Since we have no simple reduction from the minimization problem, we show that 3SAT can be polynomially transformed into MAX VERTEX BISECTION. Our proof shows that even the decision problem for $k = \lfloor |V|/2 \rfloor$ is $\mathcal{NP}$-complete. Let $I = (X, C)$ be an instance of 3SAT with variables $X = \{x_1, \ldots, x_\ell\}$ and clauses $C = \{c_0, \ldots, c_{m-1}\}$. W.l.o.g. each clause contains only pairwise different variables. We construct the following graph $G = (V, E)$ with $16m^2 + 18m$ vertices, that are partitioned into 3 sets $S, T, U$, where $S$ and $T$ are isolated vertices of size $8m^2 + 2m$ and $8m^2 + 8m$, respectively, and $U$ is an $8m$-clique. The vertices of $S$ are called $s_{i,j}$, where $0 \leq i < 8m$ again corresponds to a truth assignment of clause $c_{\lfloor i/8 \rfloor}$ and $0 \leq j < m$. The remaining $2m$ vertices of $S$ are called $s'_1, \ldots, s'_m$ and $s''_1, \ldots, s''_m$. The vertices of $U = \{u_i : 0 \leq i < 8m\}$ again correspond to a truth assignment of clause $c_{\lfloor i/8 \rfloor}$. Furthermore, we define the following adjacencies.

$$\{u_i, s_{i',j}\} \in E, \forall j, \lfloor i/8 \rfloor = \lfloor i'/8 \rfloor, i \neq i', \tag{6}$$

$$\{u_i, s'_j\}, \{u_i, s''_j\} \in E, \forall i, j \text{ and } u_i \text{ does not yield a false clause}, \tag{7}$$

$$\{u_i, s_{i',\lfloor i/8 \rfloor}\} \in E, \forall i, i' \text{ and the assignments of the clauses corresponding to } u_i, u_{i'} \text{ do not contradict.} \tag{8}$$

Again the basic idea is that there will be $m$ vertices $u \in U$ that are not in the optimum $B$ and they correspond to an assignment of the clauses of $I$.

Assume that $I$ is satisfiable and let $Y$ be a solution consisting of literals $y_1, \ldots, y_\ell$ with each $y_i$ of the form $x_i$ or $\overline{x}_i$ and consider the following partition $B_Y \cup (V \setminus B_Y)$, where

$$B_Y = S \cup \{u_i : \text{the truth assignment corresponding to } u_i \text{ differs from } Y\} .$$

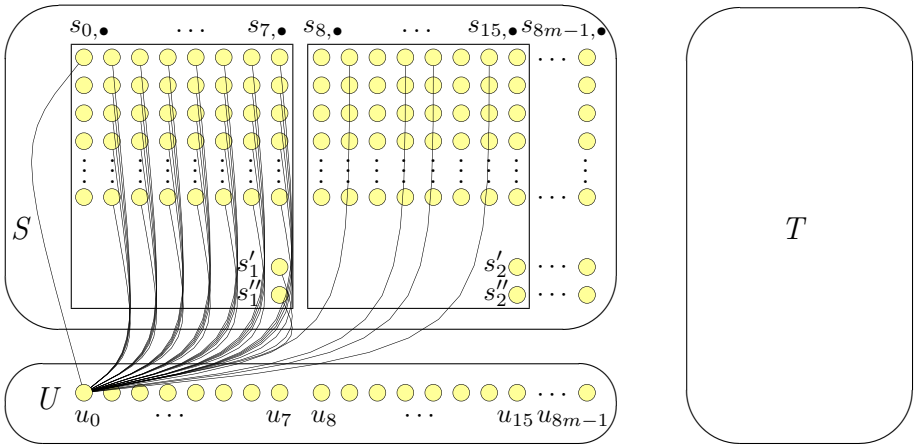


Figure 4: Example for the adjacencies of type (6)–(8) of vertex $u_{1,0}$.

Then every vertex in $B_Y$ has a neighbor in $V \setminus B_Y$. This partition induces

$$\lambda(B_Y) = |V|/2 = 8m^2 + 9m \ .$$

We show that this value can only be achieved, when $I$ is satisfiable. Thus, 3SAT can be solved by the decision problem of finding a partition $B$ with $\lambda(B) \geq 8m^2+9m$. Clearly, the transformation can be done in polynomial time and hence MAX VERTEX BISECTION is $\mathcal{NP}$-complete.

Since all vertices in $B$ must have neighbors in $V \setminus B$ (otherwise $\lambda(B) < 8m^2 + 9m$), none of the vertices of $T$ are in $B$. Vertices $s'_i$ and $s''_i$ assure that exactly one assignment per clause is chosen, otherwise at least one of them would not have a neighbor in $V \setminus B$. All vertices of $B$ have neighbors in $V \setminus B$ if and only if all clause assignments pairwise do not contradict. □

MIN VERTEX BISECTION can also be found with a slightly different definition. Let $B^* \subseteq V$ be a separator, whose removal disconnects $V$ into two sets $V_1$ and $V_2$ of equal size, where $|B^*|$ has to be minimized.

Theorem 1 also holds for this definition, which can be seen by increasing the size of clique $S$ by $15.5m^2 + 7.5m$ vertices. Analogously modifying Definition 2 does not make sense, since $B^* = V$ would always be the maximum.

# 3 Special Graph Classes

Among the graph classes for which there exist efficient algorithms for edge bisection are e.g. hypercubes [14], trees [12], cube-connected cycles graphs [13], some meshes [16], partial $k$-trees [17] and butterfly networks [2].

We show for two graph classes, hypercubes and trees that vertex bisection also becomes tractable.

### 3.1 Hypercubes

The vertices of the $d$-dimensional hypercube $Q_d$ can be represented by all bit strings of length $d$ and two vertices are connected if and only if their strings differ in exactly one position. Let $v_i$ denote bit $i$ of a vertex $v \in Q_d$ and $|v| = |\{i : v_i = 1\}|$.

**Theorem 3** *The minimum vertex bisection width of a $d$-dimensional hypercube $Q_d$ is $\binom{d}{\lfloor d/2 \rfloor}$. A minimum cut $B = B_d$ is given by*

$$B_d = \begin{cases} \{v \in Q_d : |v| < d/2\} & \text{, if } d \text{ is odd,} \\ \{v \in Q_d : |v| < d/2 \text{ or } (|v| = d/2 \text{ and } v_1 = 0)\} & \text{, if } d \text{ is even.} \end{cases}$$

This is a special case of the general isoperimetric problem on hypercubes. Minimum cuts of arbitrary sizes are given by the first vertices of the order $<_\ell$, where $v <_\ell w$ if $|v| < |w|$ or $|v| = |w|$ and $v$ precedes $w$ lexicographically. Proofs can be found in e.g. [8, 10].

### 3.2 Trees

**Theorem 4** *The minimum vertex bisection width of trees can be computed in polynomial time.*

**Proof:** Algorithm 1 computes the minimum vertex bisection width in $\mathcal{O}(n^2)$, where $n = |T|$ $(= |V|)$ is the number of vertices of the tree $T = (V, E)$. It is a dynamic programming approach related to the corresponding algorithm for edge bisection on $k$-trees [17]. Let $B^* \subseteq B$ be the set of vertices in $B$ that have a neighbor in $V \setminus B$, i.e. $B^*$ is the separator. Note that Algorithm 1 does not only compute the minimum vertex bisection width, but even all minimum vertex widths $\lambda(B)$, where $|B| = 0, \ldots, |T|$.

We prove the running time of Algorithm 1 by induction on the height $h$ of the tree after having picked its root $v$.
$\underline{h = 1:}$ $T$ consists of $v$ and its $n-1$ children. The three minima in the inner loop have to be computed over 9 times 2 elements altogether, leading to a running time $T(n) \leq 18(n-1)(n+1) < 18n^2$.
$\underline{h > 1:}$ Let $n_1, \ldots, n_k$ be the sizes of the subtrees rooted by children of $v$, such that $1 + \sum_{i=1}^{k} n_i = n$. The three minima in the first instance of min_widths now have to be computed over at most 9 times $n_i + 1$ elements for the subtree $T_i$ altogether. A better upper bound is 9 times $\min\{n_i + 1, r + 1, 1 - r + \sum_{j=1}^{i} n_j\}$.

**Algorithm 1**: minimum vertex widths on trees.

**Input**: tree $T = (V, E)$
**Output**: minimum vertex widths $\lambda_r = \lambda(B)$ with $|B| = r$ $(r = 0, \ldots, |T|)$

**begin**
  pick a vertex $v$ as root
  set $n \leftarrow |T|$ as globally known constant
  let $A_1, A_2, A_3$ be arrays with index set $\{0, \ldots, n\}$, where
  $A_1[r]$ will denote the minimum $\lambda(B)$ for $|B| = r$ and the case $v \in B^*$,
  $A_2[r]$ for the case $v \in B \setminus B^*$ and $A_3[r]$ for the case $v \notin B$
  $(A_1, A_2, A_3) \leftarrow$ min_widths$(T, v)$
  **for** $r = 0, \ldots, n$ **do** $\lambda_r \leftarrow \min\{A_1[r], A_2[r], A_3[r]\}$
**end**

**min_widths(**$T, v$**)**
**begin**
  let $A_1, A_2, A_3$ be arrays with index set $\{0, \ldots, |T|\}$
  set $A_3[0] \leftarrow 0, A_2[1] \leftarrow 0$ and all remaining values to $n + 1$
  **if** *$v$ is a leaf* **then**
    **return** $(A_1, A_2, A_3)$
  **else**
    let $M_1, M_2, M_3$ be arrays with index set $\{0, \ldots, |T|\}$
    let $T_1, \ldots, T_k$ be all trees rooted by children $v_1, \ldots, v_k$ of $v$
    **for** $i = 1, \ldots, k$ **do**
      let $B_1, B_2, B_3$ be arrays with index set $\{0, \ldots, |T_i|\}$
      $(B_1, B_2, B_3) \leftarrow$ min_widths$(T_i, v_i)$
      **for** $r = 0, \ldots, |T|$ **do**
        $M_1[r] \leftarrow \min\{A_2[x] + B_3[y] + 1, A_1[x] + B_1[y],$
        $A_1[x] + B_2[y], A_1[x] + A_3[y] \,:\, x + y = r\}$
        $M_2[r] \leftarrow \min\{A_2[x] + B_1[y], A_2[x] + B_2[y] \,:\, x + y = r\}$
        $M_3[r] \leftarrow \min\{A_3[x] + B_1[y], A_3[x] + B_2[y] + 1,$
        $A_3[x] + B_3[y] \,:\, x + y = r\}$
      $A_1 \leftarrow M_1, A_2 \leftarrow M_2, A_3 \leftarrow M_3$
    **return** $(A_1, A_2, A_3)$
**end**

This yields a running time

$$\begin{aligned} T(n) &< 18\sum_{i=1}^{k} n_i^2 + 9\sum_{i=1}^{k}(n_i+1)\Big(\Big(\sum_{j=1}^{i-1} n_j\Big)+1\Big) \\ &\leq 18\sum_{i=1}^{k} n_i^2 + 9\sum_{i=1}^{k} n_i\Big(\sum_{j=1}^{i-1} n_j\Big) + 9\sum_{i=1}^{k}\sum_{j=1}^{i-1} n_j + 9(n-1) + 9k \\ &\leq 18\sum_{i\leq j} n_i n_j + 18(n-1) \;<\; 18n^2 \;. \end{aligned}$$

The correctness of Algorithm 1 follows from the property that the minimum vertex widths (for cuts $B$ of size $r$) can first be computed separately for each subtree on a certain level of the tree (for each subtree separately for the 3 cases that the root of the subtree is in $B^*$, in $B \setminus B^*$ or not in $B$) and these minimum vertex widths can then be merged for the next higher level. A minimum cut of size $r$ on a certain level is the union of the minimum cuts of sizes $r_1, \ldots, r_k$ of its subtrees $T_1, \ldots, T_k$, where $r = r_1 + \ldots + r_k$ (plus 1 for the root in some cases). □

Note that Algorithm 1 can easily be modified such that it does not only return the minimum vertex widths, but also one (or all) corresponding cuts.

When applying Algorithm 1 to some randomly chosen tree $T$, all minimum vertex widths will probably be less than 3, since for a random tree it is very likely that there exist two disjoint subtrees (or even only one) whose sizes add up to any $r = 0, \ldots, |T|$. This implies a minimum vertex width of less than 3 (or less than 2).

A tree that has some a minimum cut of size $r$ with vertex width of 3 or greater has to contain at least 4 vertices with degree greater 2. Otherwise it is easy to verify that there exists a sequence $B_0 \subseteq B_1 \subseteq \ldots \subseteq B_{|T|}$ with $\lambda(B_i) < 3$. An example of a tree with minimum vertex bisection 3 is given in Figure 5. Theorem 5 states the existence of trees with greater minimum vertex bisection width.

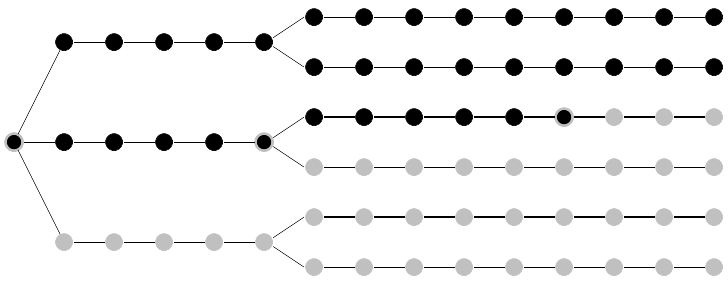

Figure 5: A tree with minimum vertex bisection width 3.

**Theorem 5** *The minimum vertex bisection of a complete $d$-tree $T$ of height $h > 0$ is at least $h - x(h)$, where $x(h) = \min\{x \in \mathbb{N} : \sum_{k=0}^{x}(d^k + 1) \geq h\}$, if $d \geq 3$ odd, and equal to 1, otherwise. Moreover, $x(h) \leq \lceil \log_d(h-1) \rceil$ for $h > 1$.*

**Proof:** Minimum vertex bisection can be bounded in a way similar to minimum edge bisection on trees, see [15]. The case $d$ even is trivial since a minimum bisection consists of the root together with the *left part* of $T$. The case $d = 1$ or $h = 1$ is also trivial. For odd $d \geq 3$ and $h = 2$ consider that there is no single vertex in $T$ that separates the vertex set evenly. For the remaining case $d \geq 3$ odd and $h > 2$ consider Figure 6. Black vertices correspond to vertices in $B$. First note that for a minimum bisection we can assume for each level of $T$ that no grey vertex is followed (from left to right) by a black vertex (by top-down reordering of subtrees). Hence, given a minimum bisection $B$, we can define a number $n_k$ for each level $k$ denoting the position of the rightmost black vertex, such that $\sum_{k=0}^{h} n_k \in \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$. The vertices of the vertical path from the root downwards denote the center vertices of each level. All vertices on the right hand side of that path are grey. Dotted lines denote paths delimiting subtrees, dashed lines denote vertices on the same level. We can assume that

$$n_{k-1} \geq \lceil (n_k - 1)/d \rceil, \tag{9}$$

since otherwise rightmost black vertices can be moved (bottom-up) without increasing the vertex width . Define $n'_h := (d^h - d^{x(h)+1})/2$, i.e. the position of the vertex of the lowest level directly left of the subtree rooted by the center vertex of level $h - x(h) - 1$. Let $B$ be a minimum bisection and assume $n_h > d^h/2$. Then Equation (9) already implies $|B| > \lceil n/2 \rceil$. So, let $n'_h < n_h < d^h/2$. Equation (9) implies that all vertices denoted by disk-shaped black vertices in Figure 6 are in $B$, such that part of the vertices of $B$, say $B'$, are already fixed by this choice of $n_h$. Now, $B'$ taken alone already induces a vertex width equal to the lower bound to be proven minus 1. All rightmost vertices of $B'$ not above level $h - x(h)$ do not have any grey neighbors, and there are too few vertices in $B \setminus B'$ to achieve this for any of the remaining rightmost vertices of $B'$ (without creating another black vertex with grey neighbors). For $d \geq 5$ it is clear that since $B \setminus B' \neq \emptyset$ the vertex width of $B$ is at least by one greater than that of $B'$. For $d = 3$ the remaining vertices $B \setminus B'$ can be set directly right of the center vertices of levels 1 to $h - x(h) - 2$ without changing the vertex width (these center vertices only have one grey neighbor). But since $|B \setminus B'| \geq h - x(h)$ again the vertex width of $B$ is at least by one greater than that of $B'$. So, finally assume that $n_h \leq n'_h$. Since $x(h) \geq \log_d(h-1) - 1$ it follows

$$\begin{aligned}
\lambda(B) &\geq \sum_{k=0}^{h-1} \Big( n_k - \lfloor n_{k+1}/d \rfloor \Big) \geq \lfloor n/2 \rfloor - n_h - \sum_{k=0}^{h-1} n_{k+1}/d \\
&\geq \lfloor n/2 \rfloor - n'_h - (\lceil n/2 \rceil - n_0)/d \geq (1 - 1/d)n/2 - 1/2 - n'_h - 1/(2d) \\
&\geq \frac{d^{h+1} - 1}{2d} - 1/2 - d^h/2 + d^{x(h)+1} - 1/(2d) \geq h - 11/6,
\end{aligned}$$

such that $\lambda(B) \geq h - x(h)$, since $x(h) \geq 1$ for $h > 2$. □

For odd $d \geq 5$ the lower bound given in Theorem 5 is sharp for heights of the form $h := \sum_{k=0}^{x}(d^k + 1)$ for some $x \in \mathbb{N}$, where a minimum bisection is given according to Figure 6. Since $h$ is even, the number of vertices $n$ is odd and the bisection in Figure 6 fulfills $|B| = (n-1)/2$. Hence, Theorem 5 remains sharp also for height $h+1$ and $h+2$. For $d = 3$ the given lower bound is sharp in general. A minimum bisection is given according to Figure 6 plus some of the vertices directly right of the center vertices of levels 1 to $h - x(h) - 1$, such that $|B| \in \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$.

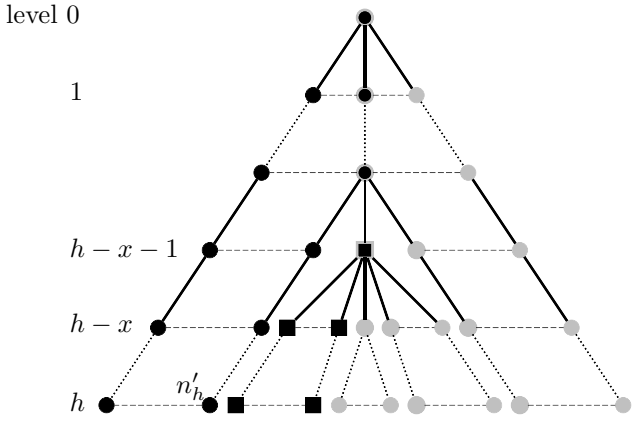


Figure 6: Minimum bisection of a $d$-tree of a height $h = \sum_{k=0}^{x}(d^k + 1)$ for some $x \in \mathbb{N}$ and some odd $d \geq 3$.

# 4 Discussion

We proved $\mathcal{NP}$-completeness of MIN VERTEX BISECTION and MAX VERTEX BISECTION. This completes a list of eight graph layout problems given in [4], all of which are now classified as $\mathcal{NP}$-complete. For two graph classes relevant for communication networks, hypercubes and trees, the minimum vertex bisection width can either be given directly or be computed in polynomial time.

## References

[1] C. J. Alpert and A. B. Kahng. Recent directions in netlist partitioning: A survey. *Integration: the VLSI journal*, 19:1–81, 1995.

[2] C. F. Bornstein, A. Litman, B. M. Maggs, R. K. Sitaraman, and T. Yatzkar. On the bisection width and expansion of butterfly networks. *Theory Comput. Systems*, 34:491–518, 2001.

[3] T. Bui, S. Chaudhuri, F. T. Leighton, and M. Sipser. Graph bisection algorithms with good average case behavior. *Combinatorica*, 7:171–191, 1987.

[4] J. Diaz, J. Petit, and M. Serna. A survey of graph layout problems. *ACM Comput. Surv.*, 34:313–356, 2002.

[5] U. Feige and R. Krauthgamer. A polylogarithmic approximation of the minimum bisection. In *IEEE Symp. Foundations of Computer Science*, pages 105–115, 2000.

[6] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified np-complete graph problems. *Theor. Comput. Sci.*, 1:237–267, 1976.

[7] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.*, 42:1115–1145, 1995.

[8] L. H. Harper. Optimal numberings and isoperimetric problems on graphs. *J. Comb. Theory*, 1(3):385–393, 1966.

[9] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103, 1972.

[10] G. O. H. Katona. The hamming-sphere has minimum boundary. *Studia Scient. Math. Hungarica*, 10:131–140, 1975.

[11] R. Klasing. The relationship between gossiping in vertex-disjoint paths mode and bisection width. *Discrete Appl. Math.*, 83(1-3):227–244, 1998.

[12] R. M. MacGregor. On Partitioning a Graph: A Theoretical and Empirical Study, PhD Thesis, University of California, Berkeley, 1978.

[13] Y. Manabe, K. Hagihara, and N. Tokura. The minimum bisection widths of the cube-connected cycles graph and cube graph. *Transactions of the Institute of Electronics and Communication Engineers of Japan*, J67-D(6):647–654, 1984.

[14] K. Nakano. Linear layouts of generalized hypercubes. *Intl. J. Foundations of Computer Science*, 14:137–156, 2003.

[15] L. Palios. Upper and lower bounds for optimal tree partitions. Technical Report GCG 47, The Geometry Center, University of Minnesota, 1994. Available from `http://www.geom.uiuc.edu/docs/preprints/online/GCG47.html`.

[16] J. Rolim, J. Tvrdik, J. Trdlička, and I. Vrťo. Bisecting de Bruijn and Kautz graphs. *Discrete Applied Mathematics*, 85:87–97, 1998.

[17] K. Soumyanath and J. S. Deogun. On the bisection width of partial $k$-trees. In *Congressus Numerantium*, volume 74, pages 25–37, 1990. (Proc. 20th Southeastern Conf. on Combinatorics, Graph Theory, and Computing).

[18] M. Stoer and F. Wagner. A simple min cut algorithm. In *Proc. Europ. Symp. Algorithms*, volume 855 of *Springer LNCS*, pages 141–147, 1994.