PLOS | ONE

RESEARCH ARTICLE

On the Treewidths of Graphs of Bounded Degree

Yinglei Song*, Menghong Yu*

School of Electronics and Information Science Jiangsu University of Science and Technology Zhenjiang, Jiangsu 212003, China

* yingleisong@gmail.com (YS); ymhzj2691@163.com (MY)

Abstract

In this paper, we develop a new technique to study the treewidth of graphs with bounded degree. We show that the treewidth of a graph $G = (V, E)$ with maximum vertex degree d is at most $(1-Ce^{-4.06\sqrt{d}})|V|$ for sufficiently large d, where C is a constant.

Citation: Song Y, Yu M (2015) On the Treewidths of Graphs of Bounded Degree. PLoS ONE 10(4): e0120880. doi:10.1371/journal.pone.0120880

Academic Editor: M. Sohel Rahman, Bangladesh University of Engineering and Technology, BANGLADESH

Received: March 4, 2014

Accepted: February 10, 2015

Published: April 7, 2015

Funding: This work is fully supported by the New Faculty Start-up Funding at Jiangsu University of Science and Technology, China, under the funding number 635301202. The funder had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

Introduction

Tree decomposition is one of the most important concepts developed in graph theory in the past two decades. In a tree decomposition, graph vertices are grouped into vertex subsets, each of the vertex subsets is represented with a single node and all nodes are connected into a tree. Tree decomposition has provided original but profound insights into structural properties of graphs. For example, tree decomposition is the fundamental tool in the proof of the graph minor theorem [13, 14, 15, 16, 17, 18]. On the other hand, tree decomposition also has important applications in algorithm design and complexity research. A generic dynamic programming framework has been available for solving many NP-hard problems on graphs using tree decomposition [2]. Based on this framework, important algorithmic and complexity results have been found for some graph theoretic optimization problems [3, 9].

Fig. 1(b) shows a tree decomposition of the graph in Fig. 1(a). Tree decomposition provides a new topological view in the structure of a graph and has been shown to be related to many deep properties related to graph minors [14, 16]. These properties have played fundamental roles in the proof of the graph minor theorem. Moreover, tree decomposition also has important implications in algorithm design. For example, treewidth provides a structure parameter for a graph and many NP-hard graph optimization problems can be efficiently solved when the treewidth of the underlying graph is small [2]. More specifically, given a tree decomposition of a graph, we can easily identify subproblems for many NP-hard optimization problems and partial optimal solutions for these subproblems can be extended or combined with an exhaustive search performed only on vertices in a single tree node. As a result, given a tree decomposition, a dynamic programming approach can be employed to solve these optimization problems.

Treewidth is a structural parameter often used to evaluate a tree decomposition. The treewidth of a tree decomposition is determined by the maximum number of vertices contained in a single tree node and the treewidth of a graph is the minimum treewidth over all its tree

decompositions. Based on the aforementioned dynamic programming framework, some NP-hard optimization problems including Maximum Independent Set and Minimum Dominating Set can be solved in time $O(f(t)|V|)$, given a tree decomposition of treewidth t for the underlying graph $G = (V, E)$ [2]. However, since $f(t)$ is in general an exponential function of t, the treewidth of the available tree decomposition often determines the computational efficiency of this dynamic programming method.

A well known example is that, a maximum independent set in a graph $G = (V, E)$ can be found in time $O(2^k|V|)$ based on a tree decomposition of treewidth k for G [2]. A more generic result states that, any graph optimization problem that can be formulated with Monadic Second Order (MSO) logic can be solved in polynomial time when a tree decomposition of bounded treewidth is available for the underlying graph [8]. Treewidth is often important for the computational efficiency of these algorithms in the sense that the most computationally intensive part in such an algorithm arises from an exponential function of the treewidth.

Computing the treewidth of a graph is an NP-hard problem [1]. Due to the difficulty of finding the treewidth of a graph, a few efficient algorithms and methods have been developed to estimate its upper bounds and lower bounds [4, 5, 6, 12]. For some graph families, an upper bound for treewidth can be estimated from other structural parameters with a simple formula.

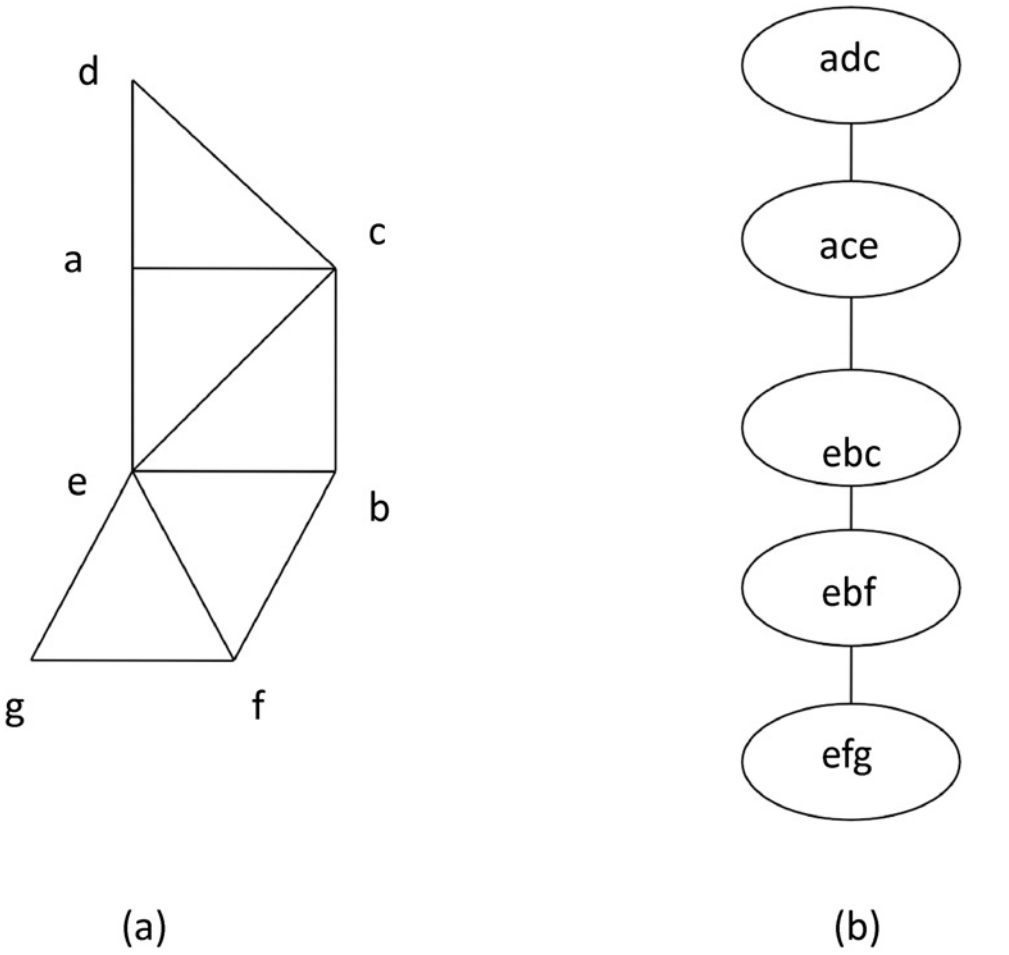

Fig 1. An example of graph tree decomposition. (a) a graph; (b) a tree decomposition of the graph in (a).

doi:10.1371/journal.pone.0120880.g001

For example, the treewidth of a planar graph is bounded from above by $3l + 1$ if the diameter of the graph is l [3, 9]. The treewidth of a graph with m edges has been shown to be less than $m/5$ [10]. In [7], an upper bound for treewidths are obtained for k-chordal graphs, where k is a constant. An improvement of this bound is recently provided in [11]. However, due to the difficulty of relating treewidth to other structural parameters, new tools are needed to obtain similar results for some graph families, such as graphs with bounded degree.

In this paper, we consider graphs of bounded degree and develop a nontrivial asymptotic upper bound for the treewidths of such graphs. Based on a new technique, α *tree backbone*, we show that the treewidth of a graph with maximum vertex degree d is asymptotically bounded from above by $(1 - Ce^{-4.06\sqrt{d}})|V|$ for large enough d, where C is a constant. The result shows that the technique of α tree backbone can be effectively used to provide new insights into the structural properties of graphs of bounded degree.

Preliminaries

The graphs in this paper are undirected graphs without loops. For a given graph $G = (V, E)$ and a vertex $v \in V$, $N(v)$ is the set of vertices that are connected to v by an edge in G and $|N(v)|$ is the *degree* of v. $N[v]$ denotes $\{v\} \cup N(v)$. The *degree* of a graph is the maximum degree of all its vertices. To simplify the notation, we use $G - v$ to represent the graph obtained by removing v and all the edges incident to v from G. For a vertex subset U, we use $G - U$ to denote the graph obtained from G by removing all vertices in U and the edges incident to them from G. For a subset $U \subseteq V$, $N(U)$ denotes the set of all vertices that are not in u but are joined by an edge to at least one vertex in U; $G[U]$ is the subgraph induced by U in G. We use $d(G)$ to denote the size of the minimum independent dominating set in graph G. A *path* in a graph is a sequence of vertices $v_1, v_2, \cdots, v_l$ such that there is a graph edge between v_i and v_{i+1} $(1 \leq i < l)$. We use $P = (v_1, v_2, \cdots, v_l)$ to represent a path P of length l. A *cycle* of length l is a sequence of vertices v_1, $v_2, \cdots, v_l$ such that there is a graph edge between v_i and $v_{(i+1) \bmod l}$.

Definition 1 [13] Let $G = (V, E)$ be a graph, where V is the set of vertices in G, E denotes the set of edges in G. The pair (T, X) is a *tree decomposition* of graph G if it satisfies the following conditions:

1. $T = (I, F)$ defines a tree, the sets of vertices and edges in T are I and F respectively,
2. $X = \{X_i | i \in I, X_i \subseteq V\}$, and $\forall u \in V, \exists i \in I$ such that $u \in X_i$,
3. $\forall (u, v) \in E, \exists i \in I$ such that $u \in X_i$ and $v \in X_i$,
4. $\forall i, j, k \in I$, if k is on the path that connects i and j in tree T, then $X_i \cap X_j \subseteq X_k$.

The *treewidth of the tree decomposition* (T, X) is defined as $\max_{i \in I} |X_i| - 1$. The *treewidth of the graph* G is the minimum treewidth over all possible tree decompositions of G.

Definition 2 Let $G = (V, E)$ be a graph, a vertex subset $F \subseteq V$ is a *feedback vertex set* if graph $G - F$ is acyclic. The *minimum feedback vertex set* in a G is the feedback vertex set with the minimum number of vertices.

An Asymptotic Upper Bound for TreeWidth

Lemma 1 If F is a feedback vertex set for graph $G = (V, E)$, the treewidth of G is bounded by $|F|$.

Proof. It is not difficult to see that since $G - F$ is a tree, its treewidth is bounded by 1. Based on such a tree decomposition, we can simply include all vertices in F to every tree node in this tree decomposition and obtain a tree decomposition for G. The treewidth of this tree decomposition is bounded by $|F|$.

Based on Lemma 3, an upper bound of the treewidth of a graph can be obtained from the size of a feedback vertex set in the graph. We focus on finding an upper bound for the minimum size of the feedback vertex set in a graph of degree bounded by d.

Definition 3 For a graph $G = (V, E)$ of degree d, an α *tree backbone* $(0 < \alpha < 1)$ of the graph is a subgraph T of G that satisfies the following:

1. T is a forest that contains a number of trees;
2. for each tree in T that contains more than 2 internal nodes, each internal node is of degree d;
3. for any vertex v of degree d, if $v \notin T$, at least αd vertices in $N(v)$ are nodes of the trees in T.

For a graph with maximum vertex degree d, an α tree backbone is a maximal forest such that each internal node in the forest is a vertex of degree d in G and has at least $(1 - \alpha)d$ child nodes in the tree. We prove in the following lemma that we are able to construct an α tree backbone in polynomial time.

Lemma 2 For a graph $G = (V, E)$ of degree d, given a positive number $\alpha < 1$, there exists an algorithm that can find an α tree backbone T in polynomial time.

Proof. We show that T can be constructed by a greedy algorithm. Initially, we mark all vertices in G to be unselected. We arbitrarily choose an unselected vertex v of degree d and mark v and all vertices in $N(v)$ as selected. We then proceed to vertices in $N(v)$. For each vertex $u \in N(v)$, all of the unselected neighbors of u are marked as selected if it is of degree d and the number of its unselected neighbors is $d - 1$. We recursively apply this procedure until we cannot mark more vertices as selected. We then include all selected vertices in T. For each remaining unselected vertex of degree d, we simply check the number of its neighbors that have been selected. If this number is smaller than αd, we include the vertex and its unselected neighbors in T and mark them as selected. It is not difficult to verify that this greedy algorithm can find an α tree backbone T in polynomial time.

For a given positive $\alpha < 1$ and a graph $G = (V, E)$ of degree at most d, based on an α tree backbone T of the graph, any vertex of degree d that is not in T has at least αd neighbors that are leaves of the trees in T. The following lemma considers only the vertices that are of degree d and not in T.

Theorem 1 Given a graph $G = (V, E)$ of maximum degree d and a positive number $\alpha < 1$, if T is an α tree backbone in G, there exists a vertex subset S that satisfies the following:

1. A connected component in $G - S$ is either a tree or a graph of degree bounded by $d - 1$;
2. $|S| \leq (\frac{2}{(1-\alpha)d} + \frac{2d-1+\sqrt{d}}{\alpha(\sqrt{d}+1)d})|V|$.

Proof. For a given α tree backbone T that contains m vertices, we consider the set of the internal nodes that are connected to a leaf in a tree in T. We denote such a set with I. We assume $|I| = s$ and denote the nodes in I with $v_1, v_2, \cdots, v_s$. In addition, we use $d_1, d_2, \cdots, d_s$ to denote the degree of $v_1, v_2, \cdots, v_s$ in T. It is not difficult to see that the following inequality holds.

$$\sum_{i=1}^{s} d_i \leq 2(m-1) < 2m \tag{1}$$

The inequality holds since each edge is counted at most twice in the summation and there are at most $m - 1$ edges in T. On the other hand, since the degree of each node in I is at least $(1 - \alpha)d$, we also have the following inequality.

$$\sum_{i=1}^{s} d_i \geq (1-\alpha)ds \tag{2}$$

Therefore, we can obtain

$$s < \frac{2m}{(1-\alpha)d} \tag{3}$$

We thus have $|I| < \frac{2m}{(1-\alpha)d}$. We now consider the vertices that are of degree d and not in T. We assume that such vertices form a vertex set V_1 and their neighbors in T form vertex set V_2. The bipartite subgraph formed between V_1 and V_2 is $H = (V_1 \cup V_2, F)$, such that for vertices $u \in V_1$ and $v \in V_2$, $(u, v) \in F$ if and only if $(u, v) \in E$. The degree of vertices in V_1 in H is at least αd. Note that we can assume the degree of each vertex in V_2 in H is at most $d - 1$, since each such vertex is connected to at least one vertex in T by an edge and its degree is at most d in G.

We now show that H contains a dominating set D of size no larger than $\frac{2d-1+\sqrt{d}}{\alpha(\sqrt{d}+1)d}m$. To see this, we apply the following operations to vertices in V_2. If $v \in V_2$ and it's degree in H is larger than a given positive number c, we simply include v in D and remove its neighbors in V_1 from H. We repeat this until no remaining vertex in V_2 have degree more than c in H, we then include all the remaining vertices in V_1 in D.

It is not difficult to see that the number of vertices that are in V_2 and included in D is at most $|V_1|/(c + 1)$, for the remaining vertices in V_1 since their degree is at least αd in graph H, we have $\alpha d|V'| \leq c|V_2|$, where V' is the set of remaining vertices in V_1. Combining the two we can obtain that the size of this dominating set is at most $\frac{|V_1|}{c+1} + \frac{c|V_2|}{\alpha d}$. On the other hand, since we have $\alpha d|V_1| \leq (d-1)|V_2|$, we get $|D| \leq \frac{d-1+c(c+1)}{(c+1)\alpha d}|V_2|$. Now if we let $c = \sqrt{d}$ and we obtain a dominating set with its size bounded by $\frac{2d-1+\sqrt{d}}{\alpha(\sqrt{d}+1)d}|V_2|$ from above.

We now consider the set $S = I \cup D$, it is not difficult to see that after removing all vertices in S, each connected component in the resulting graph $G - S$ is either a tree or a graph of degree bounded by $d - 1$. and we have $|S| \leq (\frac{2}{(1-\alpha)d} + \frac{(2d-1+\sqrt{d})}{\alpha(\sqrt{d}+1)d})|V|$. The theorem has been proved.

Based on Theorem 1, we are able to compute an asymptotic upper bound for the size of the minimum feedback vertex set in a graph G with maximum vertex degree d, where d is an integer that is sufficiently large. Specifically, Theorem 1 states that we are able to remove a fraction of approximately $2/\sqrt{d}$ of the vertices from such a graph such that each connected component in the resulting graph is either a tree or a graph with its maximum degree bounded by $d - 1$. We can recursively apply this procedure to each connected component that is not a tree and we show that this procedure can lead to an upper bound estimate for the size of the minimum feedback vertex set.

Lemma 3 Given a graph $G = (V, E)$ of maximum degree d, there exists an independent vertex subset $S \subset V$ such that $|S| \leq |V|/2$ and $G - S$ is a graph of maximum degree $d - 1$.

Proof. To obtain such an S, we can arbitrarily choose a vertex u of degree d and include it in S. We then remove u and the edges incident to it from G and recursively apply the same procedure on the resulting graph until no such vertices can be found. S induces an independent set in G and we have $|E| \geq d|S|$. On the other hand, $|E| \leq d|V|/2$. This leads to $|S| \leq |V|/2$.

Definition 4 Given a graph family F and an algorithm A that can compute a feedback vertex set in a graph in F, the *upper bound function* $u(F, A)$ is defined as $max_{G \in F}\{\frac{B(G)}{|V|}\}$, where $G = (V, E)$ is a graph in F and $B(G)$ is the feedback vertex set computed by A in G.

We consider the family of graphs of maximum degree d. We use $L(d)$ to denote the family. From Theorem 1 and Lemma 3, we are able to develop two algorithms that can compute a feedback vertex set for each graph in $L(d)$. Indeed, after computing the set S as stated in Theorem 1 or Lemma 3 in a graph $G = (V, E) \in L(d)$, each connected component in the resulting graph $G - S$ is either a tree or a graph in $L(d - 1)$. The algorithms terminate if $G - S$ is empty or

does not contain a connected component that is not a tree. Otherwise, each component that is not a tree in $G - S$ is recursively processed. In the rest of the paper, we use A_1 to denote the algorithm developed based on Theorem 1 and A_2 for the algorithm developed based on Lemma 3.

Theorem 2 For a given graph $G = (V, E)$ of maximum degree d, there exists a positive constant C such that for a sufficiently large d, G contains a feed back vertex set with size at most $(1 - Ce^{-4.06\sqrt{d}})|V|$. Such a feedback vertex set can be found in polynomial time.

PROOF. From Theorem 1, we can let $\alpha = 0.99$ and there exists an integer D_0 such that when $d > D_0$, we can find a vertex subset $S \subseteq V$ that satisfies the property that each connected component of $G - S$ is either a tree or a graph with maximum degree of $d - 1$ and $|S| \leq 2.021|V|/\sqrt{d}$. An additional requirement for selecting D_0 is specified later in the proof.

Based on D_0, the following algorithm A_3 can compute a feedback vertex set in a graph $G = (V, E)$.

1. For each connected component M that is not a tree in G and does not contain a vertex of degree larger than D_0, apply A_2 to M to get a feedback vertex set F in M and include all vertices in F into the feedback vertex set;
2. the algorithm returns if G does not contain a connected component that is not a tree and contains at least one vertex of degree larger than D_0. Otherwise it continues to step 3;
3. for each connected component E that is not a tree in G and contains at least one vertex of degree larger than D_0, find a subset S in E as described in the proof of Theorem 1, include all vertices in S into the feedback vertex set and remove S from E;
4. set the resulting graph to be G' and recursively apply the algorithm to G';

From Definition 4, a graph $G = (V, E)$ with maximum degree d has a feedback vertex set of size at most $u(L(d), A_3)|V|$. From the above description of A_3, we obtain $u(L(D_0), A_3) = u(L(D_0), A_1)$. A_3 thus returns a feedback vertex set of size at most $u(L(D_0), A_1)|V|$ in a graph $G = (V, E)$ with maximum degree D_0. From Lemma 3, we immediately obtain that $u(L(d), A_1)$ must satisfy $u(L(d), A_1) \leq (u(L(d-1), A_1) + 1)/2$. If we consider the case where $d = 2$, we find that $u(L(2), A_1) \leq \frac{1}{2}$. For a finite integer D_0, it is obvious that $u(L(D_0), A_1)$ is a positive constant strictly less than 1. Since $u(L(D_0), A_3) = u(L(D_0), A_1)$, $u(L(D_0), A_3)$ is also a positive constant strictly less than 1.

We assume A_3 returns a feedback vertex set of size $U(L(d), A_3)|V_1|$ in graph $G_1 = (V_1, E_1) \in L(d)$. From Theorem 1, it is clear that when $d > D_0$, we can find a vertex subset $S_1 \subseteq V_1$ such that each connected component of $G_1 - S_1$ is either a tree or a graph with maximum degree of $d - 1$ and $|S_1| \leq 2.021|V_1|/\sqrt{d}$. We use $F(G_1 - S_1)$ to denote the feedback vertex set obtained by A_3 on $G_1 - S_1$. Since each connected component of $G_1 - S_1$ is either a tree or a graph in $L(d - 1)$, it is not difficult to see that the following inequality holds

$$|F(G_1 - S_1)| \leq u(L(d-1), A_3)(|V_1| - |S_1|) \tag{4}$$

On the other hand, since A_3 returns $S_1 \cup F(G_1 - S_1)$ as a feedback vertex set in G_1, we have

$$u(L(d), A_3)|V_1| = |F(G_1 - S_1)| + |S_1| \tag{5}$$

$$\leq u(L(d-1), A_3)(|V_1| - |S_1|) + |S_1| \tag{6}$$

$$= (1 - u(L(d-1), A_3))|S_1| + u(L(d-1), A_3)|V_1| \tag{7}$$

Since $u(L(d-1), A_3) \leq 1$ and $|S_1| \leq 2.021|V_1|/\sqrt{d}$, we can immediately obtain the following recursion relation for $u(L(d), A_3)$ when $d > D_0$.

$$u(L(d), A_3) \leq \frac{2.021}{\sqrt{d}}(1 - u(L(d-1), A_3)) + u(L(d-1), A_3) \tag{8}$$

Since the following relation holds for a sufficiently small positive number x

$$1 - x \geq e^{-1.004x} \tag{9}$$

it is clear that we can select sufficiently large D_0 such that when $d > D_0$, we have

$$1 - \frac{2.021}{\sqrt{d+1}} \geq e^{-\frac{2.03}{\sqrt{d+1}}} \tag{10}$$

On the other hand, we have

$$\frac{2.03}{\sqrt{d+1}} < \frac{4.06}{\sqrt{d+1}+\sqrt{d}} \tag{11}$$

$$= 4.06(\sqrt{d+1} - \sqrt{d}) \tag{12}$$

We can thus obtain

$$1 - \frac{2.021}{\sqrt{d+1}} \geq e^{-4.06(\sqrt{d+1}-\sqrt{d})} \tag{13}$$

We can choose an appropriate constant C that satisfies $u(L(D_0), A_3) \leq 1 - Ce^{-4.06\sqrt{D_0}}$. We next show that $u(L(d), A_3) \leq (1 - Ce^{-4.06\sqrt{d}})$ when $d \geq D_0$. Based on the principle of induction, when $d = D_0$ the relation holds due to the selection of constant C. Now, we assume that the relation holds when $d = l$, based on the recursion relation, we have

$$u(L(l+1), A_3) \leq (1 - u(L(l), A_3))(1 - e^{-4.06(\sqrt{l+1}-\sqrt{l})}) + u(L(l), A_3) \tag{14}$$

$$\leq 1 - e^{-4.06(\sqrt{l+1}-\sqrt{l})}(1 - u(L(l), A_3)) \tag{15}$$

$$\leq 1 - Ce^{-4.06\sqrt{l+1}} \tag{16}$$

Based on the principle of induction, we know $u(L(d), A_3)$ is at most $1 - Ce^{-4.06\sqrt{d}}$ when $d \geq D_0$. In addition, from Lemma 2, A_3 needs polynomial time to find such a feedback vertex set.

Here, we need to point out that, based on Lemma 3, we can also construct a recursive relation for A_2. However, this recursion can only lead to an upper bound estimation of $(1 - 1/2^{O(d)})|V|$. This trivial asymptotic upper bound now has been significantly improved using the technique of α tree backbone. Based on the relation between the treewidth and the size of a minimum feedback vertex set, we also obtain an asymptotic upper bound for the treewidth of a graph with bounded maximum vertex degree.

Corollary 1 For a given graph $G = (V, E)$ with maximum vertex degree d, there exists a positive constant C such that for a large enough d, the treewidth of the graph is bounded from above by $(1 - Ce^{-4.06\sqrt{d}})|V|$.

Theorem 2 also provides an asymptotic upper bound for the treewidths of sparse graphs. Specifically, for a graph $G = (V, E)$, where $|E| \leq \Delta|V|$ and Δ is a positive constant independent of G, we have the following theorem.

Theorem 3 Given a graph $G = (V, E)$, where $|E| \leq \Delta|V|$ and Δ is a positive constant independent of G, there exists a positive constant C' such that for a large enough Δ, the treewidth of G is bounded from above by $(1 - C'e^{-8.12\sqrt{\Delta}})|V|$.

Proof. First, since $|E| \leq \Delta|V|$, the number of vertices with degree larger than 4Δ is at most $|V|/2$. We assume the number of such vertices is $\beta|V|$ and we have $0 \leq \beta \leq 1/2$. Now, we remove all these vertices and edges incident to them from G and the resulting graph G' is a graph with maximum vertex degree 4Δ. Apply Theorem 2 to graph G', for large enough Δ, we can obtain a tree decomposition T' for G' with its treewidth at most $(1 - Ce^{-8.12\sqrt{\Delta}})|V'|$, where C is a positive constant. We can thus obtain a tree decomposition T for G by including the removed vertices in each tree node of T'. The tree width of T is at most $\beta|V| + (1 - Ce^{-8.12\sqrt{\Delta}})(1 - \beta)|V|$. Since $\beta \leq 1/2$, the treewidth of T is bounded from above by $(1 - \frac{C}{2}e^{-8.12\sqrt{\Delta}})|V|$.

Conclusions

In this paper, we provided an original perspective on the structure of a graph with bounded degree. We develop a new technique, α tree backbone, to analyze the treewidth for graphs of bounded degree. Our analysis leads to a nontrivial asymptotic upper bound for the treewidth of such graphs.

We have seen in the proof of Theorem 1 that the size of the dominating set D in the bipartite graph H is crucial for our analysis and improvements made on its size can lead to further improved bounds for treewidth. In particular, we conjecture that the treewidth of a graph $G = (V, E)$ with maximum vertex degree d is in fact asymptotically bounded by $(1 - \frac{C}{d^c})|V|$, where C and c are constants. However, the technique based on α tree backbone may not be sufficient to obtain such an upper bound. Our future work may focus on providing a proof or counterexample for this conjecture.

Acknowledgments

The authors are grateful for the constructive comments from the anonymous reviewer on an earlier version of the paper.

Author Contributions

Conceived and designed the experiments: YS. Analyzed the data: YS. Contributed reagents/materials/analysis tools: MY. Wrote the paper: YS MY.

References

1. Arnborg S, Corneil DG, Proskurowski A. Complexity of finding embeddings in a *k*-tree. SIAM Journal on Algebraic and Discrete Methods. 1987; 8: 277–284 doi: 10.1137/0608024
2. Arnborg S, Proskurowski A. Linear time algorithms for NP-hard problems restricted to partial *k*-trees. Discrete Applied Mathematics. 1989; 23: 11–24 doi: 10.1016/0166-218X(89)90031-0
3. Baker BS. Approximate algorithms for NP-complete problems on planar graphs. Journal of the ACM. 1994; 41: 153–180 doi: 10.1145/174644.174650
4. Bodlaender HL. Better algorithms for the pathwidth and treewidth of graphs. Proceedings of the 18th International Colloqium on Automata, Languages and Programming. 1991; Lecture Notes in Computer Science 510: 544–555 doi: 10.1007/3-540-54233-7_162

5. Bodlaender HL. A linear time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing. 1996; 25: 1305–1317 doi: 10.1137/S0097539793251219
6. Bodlaender HL, Koster AMCA, Wolle T. Contraction and treewidth lower bounds. Proceedings of the 12th Annual European Symposium on Algorithms. 2004; 628–639.
7. Bodlaender HL, Thilikos DM. Treewidth for graphs with small chordality. Discrete Applied Mathematics. 1997; 79(1-3): 45–61 doi: 10.1016/S0166-218X(97)00031-0
8. Courcelle B. The monadic second order theory of graphs I: recognisable sets of finite graphs. Information and Computation. 1990; 85(1): 12–75 doi: 10.1016/0890-5401(90)90043-H
9. Eppstein D. Subgraph isomorphism in planar graphs and related problems. Journal of Graph Algorithms and Applications. 1999; 3.3: 1–27 doi: 10.7155/jgaa.00014
10. Kneis J, Mölle D, Richter S, Rossmanith P. Algorithms based on the treewidth of sparse graphs. Proceedings of the 31st Workshop on Graph Theoretic Concepts in Computer Science. 2005;385–396.
11. Kosowski A, Li B, Nisse N, Suchan K. k-chordal graphs: from cops and robber to compact routing via treewidth. Proceedings of the 39th International Colloqium on Automata, Language and Programming. 2012;610–622.
12. Lucena B. A new lower bound for tree-width using maximal cardinality search. SIAM Journal on Discrete Mathematics. 2003; 16: 345–353 doi: 10.1137/S0895480101397773
13. Robertson N, Seymour PD. Graph minors II. algorithmic aspects of tree-width. Journal of Algorithms. 1986; 7: 309–322 doi: 10.1016/0196-6774(86)90023-4
14. Robertson N, Seymour PD. Graph minors III. planar tree width. Journal of Combinatorial Theory Series B. 1984; 36: 49–64 doi: 10.1016/0095-8956(84)90013-3
15. Robertson N, Seymour PD. Graph minors IV. tree width and well-quasi-ordering. Journal of Combinatorial Theory Series B. 1990; 48: 227–254 doi: 10.1016/0095-8956(90)90120-O
16. Robertson N, Seymour PD. Graph minors V. excluding a planar graph. Journal of Combinatorial Theory Series B. 1986; 41: 92–114 doi: 10.1016/0095-8956(86)90030-4
17. Robertson N, Seymour PD. Graph minors X. obstructions to tree decomposition. Journal of Combinatorial Theory Series B. 1991; 52: 153–190 doi: 10.1016/0095-8956(91)90061-N
18. Robertson N, Seymour PD. Graph minors XIII. the disjoint paths problems. Journal of Combinatorial Theory Series B. 1995; 63: 65–110 doi: 10.1006/jctb.1995.1006