Can you beat treewidth?

Dániel Marx[*]

Abstract

It is well-known that constraint satisfaction problems (CSP) can be solved in time $n^{O(k)}$ if the treewidth of the primal graph of the instance is at most k and n is the size of the input. We show that no algorithm can be significantly better than this treewidth-based algorithm, even if we restrict the problem to some special class of primal graphs. Formally, let $\mathcal{G}$ be an arbitrary class of graphs and assume that there is an algorithm A solving binary CSP for instances whose primal graph is in $\mathcal{G}$. We prove that if the running time of A is $f(G)n^{o(k/\log k)}$, where k is the treewidth of the primal graph G and f is an arbitrary function, then the Exponential Time Hypothesis fails. We prove the result also in the more general framework of the homomorphism problem for bounded-arity relational structures. For this problem, the treewidth of the core of the left-hand side structure plays the same role as the treewidth of the primal graph above.

1 Introduction

Constraint satisfaction is a general framework that includes many standard algorithmic problems such as satisfiability, graph coloring, database queries, etc. A constraint satisfaction problem (CSP) consists of a set V of variables, a domain D, and a set C of constraints, where each constraint is a relation on a subset of the variables. The task is to assign a value from D to each variable in such a way that every constraint is satisfied (see Definition 3 for the formal definition). For example, 3SAT can be interpreted as a CSP problem where the domain is $\{0,1\}$ and the constraints in C correspond to the clauses (thus the arity of each constraint is 3).

Due to its generality, solving constraint satisfaction problems is NP-hard if we do not impose any additional restrictions on the possible instances. Therefore, the main goal of the research on CSP is to identify tractable classes and special cases of the general problem. The theoretical literature on CSP investigates two main types of restrictions. The first type is to restrict the *constraint language*, that is, the type of constraints that is allowed. This direction was initiated by the classical work of Schaefer [19] and was subsequently pursued in e.g., [1, 2, 3, 7, 15]. The second type is to restrict the *structure* induced by the constraints on the variables. The *primal graph* (or *Gaifman graph*) of a CSP instance is defined to be a graph on the variables of the instance such that there is an edge between two variables if and only if they appear together in some constraint. If the treewidth of the primal graph is k, then CSP can be solved in time $n^{O(k)}$ [10]. (Here n is the size of the input; in the cases we are interested in, the input size is polynomially bounded by the domain size and the number of variables.) The aim of this paper is to investigate whether there exists any other structural property of the primal graph that can be exploited algorithmically to speed up the search for the solution.

For a class $\mathcal{G}$ of graphs, let CSP($\mathcal{G}$) be the special case of CSP where the primal graph of the instance is assumed to be in $\mathcal{G}$. If $\mathcal{G}$ has bounded treewidth, then CSP($\mathcal{G}$) is polynomial-time solvable. The converse is also true:

Theorem 1 ([12]). *If $\mathcal{G}$ is a recursively enumerable class of graphs, then CSP($\mathcal{G}$) is polynomial-time solvable if and only if $\mathcal{G}$ has bounded treewidth (assuming FPT $\neq$ W[1]).*

The assumption FPT $\neq$ W[1] is a standard hypothesis of parameterized complexity (cf. [6, 9]). Thus bounded treewidth is the only property of the primal graph that can make the problem polynomial-time solvable. However, Theorem 1 does not rule out the possibility that there is some structural property that enables us to solve instances significantly faster than the treewidth-based algorithm of [10]. Conceivably, there can be a class $\mathcal{G}$ of graphs such that CSP($\mathcal{G}$) can be solved in time $n^{O(\sqrt{k})}$ or even in time $n^{O(\log k)}$, if k is the treewidth of the primal graph. The main result of the paper is that this is not possible; the $n^{O(k)}$ time algorithm is essentially optimal, up to an $O(\log k)$ factor in the exponent. Thus, in our specific setting, there is no other structural information beside treewidth that can be exploited algorithmically.

We prove our result under the Exponential Time Hypothesis (ETH) [14]: we assume that there is no $2^{o(n)}$ time algorithm for n-variable 3SAT. This assumption is stronger than FPT $\neq$ W[1]. The main result is the following:

[*]Institut für Informatik, Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany.
dmarx@informatik.hu-berlin.de

Theorem 2. *If there is a recursively enumerable class $\mathscr{G}$ of graphs with unbounded treewidth and a computable function f such that CSP($\mathscr{G}$) can be solved in time $f(G)\|I\|^{o(\mathrm{tw}(G)/\log \mathrm{tw}(G))}$ for instances I with primal graph $G \in \mathscr{G}$, then ETH fails.*

The main technical tool of the proof in [12] is the Excluded Grid Theorem of Robertson and Seymour [18], which states that there is an unbounded function $g(k)$ such that every graph with treewidth at least k contains a $g(k) \times g(k)$ grid as minor. The basic idea of the proof in [12] is to show that CSP($\mathscr{G}$) is not polynomial-time solvable if $\mathscr{G}$ contains every grid and then this result is used to argue that CSP($\mathscr{G}$) is not polynomial for any $\mathscr{G}$ with unbounded treewidth, since in this case $\mathscr{G}$ contains every grid as minor. However, this approach does not work if we want a tighter lower bound, as in Theorem 2. The problem is that the function $g(k)$ is very slow growing, e.g., $o(\log k)$, in the known proofs of the Excluded Grid Theorem [5]. Therefore, if the only property of graphs with treewidth at least k that we use is that they have $g(k) \times g(k)$ grid minors, then we immediately lose a lot: as CSP on the $g(k) \times g(k)$ grid can be solved in time $\|I\|^{O(g(k))}$, no lower bound stronger than $\|I\|^{o(\log \mathrm{tw}(G))}$ can be proved with this approach. Thus we need a characterization of treewidth that is tighter than the Excluded Grid Theorem.

The almost-tight bound of Theorem 2 is made possible by a novel characterization of treewidth that is tight up to a logarithmic factor. This result might be of independent interest. We characterize treewidth by the "embedding power" of the graph in the following sense. Let G and H be connected graphs, and let $G^{(q)}$ be the graph obtained from G by replacing each vertex with a clique of size q and each edge with a complete bipartite graph. If q is sufficiently large, then H is a minor of $G^{(q)}$. For example, $q = 2|E(H)|$ is certainly sufficient (if H has no isolated vertices). However, we show that if the treewidth of G is at least k, then H is a minor of $G^{(q)}$ already for $q = O(|E(H)| \log k/k)$. We prove this result by using the well-known characterizations of treewidth with separators and a $O(\log k)$ integrality gap result for the sparsest cut problem. The main idea of the proof of Theorem 2 is to use the embedding power of a graph with large treewidth to simulate a 3SAT instance efficiently.

We conjecture that Theorem 2 holds in a tight way: the $O(\log \mathrm{tw}(G))$ factor can be removed from the exponent. This seemingly minor improvement would be very important for classifying the complexity of other CSP variants. However, it seems that a much better understanding of treewidth is required before Theorem 2 can be made tight. The very least, it should be settled whether there is a polynomial-time constant-factor approximation algorithm for treewidth.

A large part of the theoretical literature on CSP follows the notation introduced by Feder and Vardi [7] and formulates the problem as a homomorphism between relational structures. This more general framework allows a clean algebraic treatment of many issues. In Section 5, we translate the lower bound of Theorem 2 into this framework (Theorem 18) to obtain a quantitative version of the main result of [12]. That is, the left-hand side classes of structures in the homomorphism problem are not only characterized with respect to polynomial-time solvability, but we prove almost-tight lower bounds on the exponent of the running time.

As observed in [12], the complexity of the homomorphism problem does not depend directly on the treewidth of the left-hand side structure, but rather on the treewidth of its core. Thus the treewidth of the core appears in Theorem 18, the analog of Theorem 2. Furthermore, as in [12], our result applies only if the left-hand side structure has bounded arity. In the unbounded-arity case, issues related to the representation of the structures arise, which change the problem considerably. The homomorphism problem with unbounded arity is far from understood: very recently, new classes of tractable structures were identified [13].

Section 2 summarizes the notation we use. Section 3 presents the new characterization of treewidth. Section 4 treats binary CSP and proves Theorem 2. Section 5 overviews the homomorphism problem and presents the main result in this context.

2 Preliminaries

Constraint satisfaction problems. We briefly recall the most important notions related to CSP. For more background, see e.g., [11, 7].

Definition 3. An instance of a *constraint satisfaction problem* is a triple (V, D, C), where:

- V is a set of variables,
- D is a domain of values,
- C is a set of constraints, $\{c_1, c_2, \dots, c_q\}$. Each constraint $c_i \in C$ is a pair $\langle s_i, R_i \rangle$, where:
  - s_i is a tuple of variables of length m_i, called the *constraint scope*, and
  - R_i is an m_i-ary relation over D, called the *constraint relation*.

For each constraint $\langle s_i, R_i \rangle$ the tuples of R_i indicate the allowed combinations of simultaneous values for the variables in s_i. The length m_i of the tuple s_i is called the *arity* of the constraint. A *solution* to a constraint satisfaction problem instance is a function f from the set of variables V to the domain of values D such that for each constraint $\langle s_i, R_i \rangle$ with $s_i = \langle v_{i_1}, v_{i_2}, \dots, v_{i_m} \rangle$, the tuple $\langle f(v_{i_1}), f(v_{i_2}), \dots, f(v_{i_m}) \rangle$ is a member of R_i. We say that an instance is *binary* if each constraint relation is binary, i.e., $m_i = 2$ for each constraint. In this paper, we consider only binary instances. It can be assumed that the instance does not contain two constraints $\langle s_i, R_i\rangle$, $\langle s_j, R_j\rangle$ with $s_i = s_j$, since in this case the two constraints can be replaced with the constraint $\langle s_i, R_i \cap R_j\rangle$.

In the input, the relation in a constraint is represented by listing all the tuples of the constraint. We denote by $\|I\|$ the size of the representation of the instance $I = (V,D,C)$. For binary constraint satisfaction problems, we can assume that $\|I\| = O(V^2D^2)$: by the argument in the previous paragraph, we can assume that there are $O(V^2)$ constraints and each constraint has a representation of length $O(D^2)$. Furthermore, it can be assumed that $|D| \le \|I\|$: elements of D that do not appear in any relation can be removed.

Let $I = (V,D,C)$ be a CSP instance and let $V' \subseteq V$ be a nonempty subset of variables. The *CSP instance* $I[V']$ *induced by* V' is $I' = (V', D, C')$, where C' is defined the following way: For each constraint $c = \langle (v_1,\dots,v_k), R\rangle$ having at least one variable in V', there is a corresponding constraint c' in C'. Suppose that $v_{i_1},\dots,v_{i_\ell}$ are the variables among $v_1,\dots,v_k$ that are in V'. Then the constraint c' is defined as $\langle (v_{i_1},\dots,v_{i_\ell}), R'\rangle$, where the relation R' is the projection of R to the components $i_1,\dots,i_\ell$, that is, R' contains an ℓ-tuple $(d'_1,\dots,d'_\ell) \in D^\ell$ if and only if there is a k-tuple $(d_1,\dots,d_k) \in R$ such that $d'_j = d_{i_j}$ for $1 \le j \le \ell$. Clearly, if f is a solution of I, then f restricted to V' is a solution of $I[V']$.

The *primal graph* of a CSP instance $I = (V,D,C)$ is a graph G with vertex set V, where $x, y \in V$ form an edge if and only if there is a constraint $\langle s_i, R_i\rangle \in C$ with $x, y \in s_i$. For a class $\mathscr{G}$ of graphs, we denote by CSP($\mathscr{G}$) the problem restricted to instances where the primal graph is in $\mathscr{G}$.

Graphs. We denote by $V(G)$ and $E(G)$ the set of vertices and the set of edges of the graph G, respectively. A graph H is a *minor* of G if H can be obtained from G by a sequence of vertex deletions, edge deletions, and edge contractions. The following alternative definition will be more relavant to our purposes: a graph H is a *minor* of G if there is a mapping ψ that maps each vertex of H to a connected subset of $V(G)$ such that $\psi(u) \cap \psi(v) = \emptyset$ for $u \ne v$, and if $u, v \in V(H)$ are adjacent in H, then there is an edge in $E(G)$ connecting $\psi(u)$ and $\psi(v)$.

A *tree decomposition* of a graph G is a tuple $(T, (B_t)_{t\in V(G)})$, where T is a tree and $(B_t)_{t\in V(T)}$ a family of subsets of $V(G)$ such that for each $e \in E(G)$ there is a node $t \in V(T)$ such that $e \subseteq B_t$, and for each $v \in V(G)$ the set $\{t \in V(T) \mid v \in B_t\}$ is connected in T. The sets B_t are called the *bags* of the decomposition. The *width* of a tree-decomposition $(T, (B_t)_{t\in V(T)})$ is $\max\{|B_t| \mid t \in V(t)\} - 1$. The *treewidth* $\mathrm{tw}(G)$ of a graph G is the minimum of the widths of all tree decompositions of G. A class $\mathscr{G}$ of graphs is of *bounded treewidth* if there is a constant c such that $\mathrm{tw}(G) \le c$ for every $G \in \mathscr{G}$.

Given a graph G, the *line graph* $L(G)$ has one vertex for each edge of G, and two vertices of $L(G)$ are connected if and only if the corresponding edges in G share an endpoint. The line graph $L(K_k)$ of the complete graph K_k will appear repeatedly in the paper. Usually we denote the vertices of $L(K_k)$ with $v_{\{i,j\}}$ ($1 \le i < j \le k$), where $v_{\{i_1,j_1\}}$ and $v_{\{i_2,j_2\}}$ are adjacent if and only if $\{i_1, j_1\} \cap \{i_2, j_2\} \ne \emptyset$.

3 Embedding in a graph with large treewidth

Given a graph G and an integer q, we denote by $G^{(q)}$ the graph obtained by replacing every vertex with a clique of size q and replacing every edge with a complete bipartite graph on $q+q$ vertices. The mapping ϕ that maps each vertex of G to the corresponding clique of $G^{(q)}$ will be called the *blow-up* mapping from G to $G^{(q)}$. It can be shown that $\mathrm{tw}(G^{(q)}) = \Theta(q \cdot \mathrm{tw}(G))$.

Clearly, if H is a graph with n vertices, then H is a subgraph of $G^{(n)}$. Furthermore, if G has a clique of size k, then H is already a subgraph of $G^{(n/k)}$. Even if G does not have a k-clique subgraph, but it does have a k-clique minor, then H is a minor of $G^{(n/k)}$. Thus a k-clique minor increases the "embedding power" of a graph by a factor of k. The main result of the section is that large treewidth implies a similar increase in embedding power. The following lemma states this formally:

Lemma 4. *There are functions $f_1(G)$, $f_2(G)$, and a universal constant c such that for every $k \ge 1$, if G is a graph with $\mathrm{tw}(G) \ge k$ and H is a graph with $|E(H)| = m \ge f_1(G)$ and no isolated vertices, then H is a minor of $G^{(q)}$ for $q = \lceil cm\log k/k\rceil$. Furthermore, such a minor mapping can be found in time $f_2(G)m^{O(1)}$.*

The value $cm\log k/k$ is optimal up to a $O(\log k)$ factor. To see this, observe that the treewidth of a graph with m edges can be $\Omega(m)$ (e.g., bounded-degree expanders) and if $\mathrm{tw}(G) = k$, then the treewidth of $G^{(q)}$ for $q = \lceil cm\log k/k\rceil$ is $O(m\log k)$. Furthermore, Lemma 4 does not remain true if m is the number of vertices of H (instead of the number of edges). Let H be a clique on m vertices, and let G be a bounded-degree graph on $O(k)$ vertices with treewidth k. It is easy to see that $G^{(q)}$ has $O(q^2k)$ edges, hence H can be a minor of $G^{(q)}$ only if $q^2k = \Omega(m^2)$, that is, $q = \Omega(m/\sqrt{k})$. The requirement $m \ge f_1(G)$ is a technical detail: due to some probabilistic arguments, our embedding technique works only if H is fairly large.

The graph $L(K_k)$, i.e., the line graph of the complete graph plays a central role in the proof of Theorem 4. The proof consists of two parts. In the first part (Section 3.1), we show that if $\mathrm{tw}(G) \ge k$, then a blow-up of $L(K_k)$ is a minor of an appropriate blow-up of G. This part of the proof is based on the characterization of treewidth by balanced separators and uses a result of Feige et al. [8] on the linear programming formulation of separation problems. In the second part (Section 3.2), we show that every graph is a minor of an appropriate blow-up of $L(K_k)$.

3.1 Embedding $L(K_k)$ in G

A *separator* is a partition of the vertices into three classes (A,B,S) $(S \neq \emptyset)$ such that there is no edge between A and B. A k-separator is a separator (A,B,S) with $|S| = k$. Given a set W of vertices and a separator (A,B,S), we say that S is a *balanced separator* (with respect to W) if $|W \cap C| \le |W|/2$ for every connected component C of $G \setminus S$. The treewidth of a graph is closely connected with the existence of balance separators:

Lemma 5 ([17], [9, Section 11.2]).

1. *If $G(V,E)$ has treewidth greater than $3k$, then there is a set $W \subseteq V$ of size $2k+1$ having no balanced k-separator.*

2. *If $G(V,E)$ has treewidth at most $k+1$, then every $W \subseteq V$ has a balanced k-separator.*

The *sparsity* of the separator (A,B,S) (with respect to W) is defined as

$$\alpha^W(A,B,S) = \frac{|S|}{|(A \cup S) \cap W| \cdot |(B \cup S) \cap W|}.$$

We denote by $\alpha^W(G)$ the minimum of $\alpha^W(A,B,S)$ for every separator (A,B,S). It is easy to see that for every G and W, $1/|W|^2 \le \alpha^W(G) \le 1/|W|$. For our applications, we need a set W such that the sparsity is close to the maximum possible, i.e., $\Omega(1/|W|)$. The following lemma shows that the non-existence of a balanced separator can guarantee the existence of such a set W:

Lemma 6. *If $|W| = 2k+1$ and W has no balanced k-separator in a graph G, then $\alpha^W(G) \ge 1/(4k+1)$.*

Proof. Let (A,B,S) be a separator of sparsity $\alpha^W(G)$; without loss generality, we can assume that $|A \cap W| \ge |B \cap W|$, hence $|B \cap W| \le k$. If $|S| > k$, then $\alpha^W(A,B,S) \ge (k+1)/(2k+1)^2 \ge 1/(4k+1)$. If $|S| \ge |(B \cup S) \cap W|$, then $\alpha^W(A,B,S) \ge 1/|(A \cup S) \cap W| \ge 1/(2k+1)$. Assume therefore that $|(B \cup S) \cap W| \ge |S|+1$. Let S' be a set of $k - |S| \ge 0$ arbitrary vertices of $W \setminus (S \cup B)$. We claim that $S \cup S'$ is a balanced separator of W. Suppose that there is a component C of $G \setminus (S \cup S')$ that contains more than k vertices of W. Component C is either a subset of A or B. However, it cannot be a subset of B, since $|B \cap W| \le k$. On the other hand, $|(A \setminus S') \cap W|$ is at most $2k+1-|(B \cup S) \cap W| - |S'| \le 2k+1-(|S|+1)-(k-|S|) \le k$. □

Remark 7. Lemma 6 does not remain true in this form for larger W. For example, if $|W| = 3k$ and W has no balanced k-separator, then $\alpha^W(G)$ can be as small as $O(1/k^2)$.

Let $W = \{w_1, \dots, w_r\}$ be a set of vertices. A *concurrent vertex flow of value ε* is a collection of $|W|^2$ flows such that for every ordered pair $(u,v) \in W \times W$, there is a flow of value ε between u and v, and the total amount of flow going through each vertex is at most 1. A flow between u and v is a weighted collection of $u-v$ paths. A $u-v$ path contributes to the load of vertex u, of vertex v, and of every vertex between u and v on the path. In the degenerate case when $u = v$, vertex $u = v$ is the only vertex where the flow between u and v goes through, that is, the flow contributes to the load of only this vertex.

The maximum concurrent vertex flow can be expressed as a linear program the following way. For $u, v \in W$, let $\mathscr{P}_{uv}$ be the set of all $u-v$ paths in G, and for each $p \in \mathscr{P}_{uv}$, let variable $p^{uv} \ge 0$ denote the amount of flow that is sent from u to v along p. Consider the following linear program:

$$\begin{aligned} &\text{maximize } \varepsilon \\ &\quad \text{s. t.} \\ &\sum_{p \in \mathscr{P}_{uv}} p^{uv} \ge \varepsilon \quad \forall u,v \in W \\ \sum_{(u,v) \in W \times W} &\sum_{p \in \mathscr{P}_{uv} : w \in p} p^{uv} \le 1 \quad \forall w \in V \qquad \text{(LP1)} \\ &p^{uv} \ge 0 \quad \forall u,v \in V, p \in \mathscr{P}_{uv} \end{aligned}$$

The dual of this linear program can be written with variables $\{\ell_{uv}\}_{u,v \in W}$ and $\{s_v\}_{v \in V}$ the following way:

$$\begin{aligned} &\text{minimize } \sum_{v \in V} s_v \\ &\quad \text{s. t.} \\ &\sum_{w \in p} s_w \ge \ell_{uv} \quad \forall u,v \in W, p \in \mathscr{P}_{uv} \ (*) \\ &\sum_{(u,v) \in W \times W} \ell_{uv} \ge 1 \quad (**) \qquad \text{(LP2)} \\ &\ell_{uv} \ge 0 \quad \forall u,v \in W \\ &s_w \ge 0 \quad \forall w \in V \end{aligned}$$

We show that if there is a separator (A,B,S) with sparsity $\alpha^W(A,B,S)$, then (LP2) has a solution with value at most $\alpha^W(A,B,S)$. Set $s_v = \alpha^W(A,B,S)/|S|$ if $v \in S$ and $s_v = 0$ otherwise; the value of such a solution is clearly $\alpha^W(A,B,S)$. For every $u,v \in W$, set $\ell_{uv} = \min_{p \in \mathscr{P}_{uv}} \sum_{w \in p} s_v$ to ensure that inequalities (*) hold. To see that (**) holds, notice first that $\ell_{uv} \ge \alpha^W(A,B,S)/|S|$ if $u \in A \cup S$, $v \in B \cup S$, as every $u-v$ path has to go through at least one vertex of S. Furthermore, if $u,v \in S$ and $u \neq v$, then $\ell_{uv} \ge$

$2\alpha^W(A,B,S)/|S|$ since in this case a $u-v$ paths meets S in at least two vertices. The expression $|(A\cup S)\cap W|\cdot|(B\cup S)\cap W|$ counts the number of ordered pairs (u,v) satisfying $u\in(A\cup S)\cap W$ and $v\in(B\cup S)\cap W$, such that pairs with $u,v\in S\cap W$, $u\neq v$ are counted twice. Therefore,

$$\sum_{(u,v)\in W\times W}\ell_{uv}\geq(|(A\cup S)\cap W|\cdot|(B\cup S)\cap W|)\cdot\frac{\alpha^W(A,B,S)}{|S|}=1,$$

which means that inequality (**) is satisfied.

The other direction is not true: a solution of (LP2) with value α does not imply that there is a separator with sparsity at most α. However, Feige et al. [8] proved that it is possible to find a separator whose sparsity is greater than that by at most a $O(\log|W|)$ factor (this result appears implicitly already in [16]):

Theorem 8 (Feige et al. [8], Leighton and Rao [16]). *If (LP2) has a solution with value α, then there is a separator with sparsity $O(\alpha\log|W|)$.*

We use Theorem 8 to obtain a concurrent vertex flow in a graph with large treewidth. This concurrent vertex flow can be used to find an $L(K_k)$ minor in the blow-up of the graph in a natural way: the flow paths correspond to the edges of K_k.

Lemma 9. *Let G be a graph with $\mathrm{tw}(G)>3k$. There are universal constants $c_1,c_2>0$ such that $L(K_k)^{(\lfloor c_1\log n\rfloor)}$ is a minor of $G^{(\lfloor c_2\log n\cdot k\log k\rfloor)}$, where n is the number of vertices of G.*

Proof. Since G has treewidth greater than $3k$, by Lemma 5, there is a subset W_0 of size at most $2k+1$ that has no balanced k-separator. By Lemma 6, $\alpha^{W_0}(G)\geq 1/(4k+1)\geq 1/(5k)$. Therefore, Theorem 8 implies that the dual linear program has no solution with value less than $1/(c_0 5k\log(2k+1))$, where c_0 is the constant hidden by the big O notation in Theorem 8. By linear programming duality, there is a concurrent flow of value at least $\alpha:=1/(c_0 5k\log(2k+1))$ connecting the vertices of W_0; let p^{uv} be a corresponding solution of (LP1).

Let $W\subseteq W_0$ be a subset of k vertices. For each pair of vertices $(u,v)\in W\times W$, let us randomly and independently choose $\lfloor\ln n\rfloor$ paths $P_{u,v,1}$, …, $P_{u,v,\lfloor\ln n\rfloor}$ of $\mathscr{P}_{uv}$ (here ln denotes the natural logarithm), where path p is chosen with probability

$$\frac{p^{uv}}{\sum_{p'\in\mathscr{P}_{uv}}(p')^{uv}}\leq\frac{p^{uv}}{\alpha}.$$

For each vertex v, the expected number of paths that contain v is at most $\lfloor\ln n\rfloor/\alpha$. By the Chernoff bound, the probability that more than $5\ln n/\alpha$ of the $k^2\ln n$ paths contain v is at most $e^{-\frac{16}{6}\ln n}\leq 1/n^2$. Therefore, with probability at least $1-1/n$, each vertex v is contained in at most $q:=\lfloor(5\ln n)/\alpha\rfloor$ paths. Note that $q\leq\lfloor c_2\log n\cdot k\log k\rfloor$, for an appropriate value of c_2.

Let ϕ be the blow-up mapping from G to $G^{(q)}$. For each path $P_{u,v,i}$ in G, we define a path $P'_{u,v,i}$ in $G^{(q)}$. Let $P_{u,v,i}=p_1p_2\dots p_r$. The path $P'_{u,v,i}$ we define consists of one vertex of $\phi(p_1)$, followed by one vertex of $\phi(p_2)$, …, followed by one vertex of $\phi(p_r)$. The vertices are selected arbitrarily from these sets, the only restriction is that we do not select a vertex of $G^{(q)}$ that was already assigned to some other path $P'_{u',v',i'}$. Since each vertex v of G is contained in at most q paths, the q vertices of $\phi(v)$ are sufficient to satisfy all the paths going through v. Therefore, we can ensure that the $k^2\lfloor\ln n\rfloor$ paths $P'_{u,v,i}$ are pairwise disjoint.

The minor mapping from $L_k^{(\lfloor\ln n\rfloor)}$ to $G_k^{(q)}$ is defined as follows. Let ψ be the blow-up mapping from $L(K_k)$ to $L(K_k)^{(\lfloor\ln n\rfloor)}$, and let $v_{\{1,2\}}$, $v_{\{1,3\}}$ …, $v_{\{k-1,k\}}$ be the $\binom{k}{2}$ vertices of $L(K_k)$, where $v_{\{i_1,i_2\}}$ and $v_{\{j_1,j_2\}}$ are connected if and only if $\{i_1,i_2\}\cap\{j_1,j_2\}\neq\emptyset$. The $\lfloor\ln n\rfloor$ vertices of $\psi(v_{i,j})$ are mapped to the $\lfloor\ln n\rfloor$ paths $P'_{i,j,1}$, …, $P'_{i,j,\lfloor\ln n\rfloor}$. Clearly, the images of the vertices are disjoint and connected. We have to show that this minor mapping maps adjacent vertices to adjacent sets. If $x\in\psi(v_{i_1,i_2})$ and $x'\in\psi(v_{j_1,j_2})$ are connected in $L_k^{(q')}$, then there is a $t\in\{i_1,i_2\}\cap\{j_1,j_2\}$. This means that the paths corresponding to x and x' both contain a vertex of the clique $\psi(w_t)$ in $G^{(q)}$, which implies that there is an edge connecting the two paths. □

With the help of the following proposition, we can make a small improvement on Lemma 9: the assumption $\mathrm{tw}(G)\geq 3k$ can be replaced by the assumption $\mathrm{tw}(G)\geq k$. This will make the result more convenient to use.

Proposition 10. *For every $k\geq 3$, $q\geq 1$, $L(K_{qk})$ is a subgraph of $L(K_k)^{(2q^2)}$.*

Proof. Let ϕ be a mapping from $\{1,\dots,qk\}$ to $\{1,\dots,k\}$ such that exactly q elements of $\{1,\dots,qk\}$ are mapped to each element of $\{1,\dots,k\}$. Let $v_{\{i_1,i_2\}}$ $(1\leq i_1<i_2\leq qk)$ be the vertices of $L(K_{qk})$ and $u^t_{\{i_1,i_2\}}$ $(1\leq i_1<i_2\leq k$, $1\leq t\leq 2q^2)$ be the vertices of $L(K_k)^{(2q^2)}$, with the usual convention that two vertices are adjacent if and only if the lower indices are not disjoint. Let $U_{\{i_1,i_2\}}$ be the clique $\{u^t_{\{i_1,i_2\}}\mid 1\leq t\leq 2q^2\}$. Let us consider the vertices of $L(K_{qk})$ in some order. Vertex $v_{\{i_1,i_2\}}$ is mapped to a vertex of $U_{\{\phi(i_1),\phi(i_2)\}}$ that was not already used for a previous vertex. If $\phi(i_1)=\phi(i_2)$, then $v_{\{i_1,i_2\}}$ is mapped to a vertex $U_{\{\phi(i_1),\phi(i_1)+1\}}$ (where addition is modulo k). It is clear that if two vertices of $L(K_{qk})$ are adjacent, then the corresponding vertices of $L(K_k)^{(2q^2)}$ are adjacent as well. We have to verify that, for a given i_1,i_2, at most $2q^2$ vertices of $L(K_{qk})$ are mapped to the clique $U_{\{i_1,i_2\}}$. As $|\psi^{-1}(i_1)|$ and

$|\psi^{-1}(i_2)|$ are both q, there are at most q^2 vertices $v_{\{j_1,j_2\}}$ with $\psi(j_1) = i_1$, $\psi(j_2) = i_2$. Furthermore, if $i_2 = i_1 + 1$, then there are $\binom{q}{2}$ additional vertices $v_{\{j_1,j_2\}}$ with $\psi(j_1) = \psi(j_2) = i_1$ that are also mapped to $U_{\{i_1,i_2\}}$. Thus at most $2q^2$ vertices are mapped to each clique $U_{\{i_1,i_2\}}$. □

Using Prop. 10 with $q = 3$ improves Lemma 9 the following way:

Lemma 11. *Let G be a graph with $\mathrm{tw}(G) \geq k$. There are universal constants $c_1, c_2 > 0$ such that $L(K_k)^{(\lfloor c_1 \log n \rfloor)}$ is a minor of $G^{(\lfloor c_2 \log n \cdot k \log k \rfloor)}$, where n is the number of vertices of G.* □

3.2 Embedding H in $L(K_k)$

As the second step of the proof of Lemma 4, we show that every (sufficiently large) graph G is a minor of $L(K_k)^{(q)}$ for $q = O(|E(H)|/k^2)$.

Lemma 12. *For every $\ell > 0$ there is a constant $c_\ell > 0$ such that for every graph $G(V,E)$ with $|E| > c_\ell$ and maximum degree at most 3, the vertices of G can be partitioned into ℓ classes $V_1, \ldots, V_\ell$ such that*

1. *$|V_i| \leq 2|V|/\ell$ for every $1 \leq i \leq \ell$, and*
2. *There are at most $4|E|/\ell^2$ edges between V_i and V_j for every $1 \leq i < j \leq \ell$.*

Furthermore, such a partition can be found in polynomial time.

Proof. Consider a random partition of the vertices of G, that is, each vertex is independently put into one of the ℓ classes with uniform probability. There are $\ell + \binom{\ell}{2}$ possible bad events: one of the ℓ classes can be too large, or there can be too many edges between two classes. We show that the probability of each of these bad events is at most $1/\ell^3$, hence with large probability none of these events occur.

The expected size of each class is $|V|/\ell$ with variance $|V|(1/\ell)(1-1/\ell)$ (since the size of a class can be expressed as the sum of independent 0-1 random variables, where the probability of 1 is $1/\ell$). By Chebyshev's Inequality, the probability that class V_i is too large is at most

$$\Pr(|V_i| > 2|V|/\ell) \leq \frac{|V|(1/\ell)(1-1/\ell)}{(|V|/\ell)^2} \leq \frac{\ell}{|V|} \leq \frac{1}{\ell^3},$$

if $|V|$ is sufficiently large.

Let $E_{i,j}$ be the set of edges between V_i and V_j. For fixed i, j, let us bound the probability that $|E_{i,j}|$ is too large. For each edge $e \in E(G)$, let random variable x_e be 1 if $e \in E_{i,j}$ and 0 otherwise; clearly $|E_{i,j}| = \sum_{e \in E(G)} x_e$. The expected value of $|E_{i,j}|$ is $2|E|/\ell^2$. The random variables x_e are not independent, thus we have to take into account the covariance between the variables to estimate the variance. It is easy to see that

$$\mathrm{cov}(x_{e_1}, x_{e_2}) = \begin{cases} (2/\ell^2)(1-2/\ell^2) & \text{if } e_1 \text{ and } e_2 \\ & \text{share two endpoints} \\ 2/\ell^3 - 4/\ell^4 & \text{if } e_1 \text{ and } e_2 \text{ share} \\ & \text{exactly one endpoint,} \\ 0 & \text{otherwise.} \end{cases}$$

Since the degree of every vertex is at most 3, each variable x_e is correlated with at most 5 variables (including itself). Thus the variance of $|E_{i,j}|$ is at most $5|E|(2/\ell^2)(1-2/\ell^2) \leq 10|E|/\ell^2$, if $|E|$ is sufficiently large. Therefore, by Chebysev's Inequality,

$$\Pr(|E_{i,j}| > 4|E|/\ell^2) \leq \frac{10|E|/\ell^2}{(2|E|/\ell^2)^2} \leq \frac{10\ell^2}{2|E|} \leq 1/\ell^3,$$

if $|E|$ is sufficiently large. Thus a random partition satisfies the requirements of the lemma with high probability. □

Lemma 13. *For every $k > 1$ there is a constant $n_k > 0$ such that for every $G(V,E)$ with $|E| > n_k$ and no isolated vertices, the graph G is a minor of $L(K_k)^{(q)}$ for $q = \lceil 108|E|/k^2 \rceil$. Furthermore, a minor mapping can be found in time polynomial in q and the size of G.*

Proof. Let $n_k = c_{\binom{k}{2}}$, where $c_{\binom{k}{2}}$ is the constant from Lemma 12. First we construct a graph $G'(V',E')$ of maximum degree 3 that contains G as a minor. This can be achieved by replacing every vertex v of G with a path on $d(v)$ vertices (where $d(v)$ is the degree of v in G); now we can ensure that the edges incident to v use distinct copies of v from the path. The new graph G' has at most $3|E|$ edges.

We show that G', hence G, is a minor of $L(K_k)^{(q)}$. Lemma 12 gives a partition of the vertices of G' into $\ell := \binom{k}{2}$ classes. Denote by $V_{\{i,j\}}$ $(1 \leq i < j \leq k)$ these classes. Let $v_{\{i,j\}}$ $(1 \leq i < j \leq k)$ be the vertices of $L(K_k)$, and let ϕ be the blow-up mapping from $L(K_k)$ to $L(K_k)^{(q)}$. The minor mapping ψ from G' to $L_k^{(q)}$ is defined the following way. First, if $u \in V_{\{i,j\}}$, then let $\psi(u)$ contain a vertex $\hat{u}$ from $\phi(v_{\{i,j\}})$. Let us enumerate the edges uw of G'. Assume that $u \in V_{\{i_1,j_1\}}$ and $w \in V_{\{i_2,j_2\}}$. If $\{i_1,j_1\} \cap \{i_2,j_2\} \neq \emptyset$, then $\hat{u}$ and $\hat{w}$ are neighbors, since every vertex of $\phi(v_{\{i_1,j_1\}})$ is a neighbor of every vertex of $\phi(v_{\{i_2,j_2\}})$. If $\{i_1,j_1\} \cap \{i_2,j_2\} = \emptyset$ with $i_1 < j_1$ and $i_2 < j_2$, then add a vertex of $\phi(v_{\{i_1,i_2\}})$ to $\psi(u)$ and another vertex of $\phi(v_{\{i_1,i_2\}})$ to $\psi(u)$; these two vertices are neighbors with each other and they are adjacent to $\hat{u}$ and $\hat{v}$. This ensures that $\psi(u)$ is connected for every $u \in V$, and there is an edge between $\psi(u)$ and $\psi(w)$ for every edge uw.

What remains to be shown is that the sets $\phi(v_{\{i,j\}})$ are large enough such that we can ensure that no vertex of

$L_k^{(q)}$ is assigned to more than one $\psi(u)$. Let us count how many vertices of $\phi(v_{\{i,j\}})$ are used when the minor mapping is constructed as described above. First, the image of each vertex v in $V_{\{i,j\}}$ uses one vertex $\hat{v}$ of $\phi(v_{\{i,j\}})$; together these vertices use at most $|V_{i,j}| \le 2|V|/\binom{k}{2}$ vertices from $\phi(v_{\{i,j\}})$. Furthermore, if $i_1 < j_1$ and $i_2 < j_2$, and $\{i_1, j_1\} \cap \{i_2, j_2\} = \emptyset$, then we use 2 vertices of $\phi(v_{\{i_1,i_2\}})$ for each edge between $V_{\{i_1,j_1\}}$ and $V_{\{i_2,j_2\}}$. There are at most $4|E'|/\binom{k}{2}^2$ edges between $V_{\{i_1,j_1\}}$ and $V_{\{i_2,j_2\}}$, which means that we use at most $8|E'|/\binom{k}{2}^2$ vertices of $\phi(v_{\{i_1,i_2\}})$ for each pair $(\{i_1,j_1\},\{i_2,j_2\})$ satisfying $i_1 < j_1$ and $i_2 < j_2$. For a given i_1, i_2, this can hold for at most $(k-1)^2$ pairs $(\{i_1,j_1\},\{i_2,j_2\})$, hence the total number of vertices we use from $\phi(v_{\{i_1,i_2\}})$ is

$$2|V'|/\binom{k}{2} + (k-1)^2 \cdot 8|E'|/\binom{k}{2}^2$$
$$\le 2|V'|/\binom{k}{2} + 32|E'|/k^2 \le 36|E'|/k^2$$
$$\le 108|E|/k^2 \le q,$$

as required. □

Putting together Lemma 11 and Lemma 13, we can prove the main result of the section:

Proof (of Lemma 4). Let $f_1(G) = n_k + k^2 c_1 \log|V(G)|$, where $k = \mathrm{tw}(G)$ and n_k is the constant from Lemma 13. Assume that $|E(H)| = m \ge f_1(G)$. By Lemma 13, H is a minor of $L(K_k)^{(q)}$ for $q := \lceil 108m/k^2 \rceil$ and a minor mapping ψ_1 can be found in polynomial time. Let $n := |V(G)|$ and $q' := \lceil q/\lfloor c_1 \log n \rfloor \rceil$ (where c_1 is the constant from Lemma 11); clearly, H is a minor of $L(K_k)^{(q'\lfloor c_1 \log n \rfloor)}$. We can assume that $c_1 \log n \ge 2$: for smaller n, the lemma automatically holds if we set c sufficiently large. Observe that m is large enough such that $q/\lfloor c_1 \log n \rfloor \ge 1$ holds, hence $q' \le 2q/\lfloor c_1 \log n \rfloor \le 4q/(c_1 \log n)$. By Lemma 11, $L(K_k)^{(\lfloor c_1 \log n \rfloor)}$ is a minor of $G^{(\lfloor c_2 \log n \cdot k \log k \rfloor)}$ and a minor mapping ψ_2 can be found in time $f_2(G)$ by brute force, for some function $f_2(G)$. Therefore, $L(K_k)^{(q'\lfloor c_1 \log n \rfloor)}$ is a minor of $G^{(q'\lfloor c_2 \log n \cdot k \log k \rfloor)}$ and it is straightforward to obtain the corresponding minor mapping ψ_3 from ψ_2. Since $q'\lfloor c_2 \log n \cdot k \log k \rfloor \le (4q/(c_1 \log n)) \cdot c_2 \log n \cdot k \log k = 4(c_2/c_1) q k \log k \le cm \log k/k$ for an appropriate constant c, we have that H is a minor of $G^{\lceil cm \log k/k \rceil}$. The corresponding minor mapping is the composition $\psi_3 \circ \psi_1$. Observe that each step can be done in polynomial time, except the application of Lemma 11, which takes $f_2(G)$ time. Thus the total running time can be bounded by $f_2(G) m^{O(1)}$. □

4 Complexity of binary CSP

In this section, we prove our main result for binary CSP (Theorem 2). The proof relies in an essential way on the Sparsification Lemma:

Theorem 14 (Impagliazzo, Paturi, and Zane [14]). *If there is a $2^{o(m)}$ time algorithm for m-clause 3-SAT, then there is a $2^{o(n)}$ time algorithm for n-variable 3-SAT.*

The main strategy of the proof of Theorem 2 is the following. First we show that a 3SAT formula ϕ with m clauses can be turned into a binary CSP instance I of size $O(m)$ (Lemma 15). By the embedding result of Lemma 4, for every $G \in \mathscr{G}$, the primal graph of I is a minor of $G^{(q)}$ for an appropriate q. This implies that we can simulate I with a CSP instance I' whose primal graph is G (Lemma 16 and Lemma 17). Now we can use the assumed algorithm for CSP($\mathscr{G}$) to solve instance I', and thus decide the satisfiability of formula ϕ. If the treewidth of G is sufficiently large, then the assumed algorithm is much better than the treewidth based algorithm, which translates into a $2^{o(m)}$ algorithm for the 3SAT instance. By Theorem 14, this means that n-variable 3SAT can be solved in time $2^{o(n)}$, i.e., ETH fails.

Lemma 15. *Given an instance of 3SAT with n variables and m clauses, it is possible to construct in polynomial time an equivalent CSP instance with $n+m$ variables, $3m$ binary constraints, and domain size 3.*

Proof. Let ϕ be a 3SAT formula with n-variables and m-clauses. We construct an instance of CSP as follows. The CSP instance contains a variable x_i ($1 \le i \le n$) corresponding to the i-th variable of ϕ and a variable y_j ($1 \le j \le m$) corresponding to the j-th clause of ϕ. Let $D = \{1,2,3\}$ be the domain. We try to describe a satisfying assignment of ϕ with these $n+m$ variables. The intended meaning of the variables is the following. If the value of variable x_i is 1 (resp., 2), then this represents that the i-th variable of ϕ is true (resp., false). If the value of variable y_j is ℓ, then this represents that the j-th clause of ϕ is satisfied by its ℓ-th literal. To ensure consistency, we add $3m$ constraints. Let $1 \le j \le m$ and $1 \le \ell \le 3$, and assume that the ℓ-th literal of the j-th clause is a positive occurrence of the i-th variable. In this case, we add the binary constraint $(x_i = 1 \vee y_j \ne \ell)$: either x_i is true or some other literal satisfies the clause. Similarly, if the ℓ-th literal of the j-th clause is a negated occurrence of the i-th variable, then we add the binary constraint $(x_i = 2 \vee y_j \ne \ell)$. It is easy to verify that if ϕ is satisfiable, then we can assign values to the variables of the CSP instance such that every constraint is satisfied, and conversely, if the CSP instance has a solution, then ϕ is satisfiable. □

If G_1 is a minor of G_2, then an instance with primal graph G_1 can be easily simulated by an instance with primal graph G_2: each variable of G_1 is simulated by a connected set of variables in G_2 that are forced to be equal.

Lemma 16. *Assume that G_1 is a minor of G_2. Given a binary CSP instance I_1 with primal graph G_1 and a minor mapping ψ from G_1 to G_2, it is possible to construct in polynomial time an equivalent instance I_2 with primal graph G_2 and the same domain.*

Proof. For simplicity, we assume that both G_1 and G_2 are connected; the proof can be easily extended to the general case. If G_2 is connected, then we can assume that ψ is onto. For each constraint $c_i = \langle s_i, R_i\rangle$ of I_1, we construct a constraint of I_2 as follows. Let $s_i = (u,v)$. Since ψ is a minor mapping, there has to be at least one edge $e = u'v'$ in G_2 such that $u' \in \psi(u)$ and $v' \in \psi(v)$. Select such an edge and add the constraint $\langle (u',v'), R_i\rangle$ to the constructed instance I_2. Furthermore, for each edge xy of G_2, we add a constraint: if $\psi^{-1}(x) = \psi^{-1}(y)$, then the new constraint is $\langle (x,y), \{(t,t) \mid t \in D\}\rangle$; if $\psi^{-1}(x) \neq \psi^{-1}(y)$, then the new constraint is $\langle (x,y), D\times D\}\rangle$. Clearly, the primal graph of I_2 is G_2.

Assume that I_1 has a solution $f_1 : V_1 \to D$. Then $f_2(v) = f_1(\psi^{-1}(v))$ is a solution of I_2. On the other hand, if I_2 has a solution $f_2 : V_2 \to D$, then we claim that $f_2(x) = f_2(y)$ holds if $\psi^{-1}(x) = \psi^{-1}(y)$. This follows from the way we defined the constraints of I_2 and from the fact that $\psi(x)$ is connected. Therefore, we can define $f_1 : V_1 \to D$ as $f_1(v) = f_2(v')$, where v' is an arbitrary member of $\psi(v)$. To see that a constraint $c_i = \langle (u,v), R_i\rangle$ of I_1 is satisfied, observe that there is a constraint $\langle (u',v'), R_i\rangle$ in I_2 for some $u' \in \psi(u)$, $v' \in \psi(v)$. This means that $(f_1(u), f_1(v)) = (f_2(u'), f_2(v')) \in R_i$, hence the constraint is satisfied. $\square$

An instance with primal graph $G^{(q)}$ can be simulated by an instance with primal graph G if we set the domain to be the q-tuples of the original domain.

Lemma 17. *Given a binary CSP instance $I_1 = (V_1, D_1, C_1)$ with primal graph $G^{(q)}$ (where G has no isolated vertices), it is possible to construct (in time polynomial in the size of the* output*) an equivalent instance $I_2 = (V_2, D_2, C_2)$ with primal graph G and $|D_2| = |D_1|^q$.*

Proof. Let ψ be the blow-up mapping from G to $G^{(q)}$ and let $D_2 = D_1^q$, i.e., D_2 is the set of q-tuples of D_1. For every $v \in V_2$, there is a natural bijection between the elements of D_2 and the $|D|^q$ possible assignments $f : \psi(v) \to D$. For each edge v_1v_2 of G, we add a constraint $c_{v_1,v_2} = \langle (v_1,v_2), R_{v_1,v_2}\rangle$ to I_2 as follows. Let $(x_1,x_2) \in D_2 \times D_2$. For $i = 1,2$, let g_i be the assignment of $\psi(v_i)$ corresponding to $x_i \in D_2$. The two assignment together define an assignment $g : \psi(v_1) \cup \psi(v_2) \to D$ on the union of their domains. We define the relation R_{v_1,v_2} such that (x_1,x_2) is a member of R_{v_1,v_2} if and only if the corresponding assignment g is a solution of the induced instance $I_1[\psi(v_1) \cup \psi(v_2)]$.

Assume that I_1 has a solution $f_1 : V_1 \to D_1$. For every $v \in V_2$, let us define $f_2(v)$ to be the member of D_2 corresponding to the assignment f_1 restricted to $\psi(v)$. It is easy to see that f_2 is a solution of I_2: this follows from the trivial fact that for every edge v_1v_2 in G, assignment f_1 restricted to $\psi(v_1) \cup \psi(v_2)$ is a solution of $I_1[\psi(v_1) \cup \psi(v_2)]$.

Assume now that I_2 has a solution $f_2 : V_2 \to D_2$. For every $v \in V_2$, there is an assignment $f_v : \psi(v) \to D_1$ corresponding to $f_2(v)$. These assignments together define an assignment $f_1 : V_1 \to D_1$. We claim that f_1 is a solution of I_1. Let $c_{u,v} = \langle (u,v), R\rangle$ be an arbitrary constraint of I_1. Assume that $u \in \psi(u')$ and $v \in \psi(v')$. If $u' \neq v'$, then $u'v'$ is an edge of G, hence there is a corresponding constraint $c_{u',v'}$ in I_2. The way $c_{u',v'}$ is defined ensures that f_1 restricted to $\psi(u') \cup \psi(v')$ is a solution of $I_1[\psi(u') \cup \psi(v')]$. In particular, this means that $c_{u,v}$ is satisfied in f_1. If $u' = v'$, then there is an edge $u'w$ in G (since G has no isolated vertices), and the corresponding constraint $c_{u',w}$ ensures that f_1 satisfies $c_{u,v}$. $\square$

Now we are ready to prove the main result:

Proof (of Theorem 2). Assume that there is an algorithm with running time $f(G)\|I\|^{\mathrm{tw}(G)/(\log \mathrm{tw}(G)\cdot\iota(\mathrm{tw}(G)))}$, where ι is an unbounded function. We can assume that ι is nondecreasing. We present a reduction from 3SAT to CSP($\mathscr{G}$) such that this reduction, together with the assumed algorithm for CSP($\mathscr{G}$), is able to solve 3SAT in subexponential time. The crucial point of the reduction is how to select an appropriate G from $\mathscr{G}$. The higher the treewidth of G, the more we gain in the running time. However, G has to be sufficiently small such that some additional factors (such as the time spent on finding G) are not too large.

Since $\mathscr{G}$ is recursively enumerable and has unbounded treewidth, there is an algorithm A that, given an integer k, finds a graph $G \in \mathscr{G}$ with $\mathrm{tw}(G) \geq k$ in time $g(k)$, for some function g. Using algorithm A, it is not difficult to construct an algorithm B that, given an integer $m \geq m_0$ (where m_0 is a constant), constructs a graph G in time $O(m)$ such that

1. $f(G)|E(G)|^{\mathrm{tw}(G)/\iota(1)} = O(m)$, where f is the function in the statement of the theorem,
2. $f_1(G) \leq m$, where f_1 is the function in Lemma 4.
3. $f_2(G) = O(m)$, where f_2 is the function in Lemma 4, and
4. $\mathrm{tw}(G) = h(m)$ for some unbounded function h.

Algorithm B simulates algorithm A for $k = 1, 2, \ldots$ and after producing the graph for a given k, it checks whether it satisfies requirements 1–3 above. Algorithm B stops after

$O(m)$ steps and outputs the last graph that satisfies the requirements. Any graph G satisfies requirements 1–3 above if m is sufficiently large. Thus there is no bound on the treewidth of the graphs produced by Algorithm B, i.e., if $h(m)$ is defined to be the treewidth of the graph found with input m, then $h(m)$ is unbounded. If the constant m_0 is sufficiently large, then algorithm B can find in time $O(m)$ at least one graph satisfying the requirements above for any $m \geq m_0$.

We use the assumed algorithm for CSP($\mathscr{G}$) to solve m-clause 3SAT in time $2^{o(m)}$. Let ϕ be a 3SAT formula with m clauses; by Lemma 15, ϕ can be turned into a binary CSP instance I_1 with $O(m)$ constraints and domain size 3. Let H be the primal graph of I_1. Let us run algorithm B with input m to obtain a graph $G \in \mathscr{G}$ and let $k := \text{tw}(G) = h(m)$. For simplicity, we assume that G has no isolated vertices as they can be handled in a straightforward way. By Lemma 4, H is a minor of $G^{(q)}$ for $q = O(m \log k/k)$ and we can find a minor mapping ψ in time $f_2(G)m^{O(1)} = m^{O(1)}$ (since $f_2(G) = O(m)$ by Property 2 above). Therefore, by Lemma 16, I_1 can be turned into an instance I_2 with primal graph $G^{(q)}$, which, by Lemma 17, can be turned into an instance I_3 with primal graph G and domain size 3^q. Clearly, $\|I_3\| = O(|E(G)|3^{2q})$. The assumed algorithm can solve I_3 in time

$$\begin{aligned} f(G)\|I_3\|^{k/(\log k \cdot \iota(k))} \\ &= f(G)|E(G)|^{k/(\log k \cdot \iota(k))} \cdot 3^{2qk/(\log k \cdot \iota(k))} \\ \leq f(G)|E(G)|^{k/\iota(1)} \cdot 3^{O(m)/\iota(k)} = O(m) \cdot 3^{O(m)/\iota(k)} = 2^{o(m)}, \end{aligned}$$

contradicting ETH (the last equality follows from the fact that $\iota(k) = \iota(h(m))$ is unbounded as m tends to infinity). □

5 Complexity of homomorphism

The aim of this section is to extend Theorem 2 in the framework of the homomorphism problem for relational structures, which is the standard way of studying CSP in the theoretical literature. As we shall see, in this formulation the complexity of the problem depends on the treewidth of the core of the left-hand side. Furthermore, as in [12], we state the result only for bounded-arity relational structures.

Let us recall the standard definitions of the homomorphism problem (see [7, 12]). A *vocabulary* τ is a finite set of relation symbols of specified arities. The *arity* of τ is the maximum of the arities of all relational symbols it contains. A τ-structure $\mathbf{A}$ consists of a finite set A called the universe of $\mathbf{A}$ and for each relation symbol $R \in \tau$, say, of arity k, a k-ary relation $R^{\mathbf{A}} \subseteq A^k$. We say that a class $\mathscr{C}$ of structures is of *bounded arity* if there is a constant r such that the arity of the vocabulary of every structure in $\mathscr{C}$ is at most r. A *homomorphism* from a τ-structure $\mathbf{A}$ to a τ-structure $\mathbf{B}$ is a mapping $h : A \to B$ from the universe of $\mathbf{A}$ to the universe of $\mathbf{B}$ that preserves all relations, that is, for all $R \in \tau$, say, of arity k, and all tuples $(a_1, \ldots, a_k) \in R^{\mathbf{A}}$ it holds that $(h(a_1), \ldots, h(a_k)) \in R^{\mathbf{B}}$. Let $\|\mathbf{A}\|$ denote the length of the representation of $\mathbf{A}$. We assume that $\|\mathbf{A}\| = O(|\tau| + |A| + \sum_{R\in\tau} |R^A| \cdot \text{arity}(R))$ for a τ-structure $\mathbf{A}$ with universe A.

A *substructure* of a relational structure $\mathbf{A}$ is a relational structure $\mathbf{B}$ over the same vocabulary τ as $\mathbf{A}$ where $B \subseteq A$ and $R^{\mathbf{B}} \subseteq R^{\mathbf{A}}$ for all $R \in \tau$. If $\mathbf{B}$ is a substructure of $\mathbf{A}$, but $\mathbf{A} \neq \mathbf{B}$, then $\mathbf{B}$ is a *proper substructure* of $\mathbf{A}$.

The notion of treewidth can be defined for relational structures the following way. A *tree decomposition* of a τ-structure $\mathbf{A}$ is a pair (T, X), where $T = (I, F)$ is a tree, and $X = (X_i)_{i\in I}$ is a family of subsets of A such that for each $R \in \tau$, say, of arity k, and each $(a_1, \ldots, a_k) \in R^{\mathbf{A}}$ there is a node $i \in I$ such that $\{a_1, \ldots, a_k\} \subseteq X_i$, and for each $a \in A$ the set $\{i \in I \mid a \in X_i\}$ is connected in T. The *width* of the decomposition (T, X) is $\max\{|X_i| \mid i \in I\} - 1$, and the *treewidth* of $\mathbf{A}$, denoted by $\text{tw}(\mathbf{A})$, is the minimum of the widths of all tree decompositions of $\mathbf{A}$.

The *primal graph* of a structure $\mathbf{A}$ with vocabulary τ is a graph with vertex set A where two elements $a', a'' \in A$ are connected if and only if there is a relational symbol $R \in \tau$, say, of arity k, such that R has a tuple $(a_1, \ldots, a_k) \in R$ with $a', a'' \in \{a_1, \ldots, a_k\}$. It can be shown that the treewidth of the primal graph of $\mathbf{A}$ equals the treewidth of $\mathbf{A}$.

A *core* of a relational structure $\mathbf{A}$ is a substructure $\mathbf{A}'$ of $\mathbf{A}$ such that there is a homomorphism from $\mathbf{A}$ to $\mathbf{A}'$, but there is no homomorphism from $\mathbf{A}$ to a proper substructure of $\mathbf{A}'$. We say that a relational structure $\mathbf{A}$ is a *core* if it is its own core. It is well-known that the every relational structure $\mathbf{A}$ has a core and the cores of $\mathbf{A}$ are isomorphic with each other. Let us denote by $\text{ctw}(\mathbf{A})$ the treewidth of the core of $\mathbf{A}$.

Given a CSP instance $I = (V, D, C)$, one can construct in polynomial time two relational structures $\mathbf{A}$ and $\mathbf{B}$ with universe V and D, respectively, such that the solutions of I correspond to the homomorphisms from $\mathbf{A}$ to $\mathbf{B}$. Thus the homomorphism problem of relational structures generalize constraint satisfaction. Formally, in the homomorphism problem the input is a pair $(\mathbf{A}, \mathbf{B})$ of relational structures and the task is to decide whether there is a homomorphism form $\mathbf{A}$ (the *left-hand side structure*) to $\mathbf{B}$ (the *right-hand side structure*). If $\mathscr{A}$ and $\mathscr{B}$ are two classes of relational structures, then we denote by HOM($\mathscr{A}, \mathscr{B}$) the restriction of the homomorphism problem where $\mathbf{A} \in \mathscr{A}$ and $\mathbf{B} \in \mathscr{B}$. We denote by the symbol $-$ the class of all relational structures. Thus HOM($\mathscr{A}, -$) restricts the structure of the constraints, while HOM($-, \mathscr{B}$) restricts the constraint language.

If $\text{ctw}(\mathbf{A}) \leq k$, then the homomorphism problem $(\mathbf{A}, \mathbf{B})$ can be solved in time $n^{O(k)}$ [12, 4] (where n is the length of the input, which is $O(\|A\|+\|B\|)$). The main result of this section is that there is no class $\mathscr{A}$ of structures such that HOM$(\mathscr{A},-)$ can be solved significantly faster:

Theorem 18. *Let $\mathscr{A}$ be a recursively enumerable class of bounded-arity relational structures such that the treewidth of the core is unbounded. If HOM$(\mathscr{A},-)$ can be solved in time $f(\mathbf{A})\|\mathbf{B}\|^{o(\mathrm{ctw}(\mathbf{A})/\log \mathrm{ctw}(\mathbf{A}))}$, where f is an arbitrary computable function, then ETH fails.*

Proof. Let $\mathscr{A}$ be a class of relational structures of maximum arity $r_{\max}$. Let $\mathscr{G}$ be the class of graphs containing the primal graph of the core of every structure $\mathbf{A}\in\mathscr{A}$. Clearly, $\mathscr{G}$ has unbounded treewidth and it is not difficult to show that $\mathscr{G}$ is recursively enumerable. We use the assumed algorithm for HOM$(\mathscr{A},-)$ to construct an algorithm for CSP$(\mathscr{G})$ that contradicts Theorem 2.

Since $\mathscr{A}$ is recursively enumerable, there is an algorithm that, given a $G\in\mathscr{G}$, outputs a structure $\mathbf{A}_G\in\mathscr{A}$ such that G is the primal graph of the core of $\mathbf{A}_G$. Let $g(G)$ be the running time of this algorithm with input G; clearly, $\|\mathbf{A}_G\|\le g(G)$. Let $I=(V,D,C)$ be an instance of binary CSP with primal graph $G\in\mathscr{G}$. Let $\mathbf{A}_G\in\mathscr{A}$ be a structure whose core $\mathbf{A}_0$ has primal graph G. (From now on, we use V both for the set of variables of instance I and for the universe $\mathbf{A}_0$.) Let τ be the vocabulary of $\mathbf{A}_G$. We construct a τ-structure $\mathbf{B}$ as follows. The universe B of $\mathbf{B}$ is $V\times D$. Let $R\in\tau$ be a relation symbol and let $R^{\mathbf{A}_0}$ be the corresponding relation in $\mathbf{A}_0$. To construct the relation $R^{\mathbf{B}}$, let us enumerate the r-tuples of $R^{\mathbf{A}_0}$, and for each $(v_1,\dots,v_r)\in R^{\mathbf{A}_0}\subseteq V^r$, let us enumerate the solutions of $I[(v_1,\dots,v_r)]$. If $(v_1,\dots,v_r)\in R^{\mathbf{A}_0}$ and f is a solution of $I[(v_1,\dots,v_r)]$, then let us add the r-tuple $((v_1,f(v_1)),\dots,(v_r,f(v_r))$ to $R^{\mathbf{B}}$. This completes the description of the relation $R^{\mathbf{B}}$ and the structure $\mathbf{B}$. Observe that the size of $R^{\mathbf{B}}$ is at most $D^{r_{\max}}$ times the size of $R^{\mathbf{A}_0}$. Therefore, the size of $\mathbf{B}$ is $(\|\mathbf{A}_0\||D|)^{O(r_{\max})}$ and can be constructed in time polynomial in its size.

We show that $\mathbf{A}_0\to\mathbf{B}$ if and only if I has a solution. Since $\mathbf{A}_0$ is the core of $\mathbf{A}_G$, it follows that $\mathbf{A}_G\to\mathbf{B}$ if and only if $\mathbf{A}_0\to\mathbf{B}$. Therefore, the assumed algorithm for HOM$(\mathscr{A},-)$ can decide the solvability of I in time

$$\begin{aligned}
&g(G)+f(\mathbf{A}_G)\|\mathbf{B}\|^{o(\mathrm{ctw}(\mathbf{A}_G)/\log\mathrm{ctw}(\mathbf{A}_G))}\\
&=g(G)+f(\mathbf{A}_G)\|\mathbf{A}_0\|^{o(\mathrm{tw}(G)/\log\mathrm{tw}(G))}\cdot|D|^{o(\mathrm{tw}(G)/\log\mathrm{tw}(G))}\\
&\qquad\le f'(G)\|I\|^{o(\mathrm{tw}(G)/\log\mathrm{tw}(G))},
\end{aligned}$$

for an appropriate function $f'(G)$ (the last step follows from the fact that $f(\mathbf{A}_G)$ and $\|\mathbf{A}_0\|$ are functions of G, and that $|D|\le\|I\|$). By Theorem 2, this implies that ETH fails.

Assume first that I has a solution $f:V\to D$. We claim that $\phi(v)=(v,f(v))$ is a homomorphism from $\mathbf{A}_0$ to $\mathbf{B}$. Indeed, if $(v_1,\dots,v_r)\in R^{\mathbf{A}_0}$, then f restricted to $\{v_1,\dots,v_r\}$ is obviously a solution of $I[\{v_1,\dots,v_r\}]$, hence $((v_1,f(v_1)),\dots,(v_r,f(v_r)))\in R^{\mathbf{B}}$ by the definition of $R^{\mathbf{B}}$.

Assume now that ϕ is a homomorphism from $\mathbf{A}_0$ to $\mathbf{B}$. Let ψ be the projection $\psi(v,d)=v$ from $V\times D$ to V. Observe that ψ is a homomorphism from $\mathbf{B}$ to $\mathbf{A}_0$. Therefore, $\psi\circ\phi$ is a homomorphism from $\mathbf{A}_0$ to itself. Since $\mathbf{A}_0$ is core, $\psi\circ\phi$ is an isomorphism of $\mathbf{A}_0$. Thus we can assume that $\psi\circ\phi$ is identity: otherwise let us replace ϕ with $\phi\circ(\psi\circ\phi)^{-1}$. If $\psi\circ\phi$ is the identity, then for every $v\in V$, $\phi(v)=(v,f(v))$ for some $f(v)\in D$. We claim that this function $f:V\to D$ is a solution of I. Let $c_i=\langle(u,v),R_i\rangle$ be an arbitrary constraint of I. Since uv is an edge of the primal graph G, there is an $R\in\tau$ such that $R^{\mathbf{A}_0}$ has a tuple $(v_1,\dots,v_r)$ containing both u and v. Therefore, $(\phi(v_1),\dots,\phi(v_r))=((v_1,f(v_1)),\dots,(v_r,f(v_r)))\in R^{\mathbf{B}}$. By the definition of $R^{\mathbf{B}}$, this means that f restricted to $\{v_1,\dots,v_r\}$ is a solution of $I[\{v_1,\dots,v_r\}]$. In particular, this means that f satisfies c_i. $\square$

Acknowledgments

I'm grateful to Martin Grohe for the many useful discussions that were crucial for obtaining the results in the paper.

References

[1] A. A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proc. 43th Symp. Foundations of Computer Science*, pages 649–658. IEEE, November 2002.

[2] A. A. Bulatov. Tractable conservative constraint satisfaction problems. In *18th Annual IEEE Symposium on Logic in Computer Science (LICS'03)*, page 321, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[3] A. A. Bulatov, A. A. Krokhin, and P. Jeavons. The complexity of maximal constraint languages. In *Proceedings of the 33rd ACM Symposium on Theory of Computing*, pages 667–674, 2001.

[4] V. Dalmau, P. G. Kolaitis, and M. Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *CP '02: Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 310–326, London, UK, 2002. Springer-Verlag.

[5] R. Diestel. *Graph theory*, volume 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, third edition, 2005.

[6] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer-Verlag, New York, 1999.

[7] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1999.

[8] U. Feige, M. T. Hajiaghayi, and J. R. Lee. Improved approximation algorithms for minimum-weight vertex separators [extended abstract]. In *STOC'05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 563–572, New York, 2005. ACM.

[9] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2006.

[10] E. C. Freuder. Complexity of k-tree structured constraint satisfaction problems. In *Proc. of AAAI-90*, pages 4–9, Boston, MA, 1990.

[11] M. Grohe. The structure of tractable constraint satisfaction problems. In *MFCS 2006*, pages 58–72, 2006.

[12] M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1, 2007.

[13] M. Grohe and D. Marx. Constraint solving via fractional edge covers. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithms*, pages 289–298, New York, NY, USA, 2006. ACM Press.

[14] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. System Sci.*, 63(4):512–530, 2001. Special issue on FOCS 98 (Palo Alto, CA).

[15] P. Jeavons, D. A. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.

[16] T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999.

[17] B. Reed. Tree width and tangles: A new connectivity measure and some applications. In R. Bailey, editor, *Surveys in Combinatorics*, volume 241 of *LMS Lecture Note Series*, pages 87–162. Cambridge University Press, 1997.

[18] N. Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *J. Combin. Theory Ser. B*, 41(1):92–114, 1986.

[19] T. J. Schaefer. The complexity of satisfiability problems. In *Conference Record of the Tenth Annual ACM Symposium on Theory of Computing (San Diego, Calif., 1978)*, pages 216–226. ACM, New York, 1978.