The generic homomorphism problem, which asks whether an input graph $G$ admits a homomorphism into a fixed target graph $H$, has been widely studied in the literature. In this article, we provide a fine-grained complexity classification of the running time of the homomorphism problem with respect to the clique-width of $G$ (denoted $cw$) for virtually all choices of $H$ under the Strong Exponential Time Hypothesis. In particular, we identify a property of $H$ called the signature number $s(H)$ and show that for each $H$, the homomorphism problem can be solved in time $O^\ast(s(H)^{cw})$. Crucially, we then show that this algorithm can be used to obtain essentially tight upper bounds. Specifically, we provide a reduction that yields matching lower bounds for each $H$ that is either a projective core or a graph admitting a factorization with additional properties—allowing us to cover all possible target graphs under long-standing conjectures.
1 INTRODUCTION

A homomorphism from a graph $G$ to a graph $H$ is an edge-preserving mapping from the vertices of $G$ to the vertices of $H$. Homomorphisms are fundamental constructs which have been studied from a wide variety of perspectives [3, 4, 15].

Our focus here will be on the class of problems which ask whether an input $n$-vertex graph $G$ admits a homomorphism to a fixed target graph $H$. This "meta-problem"—which we simply call Hom$(H)$—captures, among others, the classical c-COLORING problems when $H$ is set to the complete graph on $c$ vertices. Famously, Hell and Nešetřil [18] proved that Hom$(H)$ is polynomial-time solvable if $H$ is bipartite or has a loop, and NP-complete otherwise.

While the aforementioned result provides a basic classification of the complexity of Hom$(H)$, it does not say much in terms of how quickly one can actually solve these problems. Indeed, the usual assumption is that P≠NP is not sufficient to obtain tight bounds for the running times of algorithms. While upper bounds can be straightforwardly obtained by designing a suitable algorithm, the corresponding lower bounds usually rely on the Exponential Time Hypothesis (ETH) or the Strong Exponential Time Hypothesis (SETH), which allows for even tighter bounds [21, 22, 26]. It is not difficult to design a brute-force algorithm for the homomorphism problem that runs in time $O^*(|V(H)|^n)$ for every choice of $H$, and thanks to the breakthrough result of Cygan et al. we now know that this running time is essentially tight under the ETH [7] as long as one considers only the dependency on $n$ and $|V(H)|$.

Still, it is often possible to circumvent this lower bound and obtain significantly better runtime guarantees. One approach to do so is to consider restrictions on the class of targets: if $H$ is a complete graph then Hom$(H)$ can be solved in time $O^*(2^n)$, and there are also several algorithms that achieve running times of the form $O^*(a(H)^n)$ where $a(H)$ is some structural parameter of $H$ [14, 33, 36]. The other is to exploit the properties of the input graph $G$, which are commonly captured by a suitably defined structural parameter. The most commonly used graph parameter in this respect is treewidth [32], which informally measures how "tree-like" a graph is.

When considering treewidth, it is once again not difficult to obtain an algorithm that runs in time $O^*(|V(H)|^{tw})$, where $tw$ is the treewidth of $G$; as before, it was much more difficult to show that this is essentially optimal. The first SETH-based tight lower bound in this setting was actually shown for special cases of the related problem of LHom$(H)$, where each vertex in the graph $G$ comes with a list of admissible targets for the homomorphism [11]; this was later lifted to a full classification [28]. A nearly complete SETH-based lower bound result for Hom$(H)$ itself was only obtained recently by Okrasa and Rzążewski [29]; in particular, the result covers all targets which are so-called projective cores. It is known that almost all graphs are projective cores [19, 29, 35], and it is worth noting that the authors showed that their results can be lifted to all targets under long-standing conjectures on the properties of projective cores [24, 25].

While treewidth is the most prominent structural graph parameter, it is not the most general one that can be used to efficiently solve Hom$(H)$. Indeed, standard dynamic programming techniques can be used to obtain a $O^*((2^{|V(H)|})^{cw})$ time algorithm for the problem, where $cw$ stands for clique-width [5]: a well-studied graph parameter that is bounded not only on all graph classes of bounded treewidth but also on well-structured dense classes such as complete graphs. But is this basic algorithm generally optimal (mirroring the situation for treewidth [29]), or can one obtain better runtime dependencies on clique-width?

\footnote{There is a hierarchy of graph parameters (see, e.g., Figure 1 in Ref. [2]), where parameter $A$ is more general than parameter $B$ if there are graph classes of bounded $A$ and unbounded $B$ but the opposite is not true.}
2 CONTRIBUTION

Our aim is to obtain a detailed understanding of the fine-grained complexity of $\text{Hom}(H)$ in terms of the clique-width of $G$ and the fixed target $H$. As a starting point for our investigation, we note that Lampis used the SETH to obtain tight bounds for $c$-$\text{COLORING}$ with respect to clique-width [23]. Interestingly, already for this special case, the upper and lower bounds differ from those of the aforementioned simple dynamic programming algorithm: if $H$ is a complete graph, then $\text{Hom}(H)$ can be solved in time $O^\ast((2^{|V(H)|} - 2)^{cw})$ [23] and this is tight under the SETH. However, as noted by Piecyk and Rzążewski [31], it was not at all obvious how these bounds can be lifted to general choices of $H$.

To achieve our goals, we need to improve upon the basic dynamic programming idea to identify a “hopefully correct” base of the exponent for every choice of $H$. Toward our first result, we identify a structural property of $H$ called the signature number (denoted $B(H)$) which, intuitively, captures the number of non-trivial neighborhood classes of vertex subsets in $H$ (the signature set). We then obtain a non-trivial dynamic programming algorithm that solves $\text{Hom}(H)$ in time where the base of the exponent is precisely the signature number. We note that $B(H)$ is $2^{|V(H)|} - 2$ for complete graphs $H$, and so this result also provides a succinct and broader explanation for the running time of Lampis’ algorithm [23].

**Theorem 2.1.** Let $H$ be a fixed graph. $\text{Hom}(H)$ can be solved in time $O^\ast(s(H)^t)$ for each input graph $G$, assuming a clique-width expression of $G$ of width $t$ is provided as part of the input.

With this upper bound, we proceed to the main technical contribution of this article: establishing a corresponding lower bound under the SETH. The main difficulty here is that we need a reduction that is delicate on one hand, since it needs to preserve the clique-width, but is on the other hand also flexible enough to work for many different choices of $H$; moreover, the reduction has to rely on the signature numbers of these graphs in some way.

To provide an intuitive description of the reduction, let us focus for now on the case where $H$ is a projective core. On a high level, the main building block is an $S$-gadget which, given an arbitrary set $S$ of pairs of vertices in $H$ and two vertices $p$ and $q$ of the input graph $G$, ensures that every homomorphism $f$ satisfies $(f(p), f(q)) \in S$. After providing a generic construction for such $S$-gadgets which is clique-width preserving and works for every valid choice of $H$, we use these to obtain implication gadgets and or gadgets which restrict how a solution homomorphism can behave on a selected set of vertices in $G$. The formalization of these gadgets is the main technical hurdle toward the desired result; once that is done, we can lift the idea used in the earlier reduction of Lampis [23] that established clique-width lower bounds for $c$-$\text{COLORING}$ by reducing from Constraint Satisfaction (CSP) to $\text{Hom}(H)$. One crucial distinction in our reduction is that we use elements of the signature set (as opposed to color sets) to represent domain values in the CSP instance.

To lift these considerations to cases where $H$ is not a projective core, we unfortunately need to add an extra layer of complexity. Similarly as in the previous treewidth-based lower bound for $\text{Hom}(H)$ [29], one can base this step on conjectures of Larose and Tardif [24, 25] that classify all remaining targets as certain graph products with special properties (notably, all of the factors must be “truly projective”). The approach used for treewidth [29] was then to essentially repeat all steps of the proof for projective cores, with the added difficulty that one uses the properties of products instead of dealing directly with projective cores.

While this approach could be used here as well, instead we unify the two cases ($H$ being a projective core and $H$ being a product) by defining the notion of $W$-projectivity for some factor $W$ of $H$. In particular, if $H$ is a projective core then it itself is $H$-projective, while if $H$ is a product with a truly projective factor $H_f$ then it is $H_f$-projective. As our main result, we obtain an SETH-based
lower bound which essentially shows that for each \( W \)-projective graph \( H, s(W) \) is the optimal base of the clique-width exponent for solving \( \text{Hom}(H) \):

**Theorem 2.2.** Let \( H \) be a fixed non-trivial core with prime factorization \( H_1 \times \ldots \times H_m \). If \( H \) is \( H_i \)-projective for some \( i \in [m] \) then there is no algorithm solving \( \text{Hom}(H) \) in time \( O^\ast((s(H_i) - \epsilon)^{cw(G)}) \) for any \( \epsilon > 0 \), unless the SETH fails.

By also deliberately considering prime factorizations in the algorithm that we provide for Theorem 2.1, we can obtain an upper bound on the complexity of \( \text{Hom}(H) \) that matches the lower bound from Theorem 2.2. For a discussion explicitly relating these complexity bounds in the context of the aforementioned conjectures of Larose and Tardif, we refer to Section 7.

3 PRELIMINARIES

We use standard terminology for graph theory [8]. Let \([i]\) denote the set \(\{1, \ldots, i\}\). For a mapping \( f : A \rightarrow B \) and \( A' \subseteq A \), let \( f|_{A'} \) denote the restriction of \( f \) to \( A' \). We use the \( O^\ast(\cdot) \) notation to suppress factors polynomial in the input size.

3.1 Homomorphisms and Cores

For two graphs \( G \) and \( H \), a homomorphism from \( G \) to \( H \) is a mapping \( h : V(G) \rightarrow V(H) \), such that for every \( uv \in E(G) \) we have \( h(u)h(v) \in E(H) \). If there exists a homomorphism from \( G \) to \( H \), we denote this fact by \( G \rightarrow H \), and if \( h \) is a homomorphism from \( G \) to \( H \), we denote that by \( h : G \rightarrow H \).

If there is no homomorphism from \( G \) to \( H \), we write \( G \not\rightarrow H \). If \( G \rightarrow H \) and \( H \rightarrow G \), we say that \( G \) and \( H \) are homomorphically equivalent. In particular, since the composition of homomorphisms is a homomorphism, if \( G \) and \( H \) are homomorphically equivalent, then for every graph \( F \) we have that \( F \rightarrow G \) if and only if \( F \rightarrow H \). It is straightforward to verify that homomorphic equivalence is an equivalence relation on the class of all graphs.

If \( G \) and \( H \) are graphs such that \( G \rightarrow H \) or \( H \rightarrow G \), we say that \( G \) and \( H \) are comparable. Otherwise, \( G \) and \( H \) are incomparable.

We note that if \( H \) is a clique on \( c \) vertices, then homomorphisms form \( G \) to \( H \) are precisely proper vertex \( c \)-colorings of \( G \).

An automorphism of \( H \) is a homomorphism from \( H \) to \( H \) that is a bijection. We say that a graph \( H \) is a core if every homomorphism \( h : H \rightarrow H \) is an automorphism. Equivalently, \( H \) is a core if for every proper induced subgraph \( H' \) of \( H \) it holds that \( H \not\rightarrow H' \). We say that a core \( H' \) is a core of \( H \) if \( H' \) is an induced subgraph of \( H \) and \( H \rightarrow H' \). Clearly, each core graph is a core of itself. Each graph has a unique (up to isomorphism) core, and the core of \( H \) can be equivalently defined as the smallest (with respect to the number of vertices) graph that is homomorphically equivalent with \( H \) [19].

A graph \( H \) is ramified if \( N(u) \nsubseteq N(v) \) for every two distinct vertices \( u, v \) of \( H \). Observe that each core is ramified; otherwise one could define \( f : H \rightarrow H \) that is an identity on all vertices of \( H \) but \( u \) and set \( f(u) = v \). This would be a homomorphism to a proper subgraph of \( H \), contradicting the fact that \( H \) is a core.

We say that a graph \( H \) is trivial if its core has at most two vertices.

**Observation 1** ([18]). A graph \( H \) is trivial if and only if it is either bipartite or contains a vertex with a loop.

**Proof.** It is straightforward to observe that there exist three trivial cores: \( K_1, K_2, \) and \( K_1' \), where by \( K_1' \) we denote the graph that consists of one vertex with a loop.

If \( H \) contains a vertex \( a \) with a loop, then \( K_1' \) is the core of \( H \), as mapping every vertex of \( H \) to \( a \) yields a homomorphism. If \( H \) is bipartite, then the core of \( H \) is either \( K_1 \) (if \( H \) has no edges) or \( K_2 \).
(since mapping the vertices of one bipartition class to one vertex of $K_2$, and another bipartition class to the other is a homomorphism).

For the other direction, assume that $H$ is a non-bipartite loopless graph. Since it is loopless, $K_1^*$ cannot be its core. Clearly, $H$ has at least one edge, and therefore $H \notightarrow K_1$. Moreover, $H$ contains an odd cycle $C_{2k+1}$ as a subgraph, hence, $C_{2k+1} \rightarrow H$. If now $H \rightarrow K_2$, composition of these homomorphism gives that $C_{2k+1} \rightarrow K_2$, which is equivalent to stating that $C_{2k+1}$ is 2-colorable, a contradiction. □

Observe that trivial cores $H$ correspond precisely to the polynomial cases of the $\text{Hom}(H)$ problem. Since our aim is to focus on the NP-hard cases of the problem, from here onward we assume that the target graph is non-trivial.

### 3.2 Signature Sets

For a vertex $v$ of a graph $H$, let $N_H(v)$ denote the set of neighbors of $v$ in $H$. If the graph is clear from the context, we omit the subscript $H$ and write $N(v)$.

For a non-empty set $T \subseteq V(H)$ we say that $S(T) = \bigcap_{t \in T} N(t)$ is the signature set of $T$, i.e., the signature set of $T$ is the intersection of the neighborhood sets of all vertices in $T$. We say that a non-empty set $Q \subseteq V(H)$ is a signature set, if there exists $T$ such that $Q = S(T)$. We denote by $\mathcal{S}(H)$ the set of all signature sets of $H$, and we note that $V(H) \notin \mathcal{S}(H)$ and $\emptyset \notin \mathcal{S}(H)$.

We observe some basic properties of signature sets.

**Observation 2.** Let $T \subseteq V(H)$ be non-empty.

1. If $a \in T$, $b \in S(T)$, then $ab \in E(H)$.
2. For every $a \in T$ we have $S(T) \subseteq N_H(a)$.
3. If $T' \subseteq V(H)$ is non-empty, then $S(T \cup T') = S(T) \cap S(T')$. In particular, if $T > T' \subseteq V(H)$, then $S(T) \supseteq S(T')$.

**Proof.** The first two statements follow directly from the definition of a signature set. For the third statement, observe that $S(T \cup T') = \bigcap_{t \in T \cup T'} N(t) = \bigcap_{t \in T} N(t) \cap \bigcap_{t' \in T'} N(t)' = S(T) \cap S(T')$. □

We also note that the operation of taking a signature set is reversible on $\mathcal{S}(H)$:

**Observation 3.** If $T > T$ is a non-empty proper subset of $V(H)$, then $S(S(T)) \supseteq T$. Moreover, if $T > T \in \mathcal{S}(H)$, then $S(S(T)) = T$.

**Proof.** By the definition of a signature set, $T \times S(T) \subseteq E(H)$, so $T \subseteq S(S(T))$. For the converse direction observe that if $T \in \mathcal{S}(H)$, there exists a non-empty subset $A$ of $V(H)$ such that $T = S(A)$. Pick any $x \in S(S(T))$, then $E(H) \supseteq \{x\} \times S(T) = \{x\} \times S(S(A)) \supseteq \{x\} \times A$. Hence by definition $x \in S(S(A)) = T$. □

Let the signature number of $H$, denoted $s(H)$, be defined as $|\mathcal{S}(H)|$. As mentioned in the introduction section, the signature number plays a crucial role in our upper and lower bounds.

Observe that, if $H$ is a target and hence non-trivial, for every non-empty $T \subseteq V(H)$ we have that $S(T) \cap T = \emptyset$. From that it is easy to see that $V(H)$ never belongs to $\mathcal{S}(H)$. Since, by definition, $\emptyset \notin \mathcal{S}(H)$, we get the following bounds for $s(H)$.

**Observation 4.** Let $H$ be a graph with no loops. Then $s(H) \leq 2^{|V(H)|} - 2$.

Notice that since $2^{|V(H)|} - 2$ is the number of all proper non-empty subsets of $V(H)$, the equality in Observation 4 holds if and only if $H$ is a clique.

We note that if $H$ is a core graph, we can also bound the minimum cardinality of $\mathcal{S}(H)$.
LEMMA 3.1. Let $H$ be a core graph, $H \neq K_1$. Then $s(H) \geq |V(H)|$.

PROOF. Observe that if $H$ is a core distinct from $K_1$, then it does not contain isolated vertices. Therefore, for each $v \in V(H)$ we have $N(v) \in S(H)$. At the same time, since $H$ is a core, it is ramified. In particular, for every distinct $v, w \in V(H)$ we have $N(v) \neq N(w)$. Hence different vertices give rise to different signature sets. □

Last, we note that if $H$ is disconnected, and $H_1, \ldots, H_m$ are the connected components of $H$, then $S(H)$ is precisely the disjoint union of $S(H_1), \ldots, S(H_m)$. Indeed, if $S \in S(H_i)$ for some $i \in [m]$, then $S \in S(H)$. Moreover, observe that for every set $T$ that is not contained in $V(H_i)$ for some $i \in [m]$, we have that $S(T) = 0$, hence $S(T)$ is not included in $S(H)$. Hence we obtain the following:

Observation 5. Let $H$ be a graph, and let $H_1, \ldots, H_m$ be the connected components of $H$. Then

$$S(H) = S(H_1) \cup \cdots \cup S(H_m),$$

and for each element $S$ of $|S(H)|$ there exists a unique $i \in [m]$ such that $S \in S(H_i)$.

3.2.1 A Dual Perspective. Informally, for $T \subseteq V(H)$, the signature set $S(T)$ will be used to capture viable candidate vertices in the target graph to which shared neighbors of vertices $X \subseteq V(G)$ can be mapped by a homomorphism $h : G \to H$, assuming that $h(X) = T$. Later, in the proof of our algorithmic result, this assumption will be based on the fact that we are dynamically extending homomorphisms, and in the proof of our lower bound our first step will be to reduce the setting of extending homomorphisms to finding homomorphisms from scratch.

While not necessary to obtain our algorithmic and lower bounds for $\text{Hom}(H)$ parameterized by clique-width, we can also take an alternative perspective: Instead of restricting the possible images of the shared neighbors of vertices, we can also restrict the possible images of vertices based on the images of vertices occurring in their shared neighborhood. Of course, these two views are equivalent to each other. Let us briefly setup the latter one and argue equivalence formally.

If $S \in S(H)$, we call $T$ such that $S(T) = S$ a witness of $S$. Clearly, we can have distinct $T_1, T_2$ such that $S(T_1) = S(T_2)$, however, notice that in such a case there exists $T = T_1 \cup T_2$ such that $S(T) = S(T_1) = S(T_2)$, see Figure 1 for an illustration.

Hence, the union of any pair of witnesses is also a witness, which means that there exists a unique maximal (with respect to inclusion) witness of $S$, and we denote it by $M(S)$. In fact, it is not difficult to see that $M(S) = \{ v \in V(H) \mid S \subseteq N_H(v) \}$; for $S(M(S)) = S$ to hold, it is clearly necessary that $S \subseteq N_H(v)$ for all $v \in M(S)$. On the other hand, as $M(S)$ is maximal all $v$ for which this is true are contained in $M(S)$.

![Fig. 1. Example illustrating the relationship between witnesses and signatures. Intersection of the neighborhoods of two blue (left) or two yellow (center) vertices is the set of both red vertices. Here, the maximal witness of the two red vertices (same as their signature set) is the union of all blue and yellow vertices (right).](image-url)

ACM Transactions on Algorithms, Vol. 20, No. 3, Article 19. Publication date: May 2024.
In this way signature sets and their maximal witnesses are in one-to-one correspondence. In fact, the signature number could equivalently be defined as the “maximal witness number” and signature sets could be replaced by maximal witnesses in all our proofs:

**Lemma 3.2.** Let \( M(H) = \{ M(S) : S \in S(H) \} \), then \( S(H) = M(H) \).

**Proof.** Let \( T \) be a fixed non-empty subset of \( V(H) \) such that \( S(T) \neq \emptyset \). Observe that the definition and the maximality of \( M(S(T)) \) implies that \( S(S(T)) = \bigcap_{t \in S(T)} N(t) = M(S(T)) \). Since \( T \) and \( S(T) \) are non-empty, we get that \( S(H) \supseteq M(H) \). For the other direction observe that \( M(M(S(T))) = S(T) \). Assume the contrary, then there exists a vertex \( v \notin S(T) \) such that \( N(v) \supseteq M(S(T)) \). However, \( T \subseteq M(S(T)) \subseteq N(v) \) meaning that \( v \in S(T) \), a contradiction. Thus, \( S(H) = M(H) \). \qed

### 3.3 Clique-Width and Clique-Width Expressions

For a positive integer \( k \), we let a \( k \)-graph be a graph whose vertices are labeled by \([k]\). For convenience, we consider a graph to be a \( k \)-graph with all vertices labeled by 1. Two \( k \)-graphs are isomorphic if they are isomorphic with the labels removed. We call the \( k \)-graph consisting of exactly one vertex \( v \) (say, labeled by \( i \)) an initial \( k \)-graph and denote it by \( i(v) \).

The *clique-width* of a graph \( G \) is the smallest integer \( k \) such that \( G \) can be constructed from initial \( k \)-graphs by means of iterative application of the following three operations:

1. **Disjoint union** (denoted by \( \circ \)).
2. **Relabeling:** changing all labels \( i \) to \( j \) (denoted by \( \rho_{i \rightarrow j} \)).
3. **Edge insertion:** adding an edge from each vertex labeled by \( i \) to each vertex labeled by \( j \) (\( i \neq j \); denoted by \( \eta_{i,j} \)).

Without loss of generality, we only need to consider graph operations which do not keep the graph invariant (as operations which keep the graph invariant can be skipped); in particular, this implies that

- for each relabeling operation \( \rho_{i \rightarrow j} \), there is at least one vertex with label \( i \) and
- for each edge insertion operation \( \eta_{i,j} \), there is at least one vertex with label \( i \) and at least one vertex with label \( j \).

A construction of a graph \( G \) using the above operations (1)–(3) can be represented by a first-order term \( \sigma \) composed from variables \( i(v) \), \( v \in V(G) \), via unary function symbols \( \rho_{i \rightarrow j} \) and \( \eta_{i,j} \) (where \( i \neq j \) and \( i, j \in [k] \)) and binary function symbol \( \circ \). Such a \( \sigma \) is called a \( k \)-expression (or shortly expression) defining \( G \). The smallest \( k \) such that \( \sigma \) is a \( k \)-expression is called the **width** of \( \sigma \). Conversely, we call the \( k \)-graph that arises from a \( k \)-expression its **evaluation**.

An expression \( \sigma \) defining \( G \) can be identified with a rooted binary tree, that we call its **expression tree**. The root of the tree corresponds to a \( k \)-graph isomorphic with \( G \), and the tree itself consists of four types of nodes:

1. Each non-root leaf node corresponds to a unique vertex \( v \in V(G) \) and is labeled with \( i(v) \).
2. Each node labeled \( \circ \) has precisely two children and corresponds to the disjoint union of graphs associated with them.
3. Each node labeled \( \rho_{i \rightarrow j} \) has a single child, and corresponds to the graph obtained by applying the relabeling operation \( \rho_{i \rightarrow j} \) to the graph associated with its child.
4. Each node labeled \( \eta_{i,j} \) has a single child and corresponds to the graph obtained by applying the edge insertion operation \( \eta_{i,j} \) to the graph associated with its child.

Given a \( k \)-expression \( \sigma \) with an expression tree \( T \) and given a node \( x \in V(T) \) labeled by \( \circ \), we say that the subexpression \( \tau \) of \( \sigma \) (with outermost operation \( \circ \)) is the expression corresponding to
the expression tree $T'$ that is the unique maximal subtree of $T$ rooted at $x$; in this case we say $\tau \subseteq \sigma$ and identify $x$ with $\tau$. If $\tau$ is not the root of $\sigma$, we say that $\tau$ is a proper subexpression of $\sigma$. Let us denote the evaluation of $\tau \subseteq \sigma$ by $G_\tau$. Equivalently, $G_\tau$ is the graph corresponding to the node $\tau$ in $T$. By $V'_\tau \subseteq V(G_\tau)$ we denote the vertex set that has label $i$ in $G_\tau$.

Many graph classes are known to have constant clique-width: examples include all graph classes of constant treewidth and co-graphs [6]. Moreover, a fixed-parameter algorithm is known to compute a $k$-expression of the input where $k$ is bounded in $f(\text{cw})$ [30].

4 ALGORITHM

As our first contribution, we obtain an algorithm that plays a crucial role for upper-bounding the fine-grained complexity of $\text{Hom}(H)$.

**Theorem 2.1.** Let $H$ be a fixed graph. $\text{Hom}(H)$ can be solved in time $O'((s(H))^f)$ for each input graph $G$, assuming a clique-width expression of $G$ of width $t$ is provided as part of the input.

**Proof.** Let $\sigma$ be the given $k$-expression of $G$, and let $T$ be its expression tree. If $G$ is disconnected, we process every connected component of $G$ independently. From now, let us assume that $G$ is connected and $|V(G)| > 1$. Our algorithm will proceed dynamically in a leaf-to-root fashion along $T$.

For a node $\tau \subseteq \sigma$ of $T$, we say that $i$ is a live label in $\tau$ if there is an edge of $G$ which is incident to $V'_\tau$ and does not appear in $G_\tau$. Observe that in this case, by the construction of the clique-width expression, for every $v \in V'_\tau$ there exists such an edge incident to $v$. Denote the set of live labels in $\tau$ by $L_\tau$. Since $G$ is connected, $L_\tau \neq \emptyset$ for any proper subexpression $\tau$ of $\sigma$.

For each subexpression $\tau \subseteq \sigma$, we will compute a set $P_\tau$ consisting of functions $p : L_\tau \to S(H)$ where $p \in P_\tau$ if and only if

$$\text{there exists } h_p : G_\tau \to H \text{ s.t. for every } i \in L_\tau \text{ we have } p(i) \subseteq S(h_p(V'_i)). \quad (1)$$

We say that $p \in P_\tau$ describes the homomorphism $h_p$ in $\tau$ or, equivalently, that $h_p$ witnesses $p$ in $\tau$.

Intuitively, we use $p(i)$ to preemptively store the images of the neighbors of $V'_i$ in the final graph $G$—that is why $p(i)$ is not necessarily the exact signature, but could be any signature that occurs as a subset.\footnote{Storing in $P_\tau$ only those functions $p : L_\tau \to S(H)$ where $p(i) = S(h_p(V'_i))$, $i \in L_\tau$, would be sufficient to obtain a conceptually simpler fixed-parameter algorithm parameterized by clique-width, but in that case it is not obvious how to achieve a linear dependency on clique-width in the exponent.}

First, we describe how to compute the sets, and the intuitive reasoning underlying these computations. Readers are welcome to compare the following description with its pseudocode available in Algorithm 1. We distinguish the cases based on the outermost operation of $\tau$.

- $\tau = i(v)$ for some $i \in [k]$ (cf. Algorithm 1, Lines 2–3).
- In this case $L_\tau = \{i\}$, since we assume that $G$ is connected and $|V(G)| > 1$. We add to $P_\tau$ all functions $p : \{i\} \to S(H)$.

\[ \tau = \rho_{i \to j}(\tau') \text{ for a unique child } \tau' \text{ of } \tau \text{ (cf. Algorithm 1, Lines 4–16)}. \]

If $i \notin L_{\tau'}$, we set $P_\tau = P_{\tau'}$. Otherwise, if $i \in L_{\tau'}$ and $j \notin L_{\tau'}$, every $p \in P_\tau$ arises from some $p' \in P_{\tau'}$ by replacing $i$ by $j$ in the domain:

\[ P_\tau = \{(p' \setminus (i, S)) \cup (j, S) \mid S = p'(i) \land p' \in P_{\tau'}\}. \]

Finally, if $i \in L_{\tau'}$ and $j \in L_{\tau'}$, then $L_\tau = L_{\tau'} \setminus \{i\}$ and $P_\tau = \{p'|_{L_{\tau'}} \mid p' \in P_{\tau'} \land p'(i) = p'(j)\}$.

Intuitively, we simply rename label $i$ to $j$, however, since after this step the vertices labeled $i$ and $j$ in $\tau'$ are indistinguishable, we first ensure that the image of their potential neighbors is the same.

\[ \tau = \tau_1 \oplus \tau_2 \text{ for children } \tau_1 \text{ and } \tau_2 \text{ of } \tau \text{ (cf. Algorithm 1, Lines 17–25)}. \]
Algorithm 1: ComputeRecordSet(τ)

Fixed: Target graph H and expression tree T
Input: A node τ of T
Result: \( P_τ = \{ p : L_τ \rightarrow S(H) \mid \exists h_p : G_τ \rightarrow H \text{ s.t. } \forall i \in L_τ \ p(i) \subseteq S(h_p(V^i_τ))\} \)

1. \( P_τ \leftarrow \emptyset \)
2. if \( τ = i(v) \) for some \( i \in [k] \) then
3. \( P_τ \leftarrow \{ p : \{ i \} \rightarrow S(H)\} \)
4. else if \( τ = \rho_{i→j}(\tau') \) for a unique child \( \tau' \) of \( τ \) then
5. \( P_{\tau'} \leftarrow \text{ComputeRecordSet}(\tau') \)
6. if \( i \not\in L_{\tau'} \) then
7. \( P_τ \leftarrow P_{\tau'} \)
8. else if \( i \in L_{\tau'} \) and \( j \not\in L_{\tau'} \) then
9. for \( p' \in P_{\tau'} \) do
10. \( S \leftarrow p'(i) \)
11. \( p \leftarrow (p' \setminus (i, S)) \cup (j, S) \)
12. \( P_τ \leftarrow P_τ \cup \{ p \} \)
13. else if \( i \in L_{\tau'} \) and \( j \in L_{\tau'} \) then
14. for \( p' \in P_{\tau'} \) do
15. if \( p'(i) = p'(j) \) then
16. \( P_τ \leftarrow P_τ \cup \{ p'|_{L_{\tau'}} \} \)
17. else if \( τ = τ_1 \oplus τ_2 \) for children \( τ_1 \) and \( τ_2 \) of \( τ \) then
18. \( P_{τ_1} \leftarrow \text{ComputeRecordSet}(τ_1) \)
19. \( P_{τ_2} \leftarrow \text{ComputeRecordSet}(τ_2) \)
20. \( P_τ \leftarrow \{ p : L_τ \rightarrow S(H)\} \)
21. for \( p \in P_τ \) do
22. \( p_1 \leftarrow p|_{L_{τ_1}} \)
23. \( p_2 \leftarrow p|_{L_{τ_2}} \)
24. if \( p_1 \not\in P_{τ_1} \) or \( p_2 \not\in P_{τ_2} \) then
25. \( P_τ \leftarrow P_τ \setminus \{ p \} \)
26. else if \( τ = η_{i,j}(\tau') \) for a unique child \( \tau' \) of \( τ \) then
27. \( P_{\tau'} \leftarrow \text{ComputeRecordSet}(\tau') \)
28. for \( p' \in P_{\tau'} \) do
29. for \( S_i, S_j \in S(H) \) do
30. if \( p'(i) \supseteq S(p'(j)) \) and \( (S_i \subseteq p'(i) \text{ or } i \not\in L_τ) \) and \( (S_j \subseteq p'(j) \text{ or } j \not\in L_τ) \) then
31. \( p \leftarrow p'|_{L_τ \setminus \{i,j\}} \)
32. if \( i \in L_τ \) then
33. \( p \leftarrow p \cup (i, S_i) \)
34. if \( j \in L_τ \) then
35. \( p \leftarrow p \cup (j, S_j) \)
36. \( P_τ \leftarrow P_τ \cup \{ p \} \)
In this case $L_\tau = L_{\tau_1} \cup L_{\tau_2}$ and
\[
P_\tau = \{ p = p_1 \cup p_2 \mid p_1 \in P_{\tau_1} \land p_2 \in P_{\tau_2} \land (\forall \ell \in L_{\tau_1} \cap L_{\tau_2} : p_1(\ell) = p_2(\ell)) \}.
\]

Intuitively, we construct a homomorphism on the disjoint union of two subgraphs by “gluing together” the homomorphisms on the subgraphs. If the subgraphs share any live labels, after this step they will all be treated equally. For this reason we require the images of the neighbors of such labels to be the same in both subgraphs.

$\tau = \eta_{i,j}(\tau')$ and $P_{\tau'}$ has already been computed (cf. Algorithm 1, Lines 26–36).

In this case we set $P_\tau$ equal to
\[
\{ p : L_\tau \to S(H) \mid \exists p' \in P_{\tau'} : p'(i) \supseteq S(p')(j) \land p|_{L_\tau \setminus \{i,j\}} = p'|_{L_\tau \setminus \{i,j\}} \land p(i) \subseteq p'(i) \land p(j) \subseteq p'(j) \}.
\]

where we interpret $p(i) = \emptyset$ and $p(j) = \emptyset$ if $i \notin L_\tau$ and $j \notin L_\tau$, respectively.

Intuitively, we can add the edges between two live labels if and only if there are edges between their images in $H$. Our restriction on $p'$ is an expression of this condition in terms of images of neighbors and their signatures. This concludes the description of the algorithm.

Observe that to prove correctness of our procedure it is enough to show that a mapping $p : L_\sigma \to S(H)$ belongs to $P_\sigma$ if and only if there exists $h : G \to H$ such that for every $i \in L_\sigma$ we have $p(i) \subseteq S(h(V_i))$. As $L_\sigma = \emptyset$, this is equivalent to saying that an empty mapping $p$ belongs to $P_\sigma$ if and only if there exists $h : G \to H$. Indeed, if this is true, $G$ is homomorphic to $H$ if and only if $P_\sigma$ contains the empty mapping, i.e., if $P_\sigma = \{ \emptyset \}$ (as opposed to $P_\sigma = \emptyset$).

We prove the aforementioned equivalence not only for $\sigma$, but for every subexpression $\tau \subseteq \sigma$. The proof of each implication is provided in a form of a separate claim.

**Claim 1.** If $p \in P_\tau$, then (i) is satisfied, i.e., there exists $h : G_\tau \to H$ such that for every $i \in L_\tau$ we have $p(i) \subseteq S(h(V_i))$.

**Proof of Claim.** First, we consider the case that corresponds to the basis of our algorithm.

$\tau = i(v)$ for some $i \in [k]$. Since $p(i) \in S(H)$, by Observation 2, there exists $u \in V(H)$ such that $p(i) \subseteq N_H(u)$. Then $p$ describes the homomorphism $h : G_\tau \to H$ defined by $h(v) = u$.

Now let us assume that $\tau$ is a non-leaf node, and that the claim holds for all its children. There are three possible types of $\tau$.

$\tau = \rho_{1\to j}(\tau')$ for a unique child $\tau'$ of $\tau$. Let $p' \in P_{\tau'}$ be a function such that $p$ arises from $p'$ in the construction of $P_\tau$. Consider a witness $h : G_{\tau'} \to H$ of $p' \in P_{\tau'}$.

For every $\ell \in L_{\tau'} \setminus \{ j \}$ we have $p(\ell) = p'(\ell) \subseteq S(h(V_{\ell}')) = S(h(V_\ell))$, where the first equality follows from the definition of $p$, the containment is by the inductive assumption, and the last equality is because $V_\ell' = V_\ell$. If now $i \notin L_{\tau'}$, then $j \notin L_{\tau'}$, and thus $p$ describes $h$ in $\tau$.

Hence assume that $i \in L_{\tau'}$ and note that in such a case $j \in L_\tau$. Now $p(j) = p'(i) \subseteq S(h(V_i'))$, by definition of $p$ and the inductive assumption, and if additionally $j \in L_{\tau'}$, then $p(j) = p'(j) \subseteq S(h(V_j'))$. By Observation 2, if $j \in L_{\tau'}$, we have
\[
p(j) \subseteq S(h(V_i')) \land S(h(V_j')) = S(h(V_i')) \cup S(h(V_j')) = S(h(V_{\tau'})) = S(h(V_\tau)).
\]

Hence $p$ describes $h$ in $\tau$. On the other side, if $j \notin L_{\tau'}$, then $V_\tau = V_i'$, and we have $p(j) \subseteq S(h(V_i')) = S(h(V_\tau))$. Again, $p$ describes $h$ in $\tau$.

$\tau = \tau_1 \oplus \tau_2$ for children $\tau_1$ and $\tau_2$ of $\tau$. Let $p_1 \in P_{\tau_1}, p_2 \in P_{\tau_2}$ be functions such that $p = p_1 \cup p_2$.

Let $h_i$ be a witness of $p_i$ in $\tau_i$, for $i = 1, 2$. As the domains of $h_1$ and $h_2$ are disjoint, we can define $h = h_1 \cup h_2$. Since there are no edges between $V(G_{\tau_1})$ and $V(G_{\tau_2})$ in $G_\tau$, $h$ is a homomorphism from $G_\tau$ to $H$.

ACM Transactions on Algorithms, Vol. 20, No. 3, Article 19. Publication date: May 2024.
Observe that for all \( \ell \in L_{r_1} \setminus L_{r_2} \), we have \( p(\ell) = p_1(\ell) \subseteq S(h_1(V_{r_1}^\ell)) = S(h(V_{r_1}^\ell)) \), by inductive assumption, similarly for \( \ell \in L_{r_2} \setminus L_{r_1} \). For \( \ell \in L_{r_1} \cap L_{r_2} \), for every \( i \in \{1, 2\} \) by inductive assumption we have \( p(\ell) = p_i(\ell) \subseteq S(h_i(V_{r_i}^\ell)) \). Therefore, again by Observation 2:

\[
p(\ell) \subseteq S(h_1(V_{r_1}^\ell)) \cap S(h_2(V_{r_2}^\ell)) = S(h_1(V_{r_1}^\ell) \cup h_2(V_{r_2}^\ell)) = S(h(V_{r_1}^\ell)).
\]

Hence \( h \) is a witness of \( p \) in \( \tau \).

\( \tau = \eta_{i,j}(\tau') \) for a unique child \( \tau' \) of \( \tau \). Let \( p' \in P_{\tau'} \) be a function such that \( p \) arises from \( p' \) in the construction of \( P_{\tau'} \). Consider a witness \( h : G_{\tau'} \to H \) of \( p' \in P_{\tau'} \). First, we note that \( h \) is a homomorphism from \( G_{\tau} \) to \( H \). For that, it is enough to show that \( h \) preserves edges between \( V_{\tau'}^i \) and \( V_{\tau'}^j \). Recall that \( p'(i) \supseteq S(p'(j)) \) by definition of \( P_{\tau} \), and the inductive assumption gives us that \( p'(i) \subseteq S(h(V_{\tau'}^i)) \) and \( p'(j) \subseteq S(h(V_{\tau'}^j)) \). Along with Observations 2 and 3 this results in

\[
S(h(V_{\tau'}^i)) \supseteq p'(i) \supseteq S(p'(j)) \supseteq S(S(h(V_{\tau'}^i))) \supseteq h(V_{\tau'}^j).
\]

Hence \( h(V_{\tau'}^i) \times h(V_{\tau'}^j) \subseteq E(H) \), so \( h \) is a homomorphism from \( G_{\tau} \) to \( H \). By construction, for every \( \ell \in L_{\tau} \), it holds that \( p(\ell) \subseteq p'(\ell) \subseteq S(h(V_{\tau'}^\ell)) \). Therefore \( h \) witnesses \( p \) in \( \tau \), and that concludes the proof of claim.

Now we show the converse implication.

**Claim 2.** Let \( p : L_{\tau} \to S(H) \). Assume that (I) is satisfied, i.e., there exists \( h : G_{\tau} \to H \) such that for every \( i \in L_{\tau} \) we have \( p(i) \subseteq S(h(V_{\tau}^i)) \). Then \( p \in P_{\tau} \).

**Proof of Claim.** Again, we first consider the case that corresponds to the basis of our algorithm. \( \tau = i(\nu) \) for some \( i \in [k] \). In this case clearly \( p \in P_{\tau} \), as \( L_{\tau} = \{i\} \) and \( P_{\tau} \) contains all possible functions from \( \{i\} \) to \( S(H) \).

Now let us assume that \( \tau \) is a non-leaf node, and that the claim holds for all its children. There are three possible types of \( \tau \).

\( \tau = \eta_{i,j}(\tau') \) for a unique child \( \tau' \) of \( \tau \).

If \( i \notin L_{\tau'} \), then also \( j \notin L_{\tau} \). Moreover, in this case \( j \notin L_{\tau} \). Indeed, assume to the contrary that \( j \in L_{\tau} \). By definition of a live label, this means that \( G \) contains edges incident to \( V_{\tau'}^j \) that do not appear in \( G_{\tau} \). By definition of the clique-width expression, such an edge exists for every \( v \in V_{\tau'}^j \), in particular for every \( v \in V_{\tau}^i \subseteq V_{\tau'}^i \), contradicting the fact that \( i \notin L_{\tau'} \). Hence \( j \notin L_{\tau} \), in particular \( j \notin L_{\tau'} \). Therefore, both \( i \) and \( j \) are not live neither in \( \tau \) nor in \( \tau' \), so \( L_{\tau'} = L_{\tau} \). By the inductive assumption we have that \( p \in P_{\tau} \). Recall that we set \( P_{\tau} = P_{\tau'} \) in this case.

If now \( i \in L_{\tau'} \), then \( j \notin L_{\tau} \). Define \( p' : L_{\tau'} \to S(H) \) to be such that for every \( \ell \in L_{\tau'} \setminus \{i\} \subseteq L_{\tau} \) it holds that \( p'(\ell) = p(\ell) \), and \( p'(i) = p(j) \). By our assumption on \( h \), for every \( \ell \in L_{\tau'} \setminus \{i, j\} \) we have that

\[
p'(\ell) = p(\ell) \subseteq S(h(V_{\tau}^i)) = S(h(V_{\tau'}^i)).
\]

Moreover,

\[
p'(i) = p(j) \subseteq S(h(V_{\tau}^i)) = S(h(V_{\tau}^i) \cup h(V_{\tau}^j)) = S(h(V_{\tau}^i) \cup h(V_{\tau'}^j)) = S(h(V_{\tau'}^i)) \cap S(h(V_{\tau'}^j)),
\]

where the last equality is by Observation 2. This in particular means that if \( j \notin L_{\tau'} \), by definition of \( p' \) we have that \( p'(j) = p(j) = p'(i) \subseteq S(h(V_{\tau'}^i)) \cap S(h(V_{\tau'}^j)) \). Hence, \( p'(i) \subseteq S(h(V_{\tau'}^i)) \) and, if \( j \in L_{\tau'} \), then \( p'(j) \subseteq S(h(V_{\tau'}^j)) \). Therefore, by the inductive assumption we have that \( p' \in P_{\tau'} \).

Now it is straightforward to verify that if \( j \notin L_{\tau} \) then \( p = (p' \setminus \{i, p'(i)\}) \cup \{j, p'(i)\} \in P_{\tau} \). Otherwise, i.e., if \( j \in L_{\tau} \), we conclude that \( p = p'|_{L_{\tau}} \in P_{\tau} \).
While Theorem 2.1 serves as the upper bound that matches our target SETH-based lower bounds which is smaller than the complexity obtained by the immediate application of Theorem 2.1 without will be mapped to. For each of the disjoint connected components of the input one can then invoke

Theorem 2.1 for the corresponding connected target graph, yielding an overall runtime of

$$p_i(\ell) = p(\ell) \subseteq S(h(V^i_\ell)) \subseteq S(h(V^i_\ell)) = S(h_i(V^i_\ell)).$$

Hence, by the inductive assumption, for every $i \in \{1, 2\}$ we have $p_i \in P_{\tau_i}$. This, combined with the definition of $P_\tau$, implies that $p \in P_\tau$. 

$$\tau = \eta_{i,j}(\tau')$$ for a unique child $\tau'$ of $\tau$. Recall that the first item of our preprocessing stage guarantees that $i, j \in L_{\tau'}$. We define $p': L_{\tau'} \to S(H)$ as follows.

$$p'(\ell) = \begin{cases} S(h(V^j_\ell)) & \text{if } \ell \in \{i, j\}, \\ p(\ell) & \text{otherwise.} \end{cases}$$

By the assumption on $h$, for every $\ell \in L_{\tau'}$ we have that $p'(\ell) \subseteq S(h(V^j_\ell))$. By the inductive assumption, $p' \in P_{\tau'}$. 

We observe that since $h$ is a homomorphism, for every $a \in h(V^j_\ell) = h(V^j_\ell)$ and every $b \in h(V^j_\ell) = h(V^j_\ell)$ we have that $ab \in E(H)$. Therefore, $h(V^j_\ell) \subseteq S(h(V^j_\ell))$, so by Observation 2 $S(S(h(V^j_\ell))) \subseteq S(h(V^j_\ell))$ and furthermore $S(S(h(V^j_\ell))) \subseteq S(S(S(h(V^j_\ell))))$. By Observation 3, as $S(h(V^j_\ell)) \subseteq S(H)$, we have that $S(S(S(h^j_\ell))) = S(h(V^j_\ell))$. Thus, $S(S(S(h^j_\ell))) \subseteq S(h(V^j_\ell))$, or equivalently $S(p'(j)) \subseteq p'(i)$. For $\ell \in \{i, j\}$ we have $p(\ell) \subseteq S(h(V^j_\ell)) = p'(\ell)$, and for every $\ell \in L_{\tau'} \setminus \{i, j\}$ we have $p(\ell) = p'(\ell)$. Therefore, $p \in P_\tau$ by the definition of $P_\tau$. This concludes the proof of claim. 

For the running time of our algorithm, we note that $|P_\tau| \leq s(H)^t$ for each subexpression $\tau$ of $\sigma$. This means that in each step, the computation requires time $O(t \cdot s(H)^2 \cdot s(H)^t)$. Overall this yields a complexity of

$$O(|V(T)| \cdot t \cdot s(H)^2 \cdot s(H)^t) \subseteq O^*(s(H)^t).$$

### 5 ON PRODUCTS AND PROJECTIVITY

While Theorem 2.1 serves as the upper bound that matches our target SETH-based lower bounds for $\text{Hom}(H)$ for the “most difficult” choices of $H$, in many cases one can in fact supersede the algorithm’s runtime by exploiting well-known properties $\text{Hom}(H)$ on more simple target graphs.

One obvious example for this are disconnected target graphs; for these one can branch for each connected component of the input graph, on the connected component in the target graph that it will be mapped to. For each of the disjoint connected components of the input one can then invoke Theorem 2.1 for the corresponding connected target graph, yielding an overall runtime of

$$O^*(\max_{C \text{ conn. comp. of target}} s(C)^{cw(G)})$$

which is smaller than the complexity obtained by the immediate application of Theorem 2.1 without any prior branching (cf. Observation 5):

$$O^*(\sum_{C \text{ conn. comp. of target}} s(C)^{cw(G)})$$

But there are even connected target graphs for which we can argue the same behaviour. As a simple example showcasing this, consider the wheel graph $W_6$ (see Figure 2). Since $W_6$ is 3-colorable,
it holds that $W_6 \rightarrow K_3$, and since $K_3$ is a core and an induced subgraph of $W_6$, it is the core of $W_6$. We recall that if $H$ is a core of $H'$, then for every graph $G$ it holds that $G \rightarrow H$ if and only if $G \rightarrow H'$. Hence, having an instance $G$ of Hom($W_6$), we can compute a core of $W_6$ (since we assume that the target graph is fixed, this can be done in constant time), and use Theorem 2.1 for $H = K_3$ to decide whether $G \rightarrow W_6$ in total running time $O^*(s(K_3)^{cw(G)})$. As $s(K_3) < s(W_6)$ (as showcased in Figure 2), this yields a better running time bound than the direct use of Theorem 2.1. While this example shows that the signature number can decrease by taking an induced subgraph, we remark that it can never increase.

**Lemma 5.1.** Let $H$ and $H'$ be graphs such that $H$ is an induced subgraph of $H'$. Then $s(H) \leq s(H')$.

**Proof.** Given a simple graph $Q$, one may consider an equivalence relation $\sim_Q$ on the set of non-empty subsets of $V(Q)$ defined as follows: $V_1 \sim_Q V_2$ if and only if $V_1$ and $V_2$ have the same signature sets in $Q$. Observe that $s(Q)$ is equal to the number of equivalence classes of $\sim_Q$ minus one (as there are subsets $V$ such that $S(V) = \emptyset$). Hence, to prove the claim, it suffices to show that whenever two subsets of $V(H)$ belong to different equivalence classes of $\sim_H$, they also belong to different equivalence classes of $\sim_{H'}$. For this, consider any two non-empty subsets $V_1$ and $V_2$ of $V(H)$ such that $V_1 \not\sim_H V_2$.

Since $S(V_1) \neq S(V_2)$, by symmetry of $V_1$ and $V_2$, we can assume that there exists $v \in S(V_1) \setminus S(V_2)$. Then for every $t \in V_1$ we have $vt \in E(H) \subseteq E(H')$, i.e., $v$ belongs to the signature set of $V_1$ in $H'$. Moreover, there exists $t_0 \in V_2$ such that $vt_0 \notin E(H)$, as otherwise $v$ would belong to $S(V_2)$. As $H$ is induced subgraph of $H'$, it means that $vt_0 \notin E(H')$. Hence, $v$ does not belong to the signature set of $V_2$ in $H'$ and $V_1 \not\sim_{H'} V_2$. \hfill $\square$

At this point, we may ask whether the procedure of simply computing the unique core $H$ of the fixed target $H'$ and then applying Theorem 2.1 for $H$ could yield a tight upper bound for Hom($H'$). Unfortunately, the situation is more complicated than that, and we need to introduce a few important notions to capture the problem’s fine-grained complexity.

Let the **direct product** $H_1 \times H_2$ of graphs $H_1, H_2$ be the graph defined as follows:

\[
V(H_1 \times H_2) = V(H_1) \times V(H_2),
\]

\[
E(H_1 \times H_2) = \{(x_1, x_2)(y_1, y_2) : x_iy_i \in E(H_i) \text{ for every } i \in \{1, 2\}\}.
\]

We call $H_1$ and $H_2$ the **factors** of $H_1 \times H_2$. Clearly, the operation $\times$ is commutative, and since it is also associative, we can naturally extend the definition of direct product to more than two factors, i.e., $H_1 \times H_2 \times \ldots \times H_m = H_1 \times (H_2 \times \ldots \times H_m)$. Note that for every graph $H$ it holds that $H \times K_1^* = H$.

In the remaining part of the article we often consider vertices that are tuples. If such a vertex is an argument of some function and in cases where this does not lead to confusion, we omit one pair of brackets; similarly, we omit internal brackets in nested tuples where this does not lead to confusion.
Moreover, for any graph $H$ and for an integer $\ell$, we denote by $H^\ell$ the graph $\underbrace{H \times \ldots \times H}_\ell$. As an example, instead of writing $((x_1, x_2), y_1) \in V((H_1 \times H_1) \times H_2)$, we write $(x_1, x_2, y_1) \in V(H_1^2 \times H_2)$.

If $H = H_1 \times \ldots \times H_m$ for some graphs $H_1, \ldots, H_m$, we say that $H_1 \times \ldots \times H_m$ is a factorization of $H$. A graph $H$ on at least two vertices is prime if the fact that $H = H_1 \times H_2$ for some graphs $H_1, H_2$ implies that $H_1 = K^*_1$ or $H_2 = K^*_1$. If $H$ has a factorization $H_1 \times \ldots \times H_m$ such that for every $i \in [m]$ the graph $H_i$ is prime, we call $H_1 \times \ldots \times H_m$ a prime factorization of $H$.

**Theorem 5.2.** ([9, 16]). Any connected non-bipartite graph with more than one vertex has a unique prime factorization (into factors with possible loops).

Consider a graph $H_1 \times \ldots \times H_m$, and let $i \in [m]$. A mapping $\pi_i : V(H_1 \times \ldots \times H_m) \to V(H_i)$ such that $\pi_i(x_1, \ldots, x_r) = x_i$ is called the $(i$-th) projection of $H_1 \times \ldots \times H_m$. Clearly, such a mapping is always a homomorphism.

**Proposition 5.3.** Let $G, H_1, \ldots, H_m$ be graphs. Then $G \to H_1 \times \ldots \times H_m$ if and only if for every $i \in [m]$ we have $G \to H_i$.

**Proof.** Let $f : G \to H_1 \times \ldots \times H_m$. Then for every $i \in [m]$ we have a homomorphism $\pi_i : G \to H_i$. Conversely, if for every $i \in [m]$ we have $g_i : G \to H_i$, then we can define $g : G \to H_1 \times \ldots \times H_m$ as $g(x) = (g_1(x), \ldots, g_m(x))$. $\square$

Crucially, since there exist cores that are not prime [29], in some cases Proposition 5.3 allows us to improve the bounds given by Theorem 2.1 even if $H$ is a core, simply by considering all possible factorizations of $H$.

**Corollary 5.4.** Let $H$ be a graph with factorization $H_1 \times \ldots \times H_m$, and let $G$ be an instance graph of $\text{Hom}(H)$. Assuming that the clique-width expression $\sigma$ of $G$ of width $\text{cw}(G)$ is given, the $\text{Hom}(H)$ problem can be solved in time $O^\ast(\max_{i \in [m]} \text{cw}(H_i) \text{cw}(G))$.

**Proof.** Observe that if $G$ is an instance of $\text{Hom}(H)$, by Theorem 2.1 for every $i \in [m]$ we can decide whether $G \to H_i$ in time $O^\ast(\text{cw}(H_i))$. Then, if $G$ is a yes-instance of $\text{Hom}(H)$ for every $i \in [m]$, we return that $G$ is a yes-instance of $\text{Hom}(H)$. Otherwise, we return that $G$ is a no-instance of $\text{Hom}(H)$. The correctness of this procedure follows from Proposition 5.3. $\square$

Moreover, the notion of signature sets we introduced in the previous section behaves multiplicatively with respect to taking direct product of graphs.

**Lemma 5.5.** Let $H = H_1 \times H_2$. Then $S(H) = S(H_1) \times S(H_2)$.

**Proof.** We prove that $S(H)$ is of form $\{S(T_i) \times S(T_2) : T_i \subseteq V(H_1), S(T_2) \neq \emptyset \}$ for $i = 1, 2$. Let $T_1$ and $T_2$ be some subsets of, respectively, $V(H_1)$ and $V(H_2)$. Clearly,

$$S(T_1) \times S(T_2) = \left(\bigcap_{t \in T_1} N(t)\right) \times \left(\bigcap_{t' \in T_2} N(t')\right) = \bigcap_{(t, t') \in T_1 \times T_2} N(t, t') = S(T_1 \times T_2). \quad (2)$$

Therefore, if $S(T_1)$ and $S(T_2)$ are non-empty, we get that $S(T_1) \times S(T_2) \subseteq S(H)$.

To see that $S(H) \subseteq S(H_1) \times S(H_2)$, we show that for every $T \subseteq V(H)$ set $S(T)$ is of the form $S(T_1) \times S(T_2)$ for some $T_1, T_2$. Define $T_1$ and $T_2$ to be minimal sets such that $T \subseteq T_1 \times T_2$. Hence, by (2), $S(T_1) \times S(T_2) = S(T_1 \times T_2) \subseteq S(T)$. Moreover, for every $(s, s') \in S(T)$ we have $s \in \bigcap_{t \in T_1} N(t)$ and $s' \in \bigcap_{t' \in T_2} N(t')$, so the equality follows. $\square$
In particular, it follows from Lemma 5.5 that if $H$ is a graph with factorization $H_1 \times \ldots \times H_m$, then $s(H) = s(H_1) \ldots s(H_m)$. Therefore if there exist at least two factors $H_i, H_j$ such that $s(H_i) > 1$, $s(H_j) > 1$, Corollary 5.4 yields a better running time than Theorem 2.1.

To analyze the possible matching lower bounds for our algorithms, in the remaining part of the section, we focus only on connected non-trivial cores $H$ that are provided with their unique prime factorization $H_1 \times \ldots \times H_m$; if $H$ is prime, we technically consider this factorization to be $H \times K_1^*$ (noting that this is not a prime factorization, and that $K_1^*$ is the only non-simple graph in this article). We note that the factors of a core must satisfy some necessary conditions.

**Theorem 5.6. ([29]).** Let $H$ be a connected, non-trivial core with factorization $H = H_1 \times \ldots \times H_m$ such that $H_i \neq K_1^*$ for all $i \in [m]$. Then for every $i \in [m]$ the graph $H_i$ is a connected non-trivial core, incomparable with $H_j$ for $j \in [m] - \{i\}$.

Theorem 5.6 in particular implies that if $H$ is a connected non-trivial graph with factorization $H_1 \times \ldots \times H_m$, then at least one of the factors $H_i$ must be non-trivial, and that $K_1$ and $K_2$ never appear as factors of a connected non-trivial graph.

In the remaining part of this section we introduce a few more important definitions, in particular, the well-established notion of projectivity for non-trivial graphs $H$ [25, 27, 34].

We say that a homomorphism $f : H^\ell \rightarrow H$, for some $\ell \geq 2$, is idempotent if for each $x \in V(H)$ it holds that $f(x, \ldots, x) = x$. Graph $H$ is projective if for every $\ell \geq 2$, every idempotent homomorphism $f : H^\ell \rightarrow H$ is a projection. We note that every projective graph on at least three vertices must be connected, ramified, non-bipartite and prime [25].

Here, we introduce a generalization of the projectivity property for non-trivial cores, which turns out to be the central component required to establish the lower bound for our problem. As a first step toward this, we lift the notion of idempotency as follows. Let $H$ be a fixed non-trivial core with prime factorization $H_1 \times \ldots \times H_m$, and let $f: A \rightarrow H$ be a homomorphism where $A = H^\ell \times W$; observe that $\ell$ is uniquely determined by either $W$ being incomparable with the prime core $H$, or $W$ being $K_1^*$. We say that $f$ is $H$-idempotent if for each $x \in V(H)$, $y \in V(W)$ it holds that $f(x, \ldots, x, y) = x$.

Now, let us consider a non-trivial core $H$ which admits a prime factorization $H_1 \times \ldots \times H_m$ and let $i \in [m]$. We say that $H$ is $H_i$-projective if $H_i$ is non-trivial and every $H_i$-idempotent homomorphism $f : H_1 \times \ldots \times H_{i-1} \times H'_i \times H_{i+1} \times \ldots \times H_m \rightarrow H_i$ is a projection. In other words, for every homomorphism $f : H_1 \times \ldots \times H_{i-1} \times H'_i \times H_{i+1} \times \ldots \times H_m \rightarrow H_i$ such that for every $x \in V(H_1)$, $y_j \in V(H_j)$ for $j \in [m] - \{i\}$ it holds that $f(y_1, \ldots, y_{i-1}, x, \ldots, x, y_{i+1}, \ldots, y_m) = x$, we must have that there exists $q \in \{i, \ldots, i + \ell - 1\}$ such that $f = \pi_q$. Recall that if $H$ is a non-trivial projective core, then it must be prime, so $H \times K_1^*$ is its only possible factorization. It is straightforward to verify that in such case $H$ is $H$-projective.

Since the direct product of graphs is commutative, if $H = H_1 \times \ldots \times H_m$ is $H_i$-projective for some $i \in [m]$, to simplify the notation we often assume w.l.o.g. that $i = 1$.

**6 HARDNESS**

In this section, we focus on establishing the desired lower bounds, stated below.

**Theorem 2.2.** Let $H$ be a fixed non-trivial core with prime factorization $H_1 \times \ldots \times H_m$. If $H$ is $H_i$-projective for some $i \in M$ then there is no algorithm solving $\text{Hom}(H)$ in time $O^\ast((s(H_i) - \epsilon)^{cw(G)})$ for any $\epsilon > 0$, unless the SETH fails.

We divide our proof into two main steps. First, we show that in our setting, instead of considering the $\text{Hom}(H)$ problem, we may focus on the $\text{HOMOMORPHISM EXTENSION}$ problem (historically also studied under the equivalent formulation of a retract problem [1, 10, 17]), denoted $\text{HomExt}(H)$.
Fig. 3. An instance \((G, h')\) of \(\text{HomExt}(H)\) (upper left) and the graph \(H\) (upper right). A construction of \(G'\) (bottom left) where the colors of vertices of \(V'\) and \(\hat{H}\) illustrate which vertices are forced to be mapped to the same vertex of \(H\). A homomorphism \(f\) from \(G\) to \(H\) (bottom right) does not extend \(h'\). However, it is straightforward to verify that there exists an automorphism of \(H\) that can be composed with \(f\) to obtain an extension of \(h'\).

For a fixed \(H\), \(\text{HomExt}(H)\) takes as an instance a pair \((G', h')\), where \(G'\) is a graph and \(h' : V' \to V(H)\) is a mapping from some \(V' \subseteq V(G')\). We ask whether there exists an extension of \(h'\) to \(G'\), i.e., a homomorphism \(h : G' \to H\) such that \(h|_{V'} \equiv h'\).

The \(\text{HomExt}(H)\) is clearly a generalization of the \(\text{Hom}(H)\) problem. However, as the first step of our proof, we show that if \(H\) is a fixed non-trivial core, each instance \((G', h')\) of \(\text{HomExt}(H)\) can be transformed in polynomial time into an instance \(G\) of \(\text{Hom}(H)\), such that \(cw(G')\) and \(cw(G)\) differ only by a constant.

**Theorem 6.1.** Let \(H\) be a fixed non-trivial core. Given an instance \((G', h')\) of \(\text{HomExt}(H)\), we can construct an equivalent instance \(G\) of \(\text{Hom}(H)\) such that \(cw(G) \leq cw(G') + |V(H)|\).

**Proof.** Let \(V' \subseteq V(G')\) be the domain of \(h'\). We construct \(G\) as follows: first, we take \(G'\) and a copy \(\hat{H}\) of \(H\). For brevity, we slightly abuse the notation and assume that \(H\) and \(\hat{H}\) have the same set of vertices; formally in such a case we should define an isomorphism between \(H\) and \(\hat{H}\). Next, for every \(v \in V'\) we add all the edges with one endpoint in \(v\) and another one in \(N_{\hat{H}}(h'(v))\). That concludes the construction, see Figure 3.

We first prove that if there exists an extension \(h : G' \to H\) of \(h'\), then \(h\) can be further extended to \(G\), by setting \(h(v) = v\) for every \(v \in V(\hat{H})\). Indeed, to see that such \(h\) is a homomorphism from \(G\) to \(H\) consider \(uv \in E(G)\). If \(u, v \in V(G')\), then \(h(u)h(v) \in E(H)\), by the assumption that \(h\) is an extension of \(h'\) to \(G'\). If \(u, v \in V(\hat{H})\), then simply \(uv \in E(H)\), thus, \(h(u)h(v) = uv \in E(H)\). Finally,
assume that \( u \in V(G'), u \in V(\tilde{H}) \). Note that, by definition of \( G \), this can happen only if \( u \in V' \) and \( v \) is adjacent to \( h'(u) \in V(\tilde{H}) \). Hence, \( h(u)h(v) = h'(u)v \in E(H) \).

For the reverse direction, assume that there exists a homomorphism \( f : G \rightarrow H \). We show that there exists an extension \( h : G' \rightarrow H \) of \( h' \). Let \( \sigma : \tilde{H} \rightarrow H \) be the restriction of \( f \) to \( \tilde{H} \). Since \( H \) is a core, \( \sigma \) is an isomorphism. We claim that the function \( g \equiv \sigma^{-1} \circ f : V(G) \rightarrow V(H) \) is an extension of \( h' \) to \( G \) (so in particular, is an extension of \( h' \) to \( G' \)). Observe that by definition of \( g \), if \( v \in V(\tilde{H}) \), then \( g(v) = v \).

Clearly, \( g \) is a composition of homomorphisms, so also a homomorphism. Therefore, it remains to show that for every \( v \in V' \) we have \( h'(v) = g(v) \). Consider a vertex \( v \in V' \). Since \( H \) is a core, and \( N_{\tilde{H}}(h'(v)) \subseteq N_G(v) \), we have that \( f(v) = f(h'(v)) \). It follows that
\[
g(v) = \sigma^{-1}(f(v)) = \sigma^{-1}(f(h'(v))) = g(h'(v)).
\]
However, recall that for every \( u \in V(\tilde{H}) \) we have that \( g(u) = u \), so since \( h'(v) \in V(\tilde{H}) \), in particular, \( g(u) = h'(v) \).

To see that \( \text{cw}(G) \leq \text{cw}(G') + |V(H)| \), observe that we added exactly \( |V(H)| \) vertices to \( G' \). This means we can modify a clique-width expression \( \sigma \) for \( G' \) to obtain a clique-width expression of \( G \) as follows. Each added vertex is introduced with a designated label that is distinct from all labels used in \( \sigma \). Then each subexpression of \( \sigma \) that introduces a vertex of \( G' \) can be replaced by an expression that introduces the vertex and inserts all required edges to the added vertices. Finally, one can insert the missing edges between added vertices.

As the second step, we prove the following theorem.

**Theorem 6.2.** Let \( H \) be a fixed non-trivial core with prime factorization \( H_1 \times \ldots \times H_m \). Assume that \( H \) is \( H_i \)-projective for some \( i \in [m] \). Then there is no algorithm solving \( \text{HomExt}(H) \) on instances \( (G', h') \) in time \( O^\star \left((s(H_i) - \varepsilon)^{\text{cw}(G')}\right) \) for any \( \varepsilon > 0 \), unless the SETH fails.

Before we proceed to the proof of Theorem 6.2, we show that it implies Theorem 2.2.

**Lemma 6.3.** Let \( H \) be a fixed non-trivial core with prime factorization \( H_1 \times \ldots \times H_m \). Assume that \( H \) is \( H_i \)-projective for some \( i \in [m] \), and that there is no algorithm solving \( \text{HomExt}(H) \) in time \( O^\star \left((s(H_i) - \varepsilon)^{\text{cw}(G')}\right) \) on instances \( (G', h') \) for any \( \varepsilon > 0 \) unless the SETH fails. Then there is no algorithm solving \( \text{Hom}(H) \) in time \( O^\star \left((s(H_i) - \varepsilon)^{\text{cw}(G)}\right) \) on instances \( G \) in time for any \( \varepsilon > 0 \), unless the SETH fails.

**Proof.** Let \( H \) be a non-trivial core with a prime factorization \( H_1 \times \ldots \times H_m \). W.l.o.g. assume that \( i = 1 \). Suppose that Theorem 2.2 does not hold, i.e., there exists an algorithm \( A \) that solves every instance \( G \) of \( \text{Hom}(H) \) in time \( O^\star \left((s(H_1) - \varepsilon)^{\text{cw}(G)}\right) \).

Let \( (G', h') \) be an instance of \( \text{HomExt}(H) \). We use Theorem 6.1 to transform \( (G', h') \) into an equivalent instance \( G \) of \( \text{Hom}(H) \), such that \( \text{cw}(G) \leq \text{cw}(G') + |V(H)| \). Then, we use \( A \) to decide whether \( G \rightarrow H \) in time
\[
O^\star \left((s(H_1) - \varepsilon)^{\text{cw}(G)}\right) = O^\star \left((s(H_1) - \varepsilon)^{\text{cw}(G')} \cdot (s(H_1) - \varepsilon)^{|V(H)|}\right).
\]
Since \( H \) is a fixed graph, \( (s(H_1) - \varepsilon)^{|V(H)|} \) is a constant, and therefore \( O^\star \left((s(H_1) - \varepsilon)^{\text{cw}(G)}\right) = O^\star \left((s(H_1) - \varepsilon)^{\text{cw}(G')}\right) \). Since \( G \rightarrow H \) if and only if \( (G', h') \) is a yes-instance of \( \text{HomExt}(H) \), we get a contradiction with Theorem 6.2.

We prove Theorem 6.2 for \( i = 1 \), which covers other cases by commutativity of direct products. We begin by constructing certain gadgets.
Let $H$ be a fixed core with factorization $H_1 \times \ldots \times H_m$. We define $W = H_2 \times \ldots \times H_m$ if $m \geq 2$, and $W = K^*_1$ otherwise. Clearly, $H_1 \times W$ is a (not necessarily prime) factorization of $H$. Moreover, if for some graph $G$ we have a homomorphism $f : G \to H_1 \times \ldots \times H_m$, for $i \in [m]$ we denote by $f_i$ the homomorphism $\pi_i \circ f : G \to H_i$.

Let $S$ be a set of pairs of vertices of $H_1$, and let $w, w' \in V(W)$. We say that a tuple $(F, h', p, q)$, such that $F$ is a graph, $h' : V' \to H$ is a mapping with domain $V' \subseteq V(F)$, and $p, q \in V(F)$, is an $(S, w, w')$-gadget if

(S1) for every extension $h : F \to H$ of $h'$, it holds that $(h_1(p), h_1(q)) \in S$;
(S2) for every pair $(s_1, s_2) \in S$ there exists an extension $h : F \to H$ of $h'$ such that $h(p) = (s_1, w)$ and $h(q) = (s_2, w')$.

Lemma 6.4. Let $H$ be a non-trivial connected core with factorization $H_1 \times W$, let $S \subseteq V(H_1)^2$, and let $w, w' \in V(W)$. Assume that $H$ is $H_1$-projective. Then there exists an $(S, w, w')$-gadget.

Proof. Let $L = \{(s_1^1, s_2^1), \ldots, (s_1^l, s_2^l)\}$. Define

$$F = H_1^l \times W, \quad p = (s_1^1, \ldots, s_1^l, w), \quad q = (s_2^1, \ldots, s_2^l, w').$$

Let $V' = \{(x, x, \ldots, x, y) \mid x \in V(H_1), y \in V(W)\}$, and let $h'(x, \ldots, x, y) = (x, y)$. We claim that $(F, h', p, q)$ is an $(S, w, w')$-gadget.

The condition (S1) follows from the fact that $H$ is $H_1$-projective. Indeed, if $h : F \to H_1 \times W$ is an extension of $h'$, observe that $h_1$ must be $H_1$-idempotent, and hence a projection on one of the $\ell$ first coordinates. Therefore, we must have $(h_1(p), h_1(q)) \in S$.

For (S2), take any $(s_1^i, s_2^i) \in S$ and let $h : F \to H_1 \times W, h(x) = (\pi_1(x), \pi_{\ell+1}(x))$. Clearly, $h'$ is an extension of $h'$, and it is easy to verify that $h(p) = (s_1^i, w)$ and $h(q) = (s_2^i, w')$. \hfill $\Box$

We say that $S \subseteq V(H_1)^2$ is proper, if for every coordinate there exist two elements in $S$ that differ on that coordinate, i.e., $S$ is not of the form $\{s\} \times U$ nor $U \times \{s\}$ for some $s \in V(H_1)$ and $U \subseteq V(H_1)$. Note that if $S$ is proper and $(F, h', p, q)$ is an $(S, w, w')$-gadget constructed as in Lemma 6.4, then neither $p$ nor $q$ belong to the domain of $h'$.

For fixed vertices $a, b \in V(H_1)$, let $S_{a,b} = \{(a', b') : a' \neq a, b' \in V(H_1)\} \cup \{(a, b)\}$. We call the $(S_{a,b}, w, w')$-gadget $(F, h', p, q)$ an $((a, b), w, w')$-implication-gadget. Intuitively, an $((a, b), w, w')$-implication-gadget works as the implication $a \Rightarrow b$, since in every homomorphism $h : F \to H$ that extends $h'$, if $h_1(p) = a$, then $h_1(q) = b$.

Let $a, b, c \in V(H_1)$, $w \in V(W)$, and let $t$ be an integer. A triple $(F, h', R)$ such that $F$ is a graph, $h' : V' \to H_1 \times W$ is a partial mapping from some $V' \subseteq V(F)$, and $R$ is a subset of $V(F)$ of cardinality $t$ is an $t$-or-gadget with domain $((a, b, c), w)$ if

(O1) for every homomorphism $h : F \to H$ that is an extension of $h'$, and for every $u \in R$ we have that $h_1(u) \in \{a, b, c\}$ and there exists $v \in R$ such that $h_1(v) = a$;
(O2) for every $v \in R$ there exists a homomorphism $h : F \to H$ that is an extension of $h'$, such that $h(u) = (a, w)$ and for every $u \in R \setminus \{v\}$ it holds that $h(u) \in \{(b, w), (c, w)\}$.

Lemma 6.5. Let $H$ be a non-trivial core with factorization $H_1 \times W$. Assume that $H$ is $H_1$-projective. Then for every distinct $a, b, c \in V(H_1)$, every $w \in V(W)$ and every $t$, there exists a $t$-or-gadget $(F, h', R)$ with domain $((a, b, c), w)$.

Proof. We consider separately the cases $t = 1$ and $t = 2$. Observe that in case $t = 1$ our gadget needs to be a graph that has a vertex $r \in R$ that is always mapped to $(a, w)$. Hence, we set $F = K_1$, $R = V(F)$, and $h'(v) = (a, w)$ for $v \in V(F)$.

If $t = 2$, let $S = \{(a, b), (b, a), (a, a)\}$, we introduce an independent set $R = \{r_1, r_2\}$ and $(S, w, w')$-gadget $(F, h', r_1, r_2)$. To see that $(F, h', R)$ satisfies (O1), consider any extension $f : F \to H$ of
we can assume that $h'$ is $(F, h', r_1, r_2)$-gadget, we have $(f_1(r_1), f_1(r_2)) \in \{(a, b), (a, a), (b, a)\}$. For (O2), recall that by the property (S2) of $S$-gadget there exist extensions $f^{(1)}$ and $f^{(2)}$ of $h'$ such that $(f_1^{(1)}(r_1), f_1^{(1)}(r_2)) = (a, b)$ and $(f_1^{(2)}(r_1), f_1^{(2)}(r_2)) = (b, a)$.

Assume then that $t > 2$, and let

$$S = \{a, b, c\}^2 - \{(b, c), (c, b)\},$$
$$S_{\text{left}} = \{(a, a), (a, b), (a, c), (c, a), (c, c)\},$$
$$S_{\text{right}} = \{(a, a), (a, b), (b, a), (b, b), (c, c)\}$$

be subsets of $V(H'_1)^2$. We introduce an independent set $R = \{r_1, \ldots, r_t\}$ of $t$ vertices and one copy of $(S_{\text{left}}, w, w)$-gadget $(F_1, h'_1, r_1, r_2)$. Then, for $j \in \{2, \ldots, t - 2\}$, we introduce an $(S, w, w)$-gadget $(F_j, h'_j, r_j, r_{j+1})$ (we note that if $t = 3$, we do not introduce these). Last, we introduce one copy of $(S_{\text{right}}, w, w)$-gadget $(F_{t-1}, h'_{t-1}, r_{t-1}, r_t)$. We note that sets $S, S_{\text{left}},$ and $S_{\text{right}}$ are proper, so the domains of the partial mappings $h'_j$, $j \in \{1, \ldots, t - 1\}$, are pairwise disjoint. In particular, the union $h' = \bigcup_{j=1}^t h'_j$ is a well-defined mapping. We define $F$ to be the union of all the graphs from the introduced gadgets and claim that $(F, h', R)$ is a $t$-or-gadget.

We first show that (O1) holds. Assume that there exists an extension $f : F \to H$ of $h'$, and $j' \in [t]$ such that $f_j(r_{j'}) \not\in \{a, b, c\}$. This implies that there exists $j \in \{j' - 1, j'\}$ such that $(f_1(r_j), f_1(r_{j+1})) \not\in S'$ for any $S' \in \{S_{\text{left}}, S_{\text{right}}\}$. This is a contradiction with $(F, h'_j, p, q_j)$ being an $(S', w, w)$-gadget, as it violates (S1).

Now assume that there exists an extension $f : F \to H$ of $h'$ such that for every $j \in [t]$ we have that $f_1(r_j) \in \{b, c\}$. The definition of $S_{\text{left}}$ and $S_{\text{right}}$, respectively, implies that $f_1(r_j) = c$ and $f_1(r_j) = b$. Hence, there exists $j \in [t - 1]$ such that $f_1(r_j) = c$ and $f_1(r_{j+1}) = b$. However, observe that the pair $(c, b)$ does not belong to set $S'$, for $S' \in \{S_{\text{left}}, S_{\text{right}}\}$, and since we introduced an $S'$-gadget from $r_j$ to $r_{j+1}$, this leads to a contradiction.

To see that (O2) holds as well, fix some $r_j \in R$ and define

$$f'(r_j) = \begin{cases} 
(a, w), & \text{if } \ell = j, \\
(c, w), & \text{if } \ell < j, \\
(b, w), & \text{if } \ell > j.
\end{cases}$$

If $j = 1$, then since $(a, b) \in S_{\text{left}}, (b, b) \in S$ and $(b, b) \in S_{\text{right}}$, the property (S2) asserts that we can construct a homomorphism $f : F \to H$ that extends $h'$ and $f'$. The same holds also if $j = t$, (since $(c, c) \in S_{\text{left}}, (c, c) \in S$ and $(c, a) \in S_{\text{right}}$), and if $1 < j < t$ (since $(c, c) \in S_{\text{left}}, (c, c), (c, a), (a, b), (b, b) \in S$ and $(b, b) \in S_{\text{right}}$).

Finally, all that remains is to prove Theorem 6.2. Our reduction generalizes the construction used by Lampis [23] to reduce an $\text{SETH}$ lower-bounded $\text{CSP}$ to $\text{k-COLORING}$. Intuitively speaking, in that construction possible variable assignments are encoded by mapping specified vertices to arbitrary non-trivial subsets of the colors. The straightforward generalization of this approach to our setting would be to map to non-trivial subsets of $V(H)$. However, the structure of $H$ allows only certain configurations of subsets as images for the specified vertices in a solution for $\text{Hom}(H)$—which is precisely where the signature sets come into play.

Let $q, B \geq 2$ be integers. We reduce from the $q$-$\text{CSP-B}$ problem that is defined as follows. An instance of $q$-$\text{CSP-B}$ consists of a set $X$ of variables and a set $C$ of $q$-constraints. A $q$-constraint $c \in C$ is a $q$-tuple of elements from $X$ and a set $P(c)$ of $q$-tuples of elements from $[B]$ (i.e., $P(c) \subseteq [B]^q$). The $q$-$\text{CSP-B}$ problem asks whether there exists an assignment $\gamma : X \to [B]$, such that each constraint is satisfied, i.e., if $c = ((x_1, \ldots, x_q), P(c)) \in C$, then $(\gamma(x_1), \ldots, \gamma(x_q)) \in P(c)$. Note that we can assume that $q$-constraints in our $q$-$\text{CSP-B}$ instance may have less than $q$ vertices, as it is
always possible to add at most \( q - 1 \) dummy variables to \( X \) and add them to constraints that are of smaller size.

We need the following theorem.

**Theorem 6.6.** ([23]) For any \( B \geq 2, \varepsilon > 0 \) we have the following: assuming the SETH, there exists \( q \) such that \( n \)-variable \( q\)-CSP-B cannot be solved in time \( O^*((B - \varepsilon)^n) \).

We have all the tools to perform the final reduction.

**Proof of Theorem 6.2.** Recall that it is sufficient to prove the theorem when \( H = H_1 \times W \) is non-trivial \( H_1 \)-projective core \( (W = K_1^3 \text{ if } H = H_1) \). Fix \( \varepsilon > 0 \) and set \( B = s(H_1) \). As \( H \) is \( H_1 \)-projective, \( H_1 \) is non-trivial and hence contains at least three distinct vertices \( a, b, \) and \( c \). In particular, \( B \geq 3 \) by Lemma 3.1. Since \( H = H_1 \times W \) is a non-trivial core, \( W \) must have at least one edge \( w w' \) (it may happen that \( w = w' \)). From now on \( a, b, c, w \) and \( w' \) are fixed. Let \( q \) be the smallest number such that \( q\)-CSP-B on \( n \) variables cannot be solved in time \( O^*((B - \varepsilon)^n) \) assuming the SETH, given by Theorem 6.6.

Let \( \varphi \) be an instance of \( q\)-CSP-B, where \( X = \{x_1, \ldots, x_n\} \) is the set of variables and \( C = \{c_0, \ldots, c_{m-1}\} \) is the set of constraints. For every \( j \in \{0, \ldots, m-1\} \) denote by \( X_j \) the set of variables that appear in the constraint \( c_j \). Let \( P(c_j) = \{f_{1,j}^j, \ldots, f_{\ell,j}^j\} \) be the set of assignments from \( X_j \) to \([B]\) that satisfy the constraint \( c_j \). Let \( L = m(n[H_1] + 1) \), and let \( \lambda : [B] \to S(H_1) \) be some fixed bijection.

We construct the instance \( G_\varphi \) of \( \text{HomExt}(H) \). For each \( j \in \{0, \ldots, L - 1\} \), let \( j' = j \mod m \). Let \( R_j = \{r_{1,j}', \ldots, r_{\ell,j}'\} \), where each vertex \( r_{1,j}' \) corresponds to the assignment \( f_{1,j}^j \). We introduce the \( p_{j'}\)-or-gadget \( (F_j, h_j', R_j) \) with domain \((a, b, c, w)\).

For each \( x_i \in X_j \), and for each \( f_{k,j}^j \in P(c_{j'}) \) we do the following:

1. Let \( y = f_{k,j}^j(x_i) \in [B] \). Construct an independent set \( V_{i,j,k} \) of \( |\lambda(y)| \) vertices and an independent set \( U_{i,j,k} \) of \( |S(\lambda(y))| \) vertices.
2. For each \( d \in \lambda(y) \) select a distinct vertex \( z_d \in V_{i,j,k} \) and add an \((a, d), w, w'\)-implication-gadget from \( r_{k,j}' \) to \( z_d \). For each \( d \in S(\lambda(y)) \) select a distinct vertex \( z_d \in U_{i,j,k} \) and add an \((a, d), w, w'\)-implication-gadget from \( r_{k,j}' \) to \( z_d \).
3. Connect all vertices of \( U_{i,j,k} \) with all vertices of previously constructed sets \( V_{i,j',k'} \) for \( j' < j \) and \( k' \in [\ell] \) (see Figure 4).

The partial mapping \( h' \) is the union of all the partial mappings that are introduced by all the gadgets. This finishes the construction of the instance \( (G_\varphi, h') \) of \( \text{HomExt}(H) \).

**Claim 3.** If \( \varphi \) is a yes-instance of \( q\)-CSP-B, then there exists a homomorphism \( h : G_\varphi \to H \) that extends \( h' \).

**Proof of Claim.** If \( \varphi \) is a yes-instance of \( q\)-CSP-B, then there exists an assignment \( \gamma : X \to [B] \) satisfying each constraint. We define \( h : G_\varphi \to H \) separately on subgraphs of \( G_\varphi \), it is straightforward to verify that the union of these mappings is a well-defined homomorphism from \( G_\varphi \) to \( H \).

Fix \( j \in \{0, \ldots, L - 1\} \), and consider the or-gadget \( (F_j, h_j', R_j) \). Recall that the set \( P(c_{j'}) \) consists of all assignments of variables in \( X_{j'} \) that satisfy the constraint \( c_{j'} \). Therefore, there exists an assignment \( f_{k,j}^j \in P(c_{j'}) \) such that \( \gamma|_{X_{j'}} \equiv f_{k,j}^j \). Consider the vertex \( r_{k,j}' \in R_j \) that corresponds to that assignment. By the property (O2) of the or-gadget, we know that there exists a homomorphism \( h : F_j \to H \) that extends \( h' \), such that
Let $x_i \in X_{d'}$ and let $y = f_{d'}^{d'}(x_i) \in [B]$. Since for each $d \in \lambda(y)$ there exists a vertex $z_d \in V_{i,k}$ such that there is an $((a,d), w, w')$-implication-gadget from $r_k$ to $z_d$, the condition (1) implies that $h_1(V_{i,k}) = \lambda(y)$. We color the vertices of $V_{i,k}$ arbitrarily in a way that $h(V_{i,k}) = \lambda(y) \times \{w\}$.

Also, since for each $d \in S(\lambda(y))$ there exists a vertex $z_d \in U_{i,k}$ such that there is an $(a,d)$-implication-gadget from $r_k$ to $z_d$, the condition (1) implies that $h_1(U_{i,k}) = S(\lambda(y))$. We color the vertices of $U_{i,k}$ in a way that $h(U_{i,k}) = S(\lambda(y)) \times \{w'\}$. 

(1) $h_1(r_k) = a$,
(2) for every $r_k' \in R_j, k' \neq k$ we have that $h_1(r_k') \in \{b, c\}$.
Observe that by property (2), the implication gadgets from \( r_{k'} \) to the vertices of \( V_{i}^{j,k'} \cup U_{i}^{j,k'} \) do not put any constraints on the coloring of the sets \( V_{i}^{j,k'} \) and \( U_{i}^{j,k'} \). Therefore, for each \( v \in V_{i}^{j,k'} \) we set \( h(v) \) to be any vertex from \( S(\lambda(y)) \). Similarly, for each \( u \in U_{i}^{j,k'} \) we set \( h(u) \) to be any vertex from \( S(\lambda(y)) \).

Since \( h_{1}(r_{k'}) \in \{b,c\} \), and \( (b,z), (c,z) \in S_{a,d} \) for any \( z \in V(H_{i}) \) and \( d \in \lambda(y) \cup S(\lambda(y)) \), property (S2) applied to the implication gadgets asserts that we can always extend \( h \) to a homomorphism to the implication gadget to \( H \).

It remains to argue that the edges between the sets \( V_{i}^{j,k_{1}} \) and \( U_{i}^{j,k_{2}} \) are mapped to edges of \( H \), for any \( j_{1} < j_{2} \) and \( k_{1}, k_{2} \). However, observe that since \( y \) is an extension of some \( f_{k_{1}}^{j} \in S_{j_{1}} \) and \( f_{k_{2}}^{j} \in S_{j_{2}} \), we must have \( f_{k_{1}}^{j}(x_{i}) = f_{k_{2}}^{j}(x_{i}) = y \). Hence, \( h \) maps every \( v \in V_{i}^{j,k_{1}} \) to some element of \( \lambda(y) \times \{w\} \), and every \( u \in U_{i}^{j,k_{2}} \) to some element of \( S(\lambda(y)) \times \{w'\} \). By Observation 2, and since \( w' \in E(W) \), we get that \( h(v) \in E(H) \). That concludes the proof of the claim.

**Claim 4.** If there exists a homomorphism \( h : G_{\varphi} \to H \) that extends \( h' \), then \( \varphi \) is a yes-instance of \( q\text{-CSP-B} \).

**Proof of Claim.** We define the assignment \( \gamma : X \to [B] \) that makes every constraint from \( C \) satisfied.

Fix \( j \in \{0, \ldots, L - 1\} \), and consider the \( p_{j'} \)-or-gadget \( (F_{j}, h_{j'}^{j}, R_{j}) \). By the property (O1) of the or-gadget, there exists \( k_{j} \in [p_{j'}] \) such that \( h_{1}(r_{k_{j}}) = a \). Implication gadgets added in the step (2 whose) \( p \)-vertices were identified with \( r_{k_{j}} \) assert that \( h_{1}(V_{i}^{j,k_{j}}) \subseteq S(H) \) and \( h_{1}(U_{i}^{j,k_{j}}) \) is the signature of \( h_{1}(V_{i}^{j,k_{j}}) \). Then, by Observation 3, we have

\[
S(h_{1}(U_{i}^{j,k_{j}})) = S(S(h_{1}(V_{i}^{j,k_{j}}))) = h_{1}(V_{i}^{j,k_{j}}).
\]

Denote \( h_{1}(U_{i}^{j,k_{j}}) \) by \( T_{i}^{j} \), then \( h_{1}(V_{i}^{j,k_{j}}) = S(T_{i}^{j}) \). Let \( y_{i}^{j} = f_{k_{j}}^{j}(x_{i}) \) be the candidate assignment for \( x_{i} \in X_{j'} \) at index \( j \), recall that \( y_{i}^{j} = \lambda^{-1}(S(T_{i}^{j})) \).

Let \( i \in [n] \) be fixed and let \( j_{1}, j_{2} \in [L] \), \( j_{1} < j_{2} \) be such that \( x_{i} \in X_{j'}^{j_{1}} \cap X_{j'}^{j_{2}} \). Observe that in such case \( T_{i}^{j_{1}} \supseteq T_{i}^{j_{2}} \). Indeed, denote \( k_{1} = k_{j_{1}}, k_{2} = k_{j_{2}} \), and observe that

\[
\begin{align*}
(1) & \quad h_{1}(V_{i}^{j_{1},k_{1}}) = S(T_{i}^{j_{1}}) \quad \text{and} \quad h_{1}(U_{i}^{j_{1},k_{1}}) = T_{i}^{j_{1}}, \\
(2) & \quad h_{1}(V_{i}^{j_{2},k_{2}}) = S(T_{i}^{j_{2}}) \quad \text{and} \quad h_{1}(U_{i}^{j_{2},k_{2}}) = T_{i}^{j_{2}}.
\end{align*}
\]

Recall that each vertex from \( U_{i}^{j_{2},k_{2}} \) is adjacent to each vertex from \( V_{i}^{j_{1},k_{1}} \). Since \( h_{1} \) is a homomorphism, the same holds for their images: each vertex from \( T_{i}^{j_{1}} \) is adjacent to each vertex from \( S(T_{i}^{j_{1}}) \). Then \( S(T_{i}^{j_{1}}) \supseteq S(T_{i}^{j_{2}}) \), so \( T_{i}^{j_{1}} = S(S(T_{i}^{j_{1}})) \supseteq S(S(T_{i}^{j_{1}})) = T_{i}^{j_{2}} \).

We say that the index \( j_{1} \in \{0, \ldots, L - 1\} \) is problematic for \( i \) if there is \( j_{2} > j_{1} \) such that \( x_{i} \in X_{j'}^{j_{1}} \cap X_{j'}^{j_{2}} \) and \( T_{i}^{j_{1}} \not\supseteq T_{i}^{j_{2}} \). Since for each variable we have at most \( |H_{i}| \) problematic indices, there are at most \( |H_{i}| \cdot n \) problematic indices for all variables. Since \( L = m(|H_{i}| \cdot n + 1) \), by pigeonhole principle we get that there exists a set \( J \subseteq \{0, \ldots, L - 1\} \) of \( m \) consecutive indices such that none of them is problematic for any \( i \). For every \( i \in [n] \), we fix some \( j \in J \) such that \( x_{i} \in X_{j'} \) and set \( \gamma(x_{i}) = y_{i}^{j} \) (observe that the choice of \( j \) does not matter).

We claim that \( \gamma \) is an assignment that satisfies every constraint from \( \varphi \). Indeed, for any \( j' \in [m] \) there exists \( j \in J \) such that \( j' = j \mod m \). For every \( i \in X_{j'} \), we have \( \gamma(x_{i}) = y_{i}^{j} = f_{k_{j}}^{j}(x_{i}) \), so \( \gamma \) satisfies the constraint \( c_{j'} \).

Finally, it remains to adapt the arguments of Lampis [23] to establish the desired linear clique-width bound.
Claim 5. \( G_\varphi \) can be constructed in time polynomial in \(|\varphi|\), and we have \( \text{cw}(G_\varphi) \leq n + f(\epsilon, v) \) for some function \( f \), where \( v = |V(H)| \).

Proof of Claim. Observe that any \((S, w, w')\)-gadget constructed as in Lemma 6.4 for \( i = 1 \) has at most \(|V(H_i)|^{S-1} \cdot |V(H)| \leq v^{S-1} \) vertices. In particular, we can ensure that every implication gadget in \( G_\varphi \) has at most \( v^{O(\epsilon)} \) vertices. Moreover, we assume that all the or-gadgets of \( G_\varphi \) are constructed as in Lemma 6.5 and the subgadgets for \( S, S_{\text{left}} \) and \( S_{\text{right}} \) contain at most \( v^7 \) vertices. Then for every \( j \in \{0, \ldots, L - 1\} \), \( p_j'\)-or-gadget \( (F_j, h_j', R_j) \) has at most \((p_j' - 1) \cdot v^7 \) vertices.

For fixed \( H \) and \( \epsilon > 0 \) we have that \( B = s(H_i) \leq 2^{|V(H_i)|} - 2 \) and \( q \) is a constant that only depends on \( B, \epsilon \) (that is, on \(|V(H_i)|, \epsilon|\)). Each constraint of the \( q\)-CSP-B instance has at most \( B^q \) satisfying assignments. In particular, the number of vertices in each or-gadget is upper-bounded by \( B^q \cdot v^7 \). Therefore, it is not hard to see that the whole construction can be performed in polynomial time, if \( H \) is fixed and \( \epsilon \) is a constant.

For clique-width we use the following labels:

1. \( n \) main labels, representing the variables of \( \varphi \).
2. A single done label. Its informal meaning is that a vertex that receives this label will not be connected to anything else not yet introduced in the graph.
3. \( B^q \cdot v^7 \) constraint work labels.
4. \( qB^q \cdot v^{O(\epsilon)} \) variable-constraint incidence work labels.

To give a clique-width expression we describe how to build the graph, following essentially the steps given in the description of the construction by maintaining the following invariant: before starting iteration \( j \), all vertices of the set \( W'_i = \bigcup_{l<j} \bigcup_{k \in [p_j']} V_{i,k}^{l,j} \) have label \( i \), and all other vertices have the done label. This invariant is vacuously satisfied before the first iteration, since the graph is empty. Suppose that for some \( j \in \{0, \ldots, L - 1\} \) the invariant is true. We use the \( B^q \cdot v^7 \) constraint work labels to introduce the vertices of the \( p_j'\)-or-gadget \( (F_j, h_j', R_j) \), giving each vertex a distinct label. We use join operations to construct the internal edges of the or-gadget. Then, for each variable \( x_l \) that appears in the current constraint we do the following: we use \( B^q \cdot v^{O(\epsilon)} \) of the variable-constraint incidence work labels to introduce for all \( k \in [p_j'] \) the vertices of \( V_{i,k}^{l,j} \) and \( U_{i,k}^{l,j} \) as well as the implication gadgets connecting these to \( r_k^{l,j} \). Again we use a distinct label for each vertex, but the number of vertices (including internal vertices of the implication gadgets) is \( B^q \cdot v^{O(\epsilon)} \), so we have sufficiently many labels to use distinct labels for each of the \( q \) variables of the constraint. We use join operations to add the edges inside all implication gadgets. Then we use join operations to connect \( U_{i,k}^{l,j} \) to all vertices \( W_{i,k}^{l,j} \). This is possible, since the invariant states that all the vertices of \( W_{i,k}^{l,j} \) have the same label \( i \). We then rename all the vertices of \( U_{i,k}^{l,j} \) for all \( k \) to the done label, and do the same also for internal vertices of all implication gadgets. We proceed to the next variable of the same constraint and handle it using its own \( B^q \cdot v^{O(\epsilon)} \) labels. Once we have handled all variables of the current constraint, we rename all vertices of each \( V_{i,k}^{l,j} \) to label \( i \) for all \( k \). We then rename all vertices of the \( p_j'\)-or-gadget \( (F_j, h_j', R_j) \) gadget to the done label and increase \( j \) by 1. It is not hard to see that we have maintained the invariant and constructed all edges induced by the vertices introduced in steps up to \( j \), so repeating this process constructs the graph.

Together the claims imply Theorem 6.2 in the following way: For an arbitrary instance of \( q\)-CSP-B, our construction produces an instance of \( \text{HomExt}(H) \), and the instances are equivalent by Claim 3 and Claim 4. If one could solve \( \text{HomExt}(H) \) in \( O^*(\text{(|s(H)| - } \epsilon)^{\text{cw}(G)}) \) for some \( \epsilon > 0 \), one could use our construction to solve \( q\)-CSP-B, and by our choice of \( B \) and Claim 5 this procedure would have complexity \( O^*((B - \epsilon)^{n+\epsilon}) \) for some constant \( c \). By our choice of \( q \) according to Theorem 6.6, this contradicts the SETH.

ACM Transactions on Algorithms, Vol. 20, No. 3, Article 19. Publication date: May 2024.
7 SUMMARY AND CONCLUDING REMARKS

7.1 Extensions and Corollaries

We observe that Corollary 5.4 can be combined with Theorem 2.2 to obtain the following statement, which summarizes our results.

**Theorem 7.1.** Let \( H' \) be a fixed graph with the non-trivial connected core \( H \). Let \( H_1 \times \ldots \times H_m \) be the factorization of \( > H \). Let \( i \in [m] \) be such that \( s(H_i) = \max_{j \in [m]} s(H_j) \). Let \( G \) be an instance of \( \text{Hom}(H') \).

1. Assuming a clique-width expression \( \sigma \) of \( G \) of width \( cw(G) \) is given, the \( \text{Hom}(H') \) problem can be solved in time \( O^*(\max_{i \in [m]} s(H_i)^{cw(G)}) \).
2. Assuming SETH, if \( H \) is \( H_1 \)-projective, then there is no algorithm to solve \( \text{Hom}(H') \) in time \( O^*((\max_{i \in [m]} s(H_i) - \epsilon)^{cw(G)}) \) for any \( \epsilon > 0 \).

We note that the restriction to connected targets can be avoided by known properties of homomorphisms to disconnected graphs [29]; on the algorithmic side, one branches over all connected components of \( H \), while for the lower bound one considers the component with maximum signature number.

It is clear that obtaining a full complexity classification with respect to clique-width may require weakening the assumption in the second statement of Theorem 7.1. We recall that an analogous situation occurs in the work of Okrasa and Rzążewski [29]; as mentioned in the introduction section, the authors obtain the SETH-conditioned tight complexity bound for the \( \text{Hom}(H) \) problem parameterized by treewidth for all targets \( H \), assuming two conjectures of Larose and Tardif [24, 25]. The notion of \( H \)-projectivity allows us to restate these conjectures as one, which is not only sufficient in our setting but is also weaker in the sense of it being implied by the former two conjectures, but not necessarily equivalent to them.

**Conjecture 1.** Let \( H \) be a non-trivial core with prime factorization \( H_1 \times \ldots \times H_m \) and let \( i \in [m] \). Then \( H \) is \( H_i \)-projective.

Using Conjecture 1, we can restate our main result as follows.

**Theorem 7.2.** Let \( H' \) be a fixed graph with the non-trivial connected core \( H \). Let \( H_1 \times \ldots \times H_m \) be the prime factorization of \( H \). Let \( G \) be an instance of \( \text{Hom}(H') \).

1. Assuming the clique-width expression \( \sigma \) of \( G \) of width \( cw(G) \) is given, the \( \text{Hom}(H') \) problem can be solved in time \( O^*(\max_{i \in [m]} s(H_i)^{cw(G)}) \).
2. Assuming that Conjecture 1 and SETH hold, there is no algorithm to solve \( \text{Hom}(H') \) in time \( O^*((\max_{i \in [m]} s(H_i) - \epsilon)^{cw(G)}) \) for any \( \epsilon > 0 \).

We also observe that since each non-trivial projective core \( H \) is \( H \)-projective, in this case we already obtain a tight complexity bound.

**Corollary 7.3.** Let \( H' \) be a fixed graph with the non-trivial connected projective core \( H \). Let \( G \) be an instance of \( \text{Hom}(H') \).

1. Assuming the clique-width expression \( \sigma \) of \( G \) of width \( cw(G) \) is given, the \( \text{Hom}(H') \) problem can be solved in time \( O^*(s(H)^{cw(G)}) \).
2. There is no algorithm to solve \( \text{Hom}(H') \) in time \( O^*((s(H) - \epsilon)^{cw(G)}) \) for any \( \epsilon > 0 \), unless the SETH fails.

7.2 Generalizations and Other Research Directions

We remark that our hardness reduction is via \( \text{HomExt}(H) \), and in fact our algorithm can also easily be adapted to this setting (by removing all records that do not adhere to the partial mapping from
the input graph to \( H \) without an increase in complexity. However, since the dichotomy between \( P \) and \( \text{NP} \)-complete cases of \( \text{HomExt}(H) \) is more complicated (see [13], studied as the graph-retract problem) there exist target graphs \( H \) that are not covered by Theorem 7.2. On a similar note, let us also point out that setting up the \( \text{SETH} \)-conditioned tight complexity bounds for clique-width for a more general list problem \( L\text{Hom}(H) \) [12, 31] is widely open.

Another direction that is very closely related to our results is to determine similarly tight complexity bounds for the rank-width \( \text{rw} \) of the input graph: rank-width [20, 30] is a graph parameter that is known to be asymptotically equivalent to clique-width and is in fact used as an approximation of clique-width that can be computed in fixed-parameter tractable time. Our results together with the known relationship between clique-width and rank-width imply an upper bound of \( O^*(s(H)^{2\text{rw}+1}) \) and a \( \text{SETH} \) lower bound of \( (s(H)−\varepsilon)^{\text{rw}} \) on the complexity of \( \text{Hom}(H) \) for projective \( H \) parameterized by the rank-width of the input.

ACKNOWLEDGMENT

Karolina Okrasa acknowledges parts of this work were performed while visiting TU Wien, Vienna, Austria.

REFERENCES

[1] Hans-Jürgen Bandelt, Martin Farber, and Pavol Hell. 1993. Absolute reflexive retracts and absolute bipartite retracts. Discret. Appl. Math. 44, 1–3 (1993), 9–20. DOI: https://doi.org/10.1016/0166-218X(93)90219-E
[2] Rémy Belmonte, Eun Jung Kim, Michael Lampis, Valia Mitsou, and Yota Otachi. 2020. Grundy distinguishes treewidth from pathwidth. In Proceedings of the 28th Annual European Symposium on Algorithms (ESA ’20) (September 7–9, 2020) (Virtual Conference) (LIPIcs, Vol. 173). Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders (Eds.), Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 14:1–14:19.
[3] Jan Böker. 2021. Graph similarity and homomorphism densities. In Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP ’21) (July 12–16, 2021) (Virtual Conference) (LIPIcs, Vol. 198). Nikhil Bansal, Emanuela Merelli, and James Worrell (Eds.), Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 32:1–32:17.
[4] Andrei A. Bulatov and Aminieh Dadsetan. 2020. Counting homomorphisms in plain exponential time. In Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP ’20) (July 8–11, 2020) (Virtual Conference) (LIPIcs, Vol. 168). Artur Czumaj, Amuj Dawar, and Emanuela Merelli (Eds.), Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 21:1–21:18.
[5] Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. 2000. Linear time solvable optimization problems on graphs of bounded clique-width. Theory Comput. Syst. 33, 2 (2000), 125–150.
[6] Bruno Courcelle and Stephan Olariu. 2000. Upper bounds to the clique width of graphs. Discrete Appl. Math. 101, 1–3 (2000), 77–114.
[7] Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socała. 2017. Tight lower bounds on graph embedding problems. J. ACM 64, 3 (2017), 1–22. DOI: https://doi.org/10.1145/3051094
[8] Reinhard Diestel. 2012. Graph Theory (4th. ed.). Graduate texts in mathematics, Vol. 173. Springer.
[9] Willibald Dörfler. 1974. Primfaktorzerlegung und automorphismen des kardinalproduktes von graphen. Glasnik Matematicki 9 (1974), 15–27.
[10] P. Dukes, H. Emerson, and G. MacGillivray. 1998. Undecidable generalized colouring problems. J. Comb. Math. Comb. Comput. 26 (1998), 97–112.
[11] László Egri, Dániel Marx, and Paweł Rzążewski. 2018. Finding list homomorphisms from bounded-treewidth graphs to reflexive graphs: A complete complexity characterization. In Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science (STACS ’18) (February 28–March 3, 2018), 27:1–27:15.
[12] Tomás Feder, Pavol Hell, and Jing Huang. 2003. Bi-arc graphs and the complexity of list homomorphisms. J. Graph Theory 42, 1 (2003), 61–80. DOI: https://doi.org/10.1002/jgt.10073
[13] Tomás Feder and Moshe Y. Vardi. 1998. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. SIAM J. Comput. 28, 1 (1998), 57–104. DOI: https://doi.org/10.1137/S0097539794266766
[14] Fedor V. Fomin, Pinar Heggernes, and Dieter Kratsch. 2007. Exact algorithms for graph homomorphisms. Theory Comput. Syst. 41, 2 (2007), 381–393. DOI: https://doi.org/10.1007/s00224-007-0070-x
[15] Martin Grohe. 2007. The complexity of homomorphism and constraint satisfaction problems seen from the other side. J. ACM 54, 1 (2007), 1–24.

[16] Richard H. Hammack, Willfried Imrich, and Sandi Klavzar. 2011. *Handbook of Product Graphs*. CRC Press.

[17] Pavol Hell and Jaroslav Nesetril. 2004. *Graphs and Homomorphisms*. Oxford University Press.

[18] Pavol Hell and Jaroslav Nesetril. 1990. On the complexity of $\mathcal{H}$-coloring. J. Comb. Theory, Ser. B 48, 1 (1990), 92–110. DOI: https://doi.org/10.1016/0095-8956(90)90132-J

[19] Pavol Hell and Jaroslav Nesetril. 1992. The core of a graph. Discret. Math. 109, 1–3 (1992), 117–126.

[20] Sang il Oum and Paul Seymour. 2006. Approximating clique-width and branch-width. J. Comb. Theory, Ser. B 96, 4 (2006), 514–528. DOI: https://doi.org/10.1016/j.jctb.2005.10.006

[21] Russell Impagliazzo and Ramamohan Paturi. 2001. On the complexity of $k$-SAT. J. Comput. Syst. Sci. 62, 2 (2001), 367–375.

[22] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. 2001. Which problems have strongly exponential complexity? J. Comput. Syst. Sci. 63, 4 (2001), 512–530. DOI: https://doi.org/10.1006/jcss.2001.1774

[23] Michael Lampis. 2020. Finer tight bounds for coloring on clique-width. SIAM J. Discret. Math. 34, 3 (2020), 1538–1558. DOI: https://doi.org/10.1137/19M1280326

[24] Benoit Larose. 2002. Families of strongly projective graphs. Discuss. Math. Graph Theory 22, 2 (2002), 271–292. DOI: https://doi.org/10.7151/dmgt.1175

[25] Benoit Larose and Claude Tardif. 2001. Strongly rigid graphs and projectivity. Multiple-Valued Log. 7 (2001), 339–361.

[26] Daniel Lokshatanov, Dániel Marx, and Saket Saurabh. 2011. Lower bounds based on the exponential time hypothesis. Bull. EATCS 105 (2011), 41–72.

[27] Jaroslav Nesetril and Xuding Zhu. 2004. On sparse graphs with given colorings and homomorphisms. J. Comb. Theory, Ser. B 90, 1 (2004), 161–172. DOI: https://doi.org/10.1016/j.jctb.2003.06.001

[28] Karolina Okrasa, Marta Piecyk, and Paweł Rzążewski. 2020. Full complexity classification of the list homomorphism problem for bounded-treewidth graphs. In *Proceedings of the 28th Annual European Symposium on Algorithms (ESA ’20)* (September 7–9, 2020), Pisa, Italy (Virtual Conference) (LIPIcs, Vol. 173). Grzegorz Herman, and Peter Sanders (Eds.), Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 49–58.

[29] Karolina Okrasa and Paweł Rzążewski. 2021. Fine-grained complexity of the graph homomorphism problem for bounded-treewidth graphs. SIAM J. Comput. 50, 2 (2021), 487–508.

[30] Sang-il Oum. 2005. Approximating rank-width and clique-width quickly. In *Graph-Theoretic Concepts in Computer Science*. Dieter Kratsch (Ed.), Springer, Berlin, 49–58.

[31] Marta Piecyk and Paweł Rzążewski. 2021. Fine-grained complexity of the list homomorphism problem: Feedback vertex set and cutwidth. In *Proceedings of the 38th International Symposium on Theoretical Aspects of Computer Science (STACS’21)* (March 16–19, 2021) (Virtual Conference) (LIPIcs, Vol. 187), Markus Bläser and Benjamin Monmege (Eds.), Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 56:1–56:17.

[32] Neil Robertson and Paul D. Seymour. 1986. Graph minors. II. Algorithmic aspects of tree-width. J. Algorithms 7, 3 (1986), 309–322.

[33] Paweł Rzążewski. 2014. Exact algorithm for graph homomorphism and locally injective graph homomorphism. Inf. Process. Lett. 114, 7 (2014), 387–391.

[34] Mark H. Siggers. 2010. A new proof of the $\mathcal{H}$-coloring dichotomy. SIAM J. Discret. Math. 23, 4 (2010), 2204–2210. DOI: https://doi.org/10.1137/080736697

[35] Tomasz Łuczak and Jaroslav Nešetril. 2004. Note on projective graphs. J. Graph Theory 47, 2 (2004), 81–86.

[36] Magnus Wahlström. 2011. New plain-exponential time classes for graph homomorphism. Theory Comput. Syst. 49, 2 (2011), 273–282. DOI: https://doi.org/10.1007/s00224-010-9261-z

Received 3 June 2022; revised 30 November 2023; accepted 25 February 2024