Conjunctive Queries on Probabilistic Graphs: The Limits of Approximability

Antoine Amarilli
LTCI, Télécom Paris, Institut Polytechnique de Paris, France

Timothy van Bremen
National University of Singapore, Singapore

Kuldeep S. Meel
University of Toronto, Canada

Abstract

Query evaluation over probabilistic databases is a notoriously intractable problem – not only in combined complexity, but for many natural queries in data complexity as well [7, 14]. This motivates the study of probabilistic query evaluation through the lens of approximation algorithms, and particularly of *combined FPRASes*, whose runtime is polynomial in both the query and instance size. In this paper, we focus on tuple-independent probabilistic databases over binary signatures, which can be equivalently viewed as *probabilistic graphs*. We study in which cases we can devise combined FPRASes for probabilistic query evaluation in this setting.

We settle the complexity of this problem for a variety of query and instance classes, by proving both approximability and (conditional) inapproximability results. This allows us to deduce many corollaries of possible independent interest. For example, we show how the results of [8] on counting fixed-length strings accepted by an NFA imply the existence of an FPRAS for the two-terminal network reliability problem on directed acyclic graphs: this was an open problem until now [37]. We also show that one cannot extend a recent result [34] that gives a combined FPRAS for self-join-free conjunctive queries of bounded hypertree width on probabilistic databases: neither the bounded-hypertree-width condition nor the self-join-freeness hypothesis can be relaxed. Finally, we complement all our inapproximability results with unconditional lower bounds, showing that DNNF provenance circuits must have at least moderately exponential size in combined complexity.

2012 ACM Subject Classification Theory of computation → Database query processing and optimization (theory)

Keywords and phrases Probabilistic query evaluation, tuple-independent databases, approximation

Digital Object Identifier 10.4230/LIPIcs.ICDT.2024.15

Funding This project was supported in part by the National Research Foundation Singapore under its NRF Fellowship programme [NRF-NRFFAI1-2019-0004] and Campus for Research Excellence and Technological Enterprise (CREATE) programme, as well as the Ministry of Education Singapore Tier 1 and 2 grants R-252-000-B59-114 and MOE-T2EP20121-0011. Amarilli was partially supported by the ANR project EQUUS ANR-19-CE48-0019, by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 431183758, and by the ANR project ANR-18-CE23-0003-02 ("CQFD"). This work was done in part while Amarilli was visiting the Simons Institute for the Theory of Computing.

Acknowledgements The authors thank Octave Gaspard for pointing out some oversights in some proofs, which are corrected in the present version.

1 Introduction

Tuple-independent probabilistic databases (TID) are a simple and principled formalism to model uncertainty and noise in relational data [13, 32]. In the TID model, each tuple of a relational database is annotated with an independent probability of existence; all tuples are assumed to be independent. In the *probabilistic query evaluation* (PQE) problem, given a Boolean query Q and a TID instance I, we must compute the probability that Q holds in a subinstance sampled from I according to the resulting distribution. The PQE problem has been studied in database theory both in terms of *combined complexity*, where the query and instance are part of the input, and in *data complexity*, where the query is fixed and only the instance is given as input [35]. Unfortunately, many of the results so far [32] show that the PQE problem is highly intractable, even in data complexity for many natural queries (e.g., a path query of length three), and hence also in combined complexity.

Faced with this intractability, a natural approach is to study *approximate PQE*: we relax the requirement of computing the exact probability that the query holds, and settle for an approximate answer. This approach has been studied in data complexity [32]: for any fixed union of conjunctive queries (UCQ), we can always tractably approximate the answer to PQE, additively (simply by Monte Carlo sampling), or multiplicatively (using the Karp-Luby approximation algorithm on a disjunctive-normal-form representation of the query provenance). However, these approaches are not tractable in combined complexity, and moreover the latter approach exhibits a "slicewise polynomial" runtime of the form $O(|I|^{|Q|})$ – rather than, say, $O(2^{|Q|}\mathsf{poly}(|I|))$ – which seriously limits its practical utility. Thus, our goal is to obtain a *combined FPRAS* for PQE: by this we mean a fully polynomial-time randomized approximation scheme, giving a multiplicative approximation of the probability, whose runtime is polynomial in the query and TID (and in the desired precision). This approach has been recently proposed by van Bremen and Meel [34], who show a combined FPRAS for CQs when assuming that the query is self-join-free and has bounded hypertree width; their work leaves open the question of which other cases admit combined FPRASes.

Main Results. In this paper, following the work of Amarilli, Monet and Senellart [7] for exact PQE, we investigate the combined complexity of *approximate* PQE in the setting of *probabilistic graphs*. In other words, we study *probabilistic graph homomorphism*, which is the equivalent analogue of CQ evaluation: given a (deterministic) query graph G, and given a instance graph H with edges annotated with independent probabilities (like a TID), we wish to approximate the probability that a randomly selected subgraph $H' \subseteq H$ admits a homomorphism from G. This setting is incomparable to that of [34], because it allows for self-joins and for queries of unbounded width, but assumes that relations are binary.

Of course, the graph homomorphism problem is intractable in combined complexity if the input graphs are arbitrary (even without probabilities). Hence, we study the problem when the query graph and instance graph are required to fall in restricted graph classes, chosen to ensure tractability in the non-probabilistic setting. We use similar classes as those from [7]: *path graphs* which may be *one-way* ($\mathsf{1WP}$: all edges are oriented from left to right) or *two-way* ($\mathsf{2WP}$: edge orientations are arbitrary); *tree graphs* which may be *downward* (DWT: all edges are oriented from the root to the leaves) or *polytrees* (PT: edge orientations are arbitrary); and, for the instance graph, *directed acyclic graphs* (DAG), or *arbitrary graphs* (All).

For all combinations of these classes, we show either (i) the existence of a combined FPRAS, or (ii) the non-existence of such an FPRAS, subject to standard complexity-theoretic assumptions. We summarize our results in Table 1, respectively for graphs that are *labelled* (i.e., the signature features several binary relations), or *unlabelled* (i.e., only one binary relation). We emphasize that the signature for labelled graphs is assumed to be fixed and does not form part of the input, consistent with prior work [7] (although identical results can likely be obtained even when dropping this assumption).

▶ **Result 1.1** (Sections 3 and 4). *The results in Table 1, described in terms of the graph classes outlined above, hold.*

In summary, for the classes that we consider, our results mostly show that the general intractability of combined PQE carries over to the approximate PQE problem. The important exception is Proposition 3.1: the PQE problem for one-way path queries on *directed acyclic graphs* (DAGs) admits a combined FPRAS. We discuss more in detail below how this result is proved and some of its consequences. Another case is left open: in the unlabelled setting, we do not settle the approximability of combined PQE for one-way path queries (or equivalently downward tree queries) on arbitrary graphs. For all other cases, either exact combined PQE was already shown to be tractable in the exact setting [7], or we strengthen the #P-hardness of exact PQE from [7] by showing that combined FPRASes conditionally do not exist. We stress that our results always concern *multiplicative approximations*: as non-probabilistic graph homomorphism is tractable for the classes that we consider, we can always obtain additive approximations for PQE simply by Monte Carlo sampling. Further note that our intractability results are always shown in *combined complexity* – in data complexity, for the queries that we consider, PQE is always multiplicatively approximable via the Karp-Luby algorithm [32].

As an important consequence, our techniques yield connections between approximate PQE and *intensional* approaches to the PQE problem. Recall that the intensional approach was introduced by Jha and Suciu [21] in the setting of exact evaluation, and when measuring data complexity. They show that many tractable queries for PQE also admit tractable provenance representations. More precisely, for these queries Q, there is a polynomial-time algorithm that takes as input any database instance and computes a representation of the Boolean provenance of Q in a form which admits tractable model counting (e.g., OBDD, d-DNNF, etc.). This intensional approach contrasts with *extensional* approaches (like [14]) which exploit the structure of the query directly: comparing both approaches is still open [27].

In line with this intensional approach, we complement our conditional hardness results on approximate PQE with *unconditional* lower bounds on the *combined* size of tractable representations of query provenance. Namely, we show a moderately exponential lower bound on DNNF provenance representations for all our non-approximable query-instance class pairs:

▶ **Result 1.2** (Section 5, informal). *Let $\langle \mathcal{G}, \mathcal{H} \rangle$ be a conditionally non-approximable query-instance class pair studied in this paper. For any $\epsilon > 0$, there is an infinite family $G_1, G_2, \ldots$ of $\mathcal{G}$ queries and an infinite family $H_1, H_2, \ldots$ of $\mathcal{H}$ instances such that, for any $i > 0$, any DNNF circuit representing the provenance $\mathsf{Prov}_{H_i}^{G_i}$ has size at least $2^{\Omega\left((||G_i||+||H_i||)^{1-\epsilon}\right)}$.*

The class of DNNF circuits is arguably the most succinct circuit class in knowledge compilation that still has desirable properties [15, 16]. Such circuits subsume in particular the class of *structured DNNFs*, for which tractable approximation algorithms were recently proposed [9]. Thus, these bounds help to better understand the limitations of intensional approaches.

Consequences. Our results and techniques have several interesting consequences of potential independent interest. First, they imply that we cannot relax the hypotheses of the result of van Bremen and Meel mentioned earlier [34]. They show the following result on combined FPRASes for PQE in the more general context of probabilistic databases:

▶ **Theorem 1.3** (Theorem 1 of [34]). *Let Q be a self-join-free conjunctive query of bounded hypertree width, and H a tuple-independent database instance. Then there exists a combined FPRAS for computing the probability of Q on H, i.e., an FPRAS whose runtime is $\mathsf{poly}(|Q|, ||H||, \epsilon^{-1})$, where ϵ is the multiplicative error.*

It was left open in [34] whether intractability held without these assumptions on the query. Hardness is immediate if we do not bound the width of queries and allow arbitrary self-join-free CQs, as combined query evaluation is then NP-hard already in the non-probabilistic setting. However, it is less clear whether the self-join-freeness condition can be lifted. Our results give a negative answer, already in a severely restricted setting:

▶ **Result 1.4** (Corollaries 6.1 and 6.2). *Assuming* RP $\neq$ NP*, neither the bounded hypertree width nor self-join-free condition in Theorem 1.3 can be relaxed: even on a fixed signature consisting of a single binary relation, there is no FPRAS to approximate the probability of an input treewidth-1 CQ on an input treewidth-1 TID instance.*

A second consequence implied by our techniques concerns the *two-terminal network reliability problem* on directed acyclic graphs (DAGs). Roughly speaking, given a directed graph $G = (V, E)$ with independent edge reliability probabilities $\pi : E \to [0, 1]$, and two distinguished vertices $s, t \in V$, the two-terminal network reliability problem asks for the probability that there is a path from s to t. The problem is known to be #P-hard even on DAGs [29, Table 2]. The existence of an FPRAS for the two-terminal network reliability problem is a long-standing open question [22], and the case of DAGs was explicitly left open by Zenklusen and Laumanns [37]. Our results allow us to answer in the affirmative:

▶ **Result 1.5** (Theorem 6.3). *There exists an FPRAS for the two-terminal network reliability problem over DAGs.*

This result and our approximability results follow from the observation that path queries on directed acyclic graphs admit a compact representation of their Boolean provenance as *non-deterministic ordered binary decision diagrams* (nOBDDs). We are then able to use a recent result by Arenas et al. [8, Corollary 4.5] giving an FPRAS for counting the satisfying assignments of an nOBDD, adapted to the weighted setting.

Paper Structure. In Section 2, we review some of the technical background. We then present our main results on approximability, divided into the labelled and unlabelled case, in Sections 3 and 4 respectively. Next, in Section 5, we show lower bounds on DNNF provenance circuit sizes. In Section 6, we show some consequences for previous work [34], as well as for the two-terminal network reliability problem. We conclude in Section 7.

2 Preliminaries

We provide some technical background below, much of which closes follows that in [4] and [7].

Graphs and Graph Homomorphisms. Let σ be an non-empty finite set of labels. When $|\sigma| > 1$, we say that we are in the *labelled setting*, and when $|\sigma| = 1$, the *unlabelled setting.* In this paper, we study only *directed* graphs with edge labels from σ. A graph G over σ is a tuple (V, E, λ) with finite non-empty vertex set V, edge set $E \subseteq V^2$, and $\lambda\colon E \to \sigma$ a labelling function mapping each edge to a single label (we may omit λ in the unlabelled setting). The *size* $||G||$ of G is its number of edges. We write $x \xrightarrow{R} y$ for an edge $e = (x, y) \in E$ with label $\lambda(e) = R$, and $x \to y$ for $(x, y) \in E$ (no matter the edge label). We sometimes use a simple regular-expression-like syntax (omitting the vertex names) to represent path graphs: for example, we write $\to\to$ to represent an unlabelled path of length two, and the notation $\to^k$ to denote an unlabelled path of length k. All of this syntax extends to labelled graphs in the obvious way. A graph $H = (V', E', \lambda')$ is a *subgraph* of G, written $H \subseteq G$, if $V = V'$, $E' \subseteq E$, and λ' is the restriction of λ to E'.

A *graph homomorphism* h from a graph $G = (V_G, E_G, \lambda_G)$ to a graph $H = (V_H, E_H, \lambda_H)$ is a function $h : V_G \to V_H$ such that, for all $(u, v) \in E_G$, we have $(h(u), h(v)) \in E_H$ and $\lambda_H((h(u), h(v)) = \lambda_G((u, v))$. We write $G \rightsquigarrow H$ to say that such a homomorphism exists.

Probabilistic Graphs and Probabilistic Graph Homomorphism. A *probabilistic graph* is a pair (H, π), where H is a graph with edge labels from σ, and $\pi : E \to [0, 1]$ is a probability labelling on the edges. Note that edges e in H are annotated both by their probability value $\pi(e)$ and their σ-label $\lambda(e)$. Intuitively, π gives us a succinct specification of a probability distribution over the $2^{||H||}$ possible subgraphs of H, by independently including each edge e with probability $\pi(e)$. Formally, the distribution induced by π on the subgraphs $H' \subseteq H$ is defined by $\Pr_\pi(H') = \prod_{e\in E'} \pi(e) \prod_{e\in E\setminus E'}(1 - \pi(e))$.

In this paper, we study the *probabilistic graph homomorphism* problem PHom for a fixed set of labels σ: given a graph G called the *query graph* and a probabilistic graph (H, π) called the *instance graph*, both using labels from σ, we must compute the probability $\Pr_\pi(G \rightsquigarrow H)$ that a subgraph of H, sampled according to the distribution induced by π, admits a homomorphism from G. That is, we must compute $\Pr_\pi(G \rightsquigarrow H) := \sum_{H'\subseteq H \text{ s.t. } G\rightsquigarrow H'} \Pr_\pi(H')$.

We study PHom in *combined complexity*, i.e., when both the query graph G and instance graph (H, π) are given as input. Further, we study PHom when we restrict G and H to be taken from specific *graph classes*, i.e., infinite families of (non-probabilistic) graphs, denoted respectively $\mathcal{G}$ and $\mathcal{H}$. (Note that $\mathcal{H}$ does not restrict the probability labelling π.) To distinguish the *labelled* and *unlabelled* setting, we denote by $\mathsf{PHom}_{\mathsf{L}}(\mathcal{G}, \mathcal{H})$ the problem of computing $\Pr_\pi(G \rightsquigarrow H)$ for $G \in \mathcal{G}$ and (H, π) with $H \in \mathcal{H}$ when the fixed set of allowed labels in $\mathcal{G}$ and $\mathcal{H}$ has cardinality $|\sigma| > 1$, and likewise write $\mathsf{PHom}_{\not\mathsf{L}}(\mathcal{G}, \mathcal{H})$ when $\mathcal{G}$ and $\mathcal{H}$ are classes of unlabelled graphs. We focus on *approximation algorithms*: fixing classes $\mathcal{G}$ and $\mathcal{H}$, a *fully polynomial-time randomized approximation scheme* (FPRAS) for $\mathsf{PHom}_{\mathsf{L}}(\mathcal{G}, \mathcal{H})$ (in the labelled setting) or $\mathsf{PHom}_{\not\mathsf{L}}(\mathcal{G}, \mathcal{H})$ (in the unlabelled setting) is a randomized algorithm that runs in time $\mathsf{poly}(||G||, ||H||, \epsilon^{-1})$ on inputs $G \in \mathcal{G}$, (H, π) for $H \in \mathcal{H}$, and $\epsilon > 0$. The algorithm must return, with probability at least 3/4, a *multiplicative approximation* of the probability $\Pr_\pi(G \rightsquigarrow H)$, i.e., a value between $(1 - \epsilon) \Pr_\pi(G \rightsquigarrow H)$ and $(1 + \epsilon) \Pr_\pi(G \rightsquigarrow H)$.

Graph Classes. We study PHom on the following graph classes, which are defined on a graph G with edge labels from σ, and are either labelled or unlabelled depending on σ:

- G is a *one-way path* ($\mathsf{1WP}$) if it is of the form $a_1 \xrightarrow{R_1} \dots \xrightarrow{R_{m-1}} a_m$ for some m, with all $a_1, \dots, a_m$ being pairwise distinct, and with $R_i \in \sigma$ for $1 \leq i < m$.
- G is a *two-way path* ($\mathsf{2WP}$) if it is of the form $a_1 - \dots - a_m$ for some m, with pairwise distinct $a_1, \dots, a_m$, and each $-$ being $\xrightarrow{R_i}$ or $\xleftarrow{R_i}$ (but not both) for some label $R_i \in \sigma$.
- G is a *downward tree* (DWT) if it is a rooted unranked tree (each node can have an arbitrary number of children), with all edges pointing from parent to child in the tree.
- G is a *polytree* (PT) if its underlying undirected graph is a rooted unranked tree, without restrictions on the edge directions.
- G is a *DAG* (DAG) if it is a (directed) acyclic graph.

These refine the classes of connected queries considered in [7], by adding the DAG class. We denote by All the class of all graphs. Note that both $\mathsf{2WP}$ and DWT generalize $\mathsf{1WP}$ and are incomparable; PT generalizes both $\mathsf{2WP}$ and DWT; DAG generalizes PT; All generalizes DAG (see Figure 2 of [7]).

Boolean Provenance. We use the notion of *Boolean provenance*, or simply *provenance* [20, 3, 30]. In the context of databases, provenance intuitively represents which subsets of the instance satisfy the query: it is used in the intensional approach to probabilistic query evaluation [21]. In this paper, we use provenance to show both upper and lower bounds.

Formally, let $G = (V_G, E_G, \lambda_G)$ and $H = (V_H, E_H, \lambda_H)$ be graphs. Seeing E_H as a set of Boolean variables, a *valuation* ν of E_H is a function $\nu\colon E_H \to \{0,1\}$ that maps each edge of H to 0 or 1. Such a valuation ν defines a subgraph H_ν of H where we only keep the edges mapped to 1, formally $H_\nu = (V_H, \{e \in E_H \mid \nu(e) = 1\}, \lambda_H)$. The *provenance* of G on H is then the Boolean function Prov_H^G having as variables the edges E_H of H and mapping every valuation ν of E_H to 1 (true) or 0 (false) depending on whether $G \rightsquigarrow H_\nu$ or not. Generalizing this definition, for any integer n, for any choice of $a_1, \ldots, a_n \in V_G$ and $b_1, \ldots, b_n \in V_H$, we write $\mathsf{Prov}_H^G[a_1 := b_1, \ldots, a_n := b_n]$ to denote the Boolean function that maps valuations ν of E_H to 1 or 0 depending on whether or not there is a homomorphism $h\colon G \to H_\nu$ which additionally satisfies $h(a_i) = b_i$ for all $1 \leq i \leq n$.

For our lower bounds, we will often seek to represent Boolean formulas as the provenance of queries on graphs:

▶ **Definition 2.1.** *Given two graphs G and H, and a Boolean formula ϕ whose variables $\{e_1, \ldots, e_n\} \subseteq E_H$ are edges of H, we say that Prov_H^G represents ϕ on $(e_1, ..., e_n)$ if for every valuation $\nu : E_H \to \{0,1\}$ that maps edges not in $\{e_1, ..., e_n\}$ to 1, we have $\nu \models \phi$ if and only if $\mathsf{Prov}_H^G(\nu) = 1$.*

Circuits and Knowledge Compilation. We consider representations of Boolean functions in terms of *non-deterministic (ordered) binary decision diagrams*, as well as *decomposable circuits*, which we define below.

A *non-deterministic binary decision diagram* (nBDD) on a set of variables $V = \{v_1, \ldots, v_n\}$ is a rooted DAG D whose nodes carry a label in $V \sqcup \{0, 1, \lor\}$ and whose edges can carry an optional label in $\{0,1\}$, subject to the following requirements:

1. there are exactly two leaves (called *sinks*), one labelled by 1 (the 1-*sink*), and the other by 0 (the 0-*sink*);
2. internal nodes are labelled either by $\lor$ (called an $\lor$-*node*) or by a variable of V (called a *decision node*); and
3. each decision node has exactly two outgoing edges, labelled 0 and 1; the outgoing edges of $\lor$-nodes are unlabelled.

The size $||D||$ of D is its number of edges. Let ν be a valuation of V, and let π be a path in D going from the root to one of the sinks. We say that π is *compatible* with ν if for every decision node n of the path, letting $v \in V$ be the variable labelling n, then π passes through the outgoing edge of n labelled with $\nu(v)$. In particular, no constraints are imposed at $\lor$-nodes; thus, we may have that multiple paths are compatible with a single valuation. The nBDD D *represents* a Boolean function, also written D by abuse of notation, which is defined as follows: for each valuation ν of V, we set $D(\nu) := 1$ if there exists a path π from the root to the 1-sink of D that is compatible with ν, and set $D(\nu) := 0$ otherwise. Given an nBDD D over variables V, we denote by $\mathsf{Mods}(D)$ the set of satisfying valuations ν of D such that $D(\nu) = 1$, and by $\mathsf{MC}(D)$ the number $|\mathsf{Mods}(D)|$ of such valuations. Further, given a rational probability function $w : V \to [0,1]$ on the variables of V, define $\mathsf{WMC}(D, w)$ to be the probability that a random valuation ν satisfies F, that is, $\mathsf{WMC}(D, w) = \sum_{\nu \in \mathsf{Mods}(D)} \prod_{x \in V \text{ s.t. } \nu(x)=1} w(x) \prod_{x \in V \text{ s.t. } \nu(x)=0} (1 - w(x))$.

In this paper, we primarily focus on a subclass of nBDDs called *non-deterministic ordered binary decision diagrams* (nOBDDs). An nOBDD D is an nBDD for which there exists a strict total order $\prec$ on the variables V such that, for any two decision nodes $n \neq n'$ such that there is a path from n to n', then, letting v and v' be the variables that respectively label n and n', we have $v \prec v'$. This implies that, along any path going from the root to a sink, the sequence of variables will be ordered according to V, with each variable occurring at most once. We use nOBDDs because they admit tractable approximate counting of their satisfying assignments, as we discuss later.

We also show lower bounds on a class of *circuits*, called *decomposable negation normal form* (DNNF) circuits. A *circuit* on a set of variables V is a directed acyclic graph $C = (G, W)$, where G is a set of *gates*, where $W \subseteq G \times G$ is a set of edges called *wires*, and where we distinguish an *output gate* $g_0 \in G$. The *inputs* of a gate $g \in G$ are the gates g' such that there is a wire (g', g) in W. The gates can be labelled with variables of V (called a *variable gate*), or with the Boolean operators $\vee$, $\wedge$, and $\neg$. We require that gates labelled with variables have no inputs, and that gates labelled with $\neg$ have exactly one input. A circuit C defines a Boolean function on V, also written C by abuse of notation. Formally, given a valuation ν of V, we define inductively the *evaluation* ν' of the gates of C by setting $\nu'(g) := \nu(v)$ for a variable-gate g labelled with variable v, and setting $\nu'(g)$ for other gates to be the result of applying the Boolean operators of g to $\nu'(g_1), \ldots, \nu'(g_n)$ for the inputs $g_1, \ldots, g_n$ of g. We then define $C(\nu)$ to be $\nu'(g_0)$ where g_0 is the output gate of C.

The circuit is in *negation normal form* if negations are only applied to variables, i.e., for every $\neg$-gate, its input is a variable gate. The circuit is *decomposable* if the $\wedge$-gates always apply to inputs that depend on disjoint variables: formally, there is no $\wedge$-gate g with two distinct inputs g_1 and g_2, such that some variable v labels two variable gates g'_1 and g'_2 with g'_1 having a directed path to g_1 and g'_2 having a directed path to g_2. A *DNNF* is a circuit which is both decomposable and in negation normal form. Note that we can translate nOBDDs in linear time to DNNFs, more specifically to *structured DNNFs* [4, Proposition 3.8].

Approximate Weighted Counting for nOBDDs. Recently, Arenas et al. [9] showed the following result on approximate counting of satisfying assignments of an nOBDD.

▶ **Theorem 2.2** (Corollary 4.5 of [8])**.** *Let D be an nOBDD. Then there exists an FPRAS for computing* $\mathsf{MC}(D)$*.*

For our upper bounds, we need a slight strengthening of this result to apply to *weighted model counting* (WMC) in order to handle probabilities. This can be achieved by translating the approach used in [34, Section 5.1] to the nOBDD setting. We thus show (see Appendix A):

▶ **Theorem 2.3.** *Let D be an nOBDD, and $w : \mathsf{vars}(D) \to [0, 1]$ be a rational probability function defined on the variables appearing in D. Then there exists an FPRAS for computing* $\mathsf{WMC}(D, w)$*, running in time polynomial in $||D||$ and w.*

3 Results in the Labelled Setting

We now move on to the presentation of our results. We start with the *labelled* setting of probabilistic graph homomorphism in which the fixed signature σ of the query and instance graph contains more than one label ($|\sigma| > 1$). Our results are summarized in Table 1a.

Table 1 Results on approximation proved in this paper. Key: white ( ) means that the problem lies in P; light grey (▨) means that it is #P-hard but admits an FPRAS; dark grey (■) means #P-hardness and non-existence of an FPRAS, assuming RP ≠ NP. All cells without a reference to a corresponding proposition are either implied by one of the other results in this paper, or pertain to exact complexity and were already settled in [7].

(a) Complexity of $\mathsf{PHom_L}(\mathcal{G}, \mathcal{H})$.

$\mathcal{G} \downarrow$	$\mathcal{H} \rightarrow$					
	1WP	2WP	DWT	PT	DAG	All
1WP				▨	▨ 3.1	■ 3.3
2WP			■ 3.7	■	■	■
DWT			■ 3.5	■	■	■
PT			■	■	■	■

(b) Complexity of $\mathsf{PHom_{L\!\!/}}(\mathcal{G}, \mathcal{H})$.

$\mathcal{G} \downarrow$	$\mathcal{H} \rightarrow$					
	1WP	2WP	DWT	PT	DAG	All
1WP					▨ 4.1	?
2WP				■ 4.3	■	■
DWT					▨ 4.2	?
PT				■	■	■

1WP on DAG. We start by showing the tractability of approximation for $\mathsf{PHom_L}(\mathsf{1WP}, \mathsf{DAG})$, which also implies tractability of approximation for $\mathsf{PHom_L}(\mathsf{1WP}, \mathsf{PT})$, since $\mathsf{PT} \subseteq \mathsf{DAG}$.

▶ **Proposition 3.1.** $\mathsf{PHom_L}(\mathsf{1WP}, \mathsf{DAG})$ *is* #P*-hard already in data complexity, but it admits an FPRAS.*

For #P-hardness, the result already holds in the unlabelled setting, so it will be shown in Section 4 (see Proposition 4.1). Hence, we focus on the upper bound. We rely on the notion of a *topological ordering* of the edges of a directed acyclic graph $H = (V, E)$: it is simply a strict total order $(E, \prec)$ with the property that for every consecutive pair of edges $e_1 = (a_1, a_2)$ and $e_2 = (a_2, a_3)$, we have that $e_1 \prec e_2$. Let us fix such an ordering.

Proof of Proposition 3.1. We will show that every 1WP query on a DAG instance admits an nOBDD representation of its provenance, which we can compute in combined polynomial time. We can then apply Theorem 2.3, from which the result follows. Let $G = a_1 \xrightarrow{R_1} \dots \xrightarrow{R_m} a_{m+1}$ be the input path query, and H the instance graph. We make the following claim:

▷ Claim 3.2. For every $v \in H$, we can compute in time $O(||G|| \times ||H||)$ an nOBDD representing $\mathsf{Prov_L}{}_H^G[a_1 := v]$ which is ordered by the topological ordering $\prec$ fixed above.

Proof. Writing $H = (V, E)$, we build an nBDD D consisting of the two sinks and of the following nodes:

- $|V| \times ||G||$ $\vee$-nodes written $n_{u,i}$ for $u \in V$ and $1 \leq i \leq m$; and
- $|E| \times ||G||$ decision nodes written $d_{e,i}$ for $e \in E$ and $1 \leq i \leq m$ which test the edge e.

Each $\vee$-node $n_{u,i}$ for $u \in V$ and $1 \leq i \leq m$ has outgoing edges to each $d_{e,i}$ for every edge e emanating from u which is labelled R_i. For each decision node $d_{e,i}$, letting w be the target of edge e, then $d_{e,i}$ has an outgoing 0-edge to the 0-sink and an outgoing 1-edge to either $n_{w,i+1}$ if $i < m$ or to the 1-sink if $i = m$. The root of the nBDD is the node $n_{v,1}$.

This construction clearly respects the time bound. To check correctness of the resulting nBDD, it is immediate to observe that, for any path from the root to a sink, the sequence of decision nodes traversed is of the form $d_{e_1,1}, \dots, d_{e_k,k}$ where the $e_1, \dots, e_k$ form a path of consecutive edges starting at v and successively labelled $R_1, \dots, R_k$. This implies that the nBDD is in fact an nOBDD ordered by $\prec$. Further, such a path reaches the 1-sink iff $k = m$ and all decisions are positive, which implies that whenever the nOBDD accepts a subgraph H' of H then indeed H' contains a match of G mapping a_1 to v. For the converse direction, we observe that, for any subgraph H' of H containing a match of G mapping a_1 to v, then, letting $e_1, \ldots, e_m$ be the successive edges traversed in the match of G, there is a path from the root of D to the 1-sink which tests these edges in order. This establishes correctness and concludes the proof of the claim. ◁

Now observe that $\mathsf{Prov_L}{}_H^G = \mathsf{Prov_L}{}_H^G[a_1 := v_1] \vee \cdots \vee \mathsf{Prov_L}{}_H^G[a_1 := v_n]$, where $v_1, \ldots, v_n$ are precisely the vertices of H. Thus, it suffices to simply take the disjunction of each nOBDD obtained using the process above across every vertex in H, which yields in linear time the desired nOBDD. From here we can apply Theorem 2.3, concluding the proof. ◀

1WP on arbitrary graphs. We show, however, that tractability of approximation does *not* continue to hold when relaxing the instance class from DAG to arbitrary graphs. This also implies that more expressive classes of query graphs – such as 2WP, DWT, and PT also cannot be tractable to approximate on All instances.

▶ **Proposition 3.3.** $\mathsf{PHom_L}(\mathsf{1WP}, \mathsf{All})$ *does not admit an FPRAS unless* $\mathsf{RP} = \mathsf{NP}$.

Proof. Our result hinges on the following claim:

▷ Claim 3.4. Let $d > 1$ be a constant. Given a monotone 2-CNF formula ϕ on n variables where each variable occurs in at most d clauses, we can build in time $O(|\phi|)$ a 1WP G_ϕ and All graph H_ϕ containing edges $(e_1, \ldots, e_n)$ such that $\mathsf{Prov}_{H_\phi}^{G_\phi}$ represents ϕ on $(e_1, \ldots, e_n)$.

Proof. Let $\phi = \bigwedge_{1 \le i \le m} (X_{f_1(i)} \vee X_{f_2(i)})$ be the input CNF instance over the variables $\{X_1, \ldots, X_n\}$. As we are in the labelled setting, let U and R be two distinct labels from the signature. Define the 1WP query graph G_ϕ to be $\xrightarrow{U} \left(\xrightarrow{R}{}^{d+2} \xrightarrow{U} \right)^m$. The instance All graph H_ϕ is defined in the following way:

- For all $1 \le i \le n$, add an edge $a_i \xrightarrow{R} b_i$.
- Add an edge $c_0 \xrightarrow{U} d_0$ and for each clause $1 \le j \le m$, an edge $c_j \xrightarrow{U} d_j$.
- For each clause $1 \le j \le m$ and variable X_i occurring in that clause, let p be the number of this occurrence of X_i in the formula (i.e., the occurrence of X_i in the j-th clause is the p-th occurrence of X_i), with $1 \le p \le d$ by assumption on ϕ. Then add a path of length p of R-edges from d_{j-1} to a_i and a path of length $(d+1) - p$ of R-edges from b_i to c_j.

The construction of G_ϕ and H_ϕ is in $O(|\phi|)$. Furthermore, notice the following ($\star$). For any $1 \le i \le n$, the edge $e = a_i \xrightarrow{R} b_i$ has at most d incoming R-paths and d outgoing R-paths; the outgoing paths have pairwise distinct *length* (i.e., the number of edges until the next edge is a U-edge), and likewise for the incoming paths. What is more, each incoming R-path of length p corresponds to an outgoing path of length $(d+1) - p$ and together they connect some d_{j-1} to some c_j via the edge e, where the j-th clause contains variable X_i.

Now, define $(e_1, \ldots, e_n)$ to be precisely the edges of the form $a_i \xrightarrow{R} b_i$ for every $1 \le i \le n$. Intuitively, the presence or absence of each of these edges corresponds to the valuation of each variable in ϕ. We claim that $\mathsf{Prov}_{H_\phi}^{G_\phi}$ represents ϕ on $(e_1, \ldots, e_n)$. It will suffice to show that there is a bijection between the satisfying valuations of ϕ, and the subgraphs of H_ϕ that both (i) contain all the edges not in $(e_1, \ldots, e_n)$, as these are fixed to 1, and (ii) admit a homomorphism from G_ϕ.

Indeed, consider the bijection defined in the obvious way: keep the edge $a_i \xrightarrow{R} b_i$ iff X_i is assigned to true in the valuation. First suppose that some valuation of $\{X_1, \ldots, X_n\}$ satisfies ϕ. Then, for each clause $1 \le j \le m$, there is a variable in the clause which evaluates to true. We build a match of G_ϕ on the corresponding possible world of H_ϕ by mapping the

j-th U-edge to $c_j \xrightarrow{U} d_j$ for all $0 \leq j \leq m$, and mapping the R-paths for each $1 \leq j \leq m$ by picking a variable X_i witnessing that the clause is satisfied and going via the path of length $1 + (p) + ((d+1) - p) = d + 2$ that uses the edge $a_i \xrightarrow{R} b_i$, which is present by assumption.

Conversely, assume that we have a match of G_ϕ on a possible world of H_ϕ. We show that the corresponding valuation satisfies ϕ. Consider the edge $c_j \xrightarrow{U} d_j$ to which the first U-edge is mapped. The R-path that follows must be mapped to a path from d_j to some a_i, and then take the edge $a_i \xrightarrow{R} b_i$, whose presence witnesses that the corresponding variable X_i is true. But importantly, in order for the path to have length precisely $d + 2$ before reaching the next U-edge, it must be the case that the length of the path before and after the edge $a_i \xrightarrow{R} b_i$ sums up to $d + 1$. As a result of ($\star$), this is only possible by taking a path that leads to $c_{j+1} \xrightarrow{U} d_{j+1}$, and so we know that variable X_i occurs in the j-th clause so that clause is satisfied. Repeating the argument shows that all clauses from the j-th onwards are satisfied, and as we have $m + 1$ U-edges in the graph H_ϕ and $m + 1$ U-edges in the graph G_ϕ we know that in fact we must have mapped the first U-edge to the first U-edge (i.e., $j = 0$), and all clauses are satisfied. ◁

By [31, Theorem 2], counting the independent sets of a graph of maximal degree 6 admits an FPRAS only if RP = NP. It is not hard to see that this problem is equivalent to counting satisfying assignments of a monotone 2-CNF formula in which a variable can appear in up to 6 clauses (see, for example, [25, Proposition 1.1]). Thus, we can apply Claim 3.4 above for the class of formulas in which $d = 6$ to obtain (deterministic) graphs G_ϕ and H_ϕ, and then build a probabilistic graph H'_ϕ identical to H_ϕ, in which the edges $(e_1, \ldots, e_n)$ are assigned probability 0.5 and all other edges probability 1, giving the desired reduction. ◀

DWT on DWT. Having classified the cases of one-way path queries (1WP) on all instances classes considered, we turn to more expressive queries. The next two query classes to consider are two-way path queries (2WP) and downward trees queries (DWT). For these query classes, exact computation on 2WP instances is tractable by [7], so the first case to classify is that of DWT instances. Exact computation is intractable in this case by [7], and we show here that, unfortunately, approximation is intractable as well, so that the border for exact tractability coincides with that for approximate tractability. We first focus on DWT queries:

▶ **Proposition 3.5.** $\mathsf{PHom}_{\mathsf{L}}(\mathsf{DWT}, \mathsf{DWT})$ *does not admit an FPRAS unless* RP = NP.

Proof. Our result hinges on the following, whose proof adapts [26, Proposition 2.4.3]:

▷ Claim 3.6. Given a monotone 2-CNF formula ϕ on n variables, we can build in time $O(|\phi| \log |\phi|)$ DWT graphs G_ϕ and H_ϕ, with the latter containing edges $(e_1, \ldots, e_n)$ such that $\mathsf{Prov}_{H_\phi}^{G_\phi}$ represents ϕ on $(e_1, \ldots, e_n)$.

Proof. Let $\phi = \bigwedge_{1 \leq i \leq m} (X_{f_1(i)} \vee X_{f_2(i)})$ be the input CNF instance over the variables $\{X_1, \ldots, X_n\}$. We let $L = \lceil \log_2 m \rceil$ be the number of bits needed to write clause numbers in binary. As we are in the labelled setting, let 0 and 1 be two distinct labels from the signature. Construct the query graph G_ϕ as follows:

- For all $1 \leq i \leq m$, add an edge $z \xrightarrow{0} x_i$.
- For each $1 \leq i \leq m$, letting $b_1 \cdots b_L$ be the clause number i written in binary, add a path of L edges $x_i \xrightarrow{b_1} y_{i,1} \xrightarrow{b_2} \ldots \xrightarrow{b_{L-1}} y_{i,L-1} \xrightarrow{b_L} y_{i,L}$.

Now, construct the DWT instance H_ϕ as follows:

- For all $1 \leq i \leq n$, add the edges $a \xrightarrow{0} c_i$.

- For all $1 \leq i \leq n$ and $1 \leq j \leq m$ such that X_i occurs in the j-th clause of ϕ (i.e., X_i is in $f_1^{-1}(j)$ or $f_2^{-1}(j)$), letting $b_1 \cdots b_L$ be the clause number j written in binary, add a path of L edges $c_i \xrightarrow{b_1} d_{i,j,1} \xrightarrow{b_2} \ldots \xrightarrow{b_{L-1}} d_{i,j,L-1} \xrightarrow{b_L} d_{i,j,L}$.

It is clear that $G_\phi \in \mathsf{DWT}$, $H_\phi \in \mathsf{DWT}$, and that both graphs can be built in time $O(|\phi| \log |\phi|)$. Now, define $(e_1, \ldots, e_n)$ to be the edges of the form $a \xrightarrow{0} c_i$ for every $1 \leq i \leq n$.

We claim that $\mathsf{Prov}_{H_\phi}^{G_\phi}$ represents ϕ on $(e_1, \ldots, e_n)$. It suffices to show that there is a bijection between the satisfying valuations ν of ϕ, and the subgraphs of H_ϕ that both (i) contain all the edges not in $(e_1, \ldots, e_n)$, as these are fixed to 1, and (ii) admit a homomorphism from G_ϕ. Indeed, consider the bijection defined in the obvious way: keep the edge $a \xrightarrow{T} c_i$ iff X_i is assigned to true in the valuation. First, if there is a homomorphism from G_ϕ to such a subgraph, then the root z of the query must be mapped to a (since this is the only element with outgoing paths of length $L + 1$ as prescribed by the query), and then it is clear that the image of any such homomorphism must take the form of a DWT instance that contains, for each clause number $1 \leq i \leq m$, a path of length L representing this clause number. This witnesses that the valuation ν makes a variable true which satisfies clause i. Hence, ν is a satisfying assignment of ϕ. Conversely, for every satisfying assignment ν, considering the corresponding subgraph of H_ϕ, we can construct a homomorphism mapping the edges of G_ϕ to the edges of H_ϕ, by mapping the path of every clause to a path connected to a variable that witnesses that this clause is satisfied by ν. ◁

The result then follows by an argument analogous to the one in Proposition 3.3. ◀

2WP on DWT. We then move to 2WP queries:

▶ **Proposition 3.7.** $\mathsf{PHom}_{\mathsf{L}}(\mathsf{2WP}, \mathsf{DWT})$ *does not admit an FPRAS unless* $\mathsf{RP} = \mathsf{NP}$.

This result follows from a general reduction technique from DWT queries on DWT instances to 2WP queries on DWT instances, which allows us to conclude using the result already shown on DWT queries (Proposition 3.5). We note that this technique could also have been used to simplify the proofs of hardness of exact computation in [7] and [2]. We claim:

▶ **Lemma 3.8.** *For any* DWT *query G, we can compute in time $O(||G||)$ a* 2WP *query G' which is equivalent to G on* DWT *instances: for any* DWT *H, there is a homomorphism from G to H iff there is a homomorphism from G' to H.*

For lack of space, we give only the construction of G' here, and defer the full proof of the correctness of this construction to Appendix B.

Proof. Let G be a DWT query. We build G' following a tree traversal of G. More precisely, we define the translation inductively as follows. If G is the trivial query with no edges, then we let the translation of G be the trivial query with no edges. Otherwise, let x be the root of G, let $x \xrightarrow{R_1} y_1, \ldots, x \xrightarrow{R_n} y_n$ be the successive children, and call $G_1, \ldots, G_n$ the DWT subqueries of G respectively rooted at $y_1, \ldots, y_n$. We define the translation of G to be $\xrightarrow{R_1} G_1' \xleftarrow{R_1} \cdots \xrightarrow{R_n} G_n' \xleftarrow{R_n}$, where $G_1', \ldots, G_n'$ are the respective translations of G_1, ..., G_n. This translation is in linear time, and the translated query has twice as many edges as the original query. ◀

Lemma 3.8 allows us to conclude from Proposition 3.5, as it allows us to reduce in linear time (in combined complexity) the evaluation of a DWT query on a DWT probabilistic instance to the evaluation of an equivalent 2WP query on the same instance. This establishes that any approximation algorithm for 2WP queries on DWT instances would give an approximation for DWT queries on DWT instances, which by Proposition 3.5 is conditionally impossible.

These results complete Table 1, concluding the classification of the complexity of PHom in the labelled setting: all cases that were intractable for exact computation are also hard to approximate, with the notable exception of 1WP queries on DAG instances.

4 Results in the Unlabelled Setting

We now turn to the *unlabelled* setting of probabilistic graph homomorphism, where the signature σ has only one label ($|\sigma| = 1$). Our results are summarized in Table 1b: we settle all cases except $\mathsf{PHom}_{\not L}(\mathsf{1WP}, \mathsf{All})$ and $\mathsf{PHom}_{\not L}(\mathsf{DWT}, \mathsf{All})$, for which we do not give an FPRAS or hardness of approximation result. Note that both problems are #P-hard for exact computation [7]. Further, they are in fact equivalent, because DWT queries are equivalent to 1WP queries in the unlabelled setting (stated in [7] and reproved as Proposition 4.2 below).

1WP on DAG. We start with 1WP queries, and state the following:

▶ **Proposition 4.1.** $\mathsf{PHom}_{\not L}(\mathsf{1WP}, \mathsf{DAG})$ *is #P-hard already in data complexity, but it admits an FPRAS.*

The positive result directly follows from the existence of an FPRAS in the labelled setting, which we have shown in the previous section (Proposition 3.1). By contrast, the #P-hardness does not immediately follow from previous work, as DAG queries were not studied in [7]. We can nevertheless obtain it by inspecting the usual #P-hardness proof of PQE for the CQ $\exists x\, y\ R(x), S(x, y), T(y)$ on TID instances [32]. We give a proof in Appendix C.

DWT on DAG. We can easily generalize the above result from 1WP queries to DWT queries, given that they are known to be equivalent in the unlabelled setting:

▶ **Proposition 4.2** ([7])**.** $\mathsf{PHom}_{\not L}(\mathsf{DWT}, \mathsf{DAG})$ *is #P-hard already in data complexity, but admits an FPRAS.*

This is implicit in [7, Proposition 5.5]: we give a self-contained proof in Appendix D.

2WP on PT. In contrast to 1WP queries, which are exactly tractable on PT instances and admit an FPRAS on DAG instances, 2WP queries have no FPRAS already on PT instances:

▶ **Proposition 4.3.** $\mathsf{PHom}_{\not L}(\mathsf{2WP}, \mathsf{PT})$ *does not admit an FPRAS unless* RP = NP.

Proof. It suffices to prove the claim below, which is the analogue to the unlabelled setting of Claim 3.6 after having transformed the query to 2WP via Lemma 3.8:

▷ Claim 4.4. Given a monotone 2-CNF formula ϕ on n variables, we can build in time $O(|\phi| \log |\phi|)$ an unlabelled 2WP graph G_ϕ and unlabelled PT graph H_ϕ, with the latter containing edges $(e_1, \ldots, e_n)$ such that $\mathsf{Prov}_{H_\phi}^{G_\phi}$ represents ϕ on $(e_1, \ldots, e_n)$.

We show this claim via a general-purpose reduction from the labelled setting to the unlabelled setting, which works in fact for All queries on All graphs. This reduction codes labels via specific unlabelled paths; a similar but ad-hoc technique was used to prove [7, Proposition 5.6]:

▶ **Lemma 4.5.** *For any constant $k \geq 2$, given a* All *query G and* All *graph H on a labelled signature with relation labels $\{1, \dots, k\}$, we can construct in linear time an unlabelled* All *query G' and* All *graph H' such that there is a (labelled) homomorphism from G to H iff there is an (unlabelled) homomorphism from G' to H'. Further, if G is a* 2WP *then G' is also a* 2WP*, and if H is a* PT *then H' is also a* PT*.*

Proof. We construct G' from G and H' from H by replacing every edge by a fixed path that depends on the label of the edge. Specifically, we consider every edge $x \xrightarrow{i} y$ of the query, where x is the source, t is the target, and $1 \leq i \leq k$ is the label. We code such an edge in G' by a path defined as follows: $x \rightarrow^{k+3} \leftarrow \rightarrow^{i+1} \leftarrow^{k+2} y$, where exponents denote repeated edges and where intermediate vertices are omitted. We code the instance H to H' in the same way. This process is clearly linear-time, and it is clear that if G is a 2WP then G' is also a 2WP, and that if H is a PT then H' is also a PT. Further, to establish correctness of the reduction, one direction of the equivalence is trivial: a homomorphism h from G to H clearly defines a homomorphism from G' to H' by mapping the coding in G' of every edge e of G to the coding of the image of e by h in H'

What is interesting is the converse direction of the equivalence. We establish it via a claim on homomorphic images of the coding of individual edges: for any $1 \leq i \leq k$, letting e' be the coding of an edge $e = x \xrightarrow{i} y$, for any homomorphism h' from e' to H', there must exist an edge $f = a \xrightarrow{i} b$ in H such that h' maps x to a and y to b. This claim implies the converse direction of the equivalence: if there is a homomorphism h' from G' to H', then applying the claim to the restrictions of h' to the coding of each edge of G, we see that h' defines a function h that maps the vertices of G to vertices of H, and that h is a homomorphism. Hence, all that remains is to prove the claim, which we do in the rest of the proof.

Consider an edge $e = x \xrightarrow{i} y$ as in the claim statement, and let e' be its coding and h' the homomorphism mapping e' to H'. Observe that, in H', the only directed paths of length $k+3$ are the first $k+3$ edges of the coding of edges of H. (This hinges on the fact that the paths of length $k+3$ defined in the coding of edges of H are never adjacent in H' to another edge that goes in the same direction, even across multiple edges, and no matter the directions of edges in H.) This means that, considering the directed path $\rightarrow^{k+3}$ at the beginning of e', there must be an edge $f = a \xrightarrow{j} b$ of H, with coding f' in H', such that the source x of e is mapped to the source a of f, and the first $k+3$ edges of e' are mapped to the first $k+3$ edges of f'. What remains to be shown is that $i = j$ and that y is mapped to b.

To this end, we continue studying what can be the image of e' into f'. After the directed path $\rightarrow^{k+3}$, the next edge $\leftarrow$ of e' must have been mapped forward to the next edge $\leftarrow$ of f': indeed, it cannot be mapped backwards on the last edge of the preceding path $\rightarrow^{k+3}$ because $k+3 > 1$ and $i+1 > 1$ so the next edges $\rightarrow^{i+1}$ would then have no image. Then the next directed path $\rightarrow^{i+1}$ of e' is mapped in f', necessarily forward because we fail if we map the first edge backwards: this implies that there at least as many edges going in that direction in f' as there are in e', i.e., $i \leq j$. Now, the last path $\leftarrow^{k+2}$ of e' cannot be mapped backwards because $k+2 > i+1$, so we must map it forwards in f': for this to be possible, we must have reached the end of the directed path $\rightarrow^{j+1}$ in f', so that we have $j = i$. We are now done reading e' and f', so we have indeed mapped y to b. This, along with $i = j$, establishes that the claim is true, and concludes the proof. ◀

We can thus prove Claim 4.4, starting from Claim 3.6 and translating it first via Lemma 3.8 and then via Lemma 4.5. Using the same argument as in Proposition 3.3, we conclude the proof of Proposition 4.3. ◀

5 DNNF Lower Bounds

In this section, we investigate how to represent the provenance of the query-instance pairs that we consider. More specifically, we study whether there exist polynomially-sized representations in tractable circuit classes of Boolean provenance functions Prov_H^G, for $G \in \mathcal{G}$ and $H \in \mathcal{H}$ in the graph classes studied in this paper. Certainly, for every graph class $\mathcal{G}$ and $\mathcal{H}$, the (conditional) non-existence of an FPRAS for $\mathsf{PHom}(\mathcal{G}, \mathcal{H})$ implies that, conditionally, we cannot compute nOBDD representations of provenance in polynomial time combined complexity – as otherwise we could obtain an FPRAS via Theorem 2.3. In fact, beyond nOBDDs, it follows from [9, Theorem 6.3] that, conditionally, we cannot tractably compute provenance representations even in the more general class of *structured DNNFs*. Indeed, as for nOBDDs, fixed edges in the reductions can be handled by conditioning [28, Proposition 4].

However, even in settings where there is conditionally no combined FPRAS, it could be the case that there are polynomial-*sized* tractable circuits that are difficult to compute, or that we can tractably compute circuits in a more general formalism such as *unstructured* DNNF circuits. The goal of this section is to give a negative answer to these two questions, for all of the non-approximable query-instance class pairs studied in Sections 3 and 4.

Specifically, we show moderately exponential lower bounds on the size of DNNF circuits for infinite families of graphs taken from these classes. Remember that DNNF is arguably the most general knowledge compilation circuit class that still enjoys some tractable properties [16]. Hence, these lower bounds imply that no tractable provenance representation exists in other tractable subclasses of DNNFs, e.g., structured DNNFs [28], or Decision-DNNFs [10]. We also emphasize that, unlike the intractability results of Sections 3 and 4 which assumed $\mathsf{RP} \neq \mathsf{NP}$, all of the DNNF lower bounds given here are unconditional.

We first show a *strongly* exponential lower bound for labelled $\mathsf{1WP}$ on All instances:

▶ **Proposition 5.1.** *There is an infinite family $G_1, G_2, \ldots$ of labelled* $\mathsf{1WP}$ *queries and an infinite family $H_1, H_2, \ldots$ of labelled* All *instances such that, for any $i > 0$, any DNNF circuit representing the Boolean function $\mathsf{Prov}_{H_i}^{G_i}$ has size $2^{\Omega(||G_i||+||H_i||)}$.*

Proof. By *treewidth* of a monotone 2-CNF formula, we mean the treewidth of the graph on the variables whose edges correspond to clauses in the expected way; and by *degree* we mean the maximal number of clauses in which any variable occurs. Let us consider an infinite family $\phi_1, \phi_2, \ldots$ of monotone 2-CNF formulas of constant degree $d = 3$ whose treewidth is linear in their size: this exists by [18, Proposition 1, Theorem 5]. We accordingly know by [4, Corollary 8.5] that any DNNF computing ϕ_i must have size $2^{\Omega(|\phi_i|)}$ for all $i > 1$. Using Claim 3.4, we obtain infinite families $G_1, G_2, \ldots$ of $\mathsf{1WP}$ and $H_1, H_2, \ldots$ of All graphs such that $\mathsf{Prov}_{H_i}^{G_i}$ represents ϕ_i on some choice of edges, and we have $||G_i|| + ||H_i|| = O(|\phi_i|)$ for all $i > 0$ (from the running time bound). Now, any representation of $\mathsf{Prov_L}_{H_i}^{G_i}$ as a DNNF can be translated in linear time to a representation of ϕ_i as a DNNF of the same size, simply by renaming the edges $(e_1, \ldots, e_n)$ to the right variables, and replacing all other variables by the constant 1. This means that the lower bound on the size of DNNFs computing ϕ_i also applies to DNNFs representing $\mathsf{Prov}_{H_i}^{G_i}$, i.e., they must have size at least $2^{\Omega(|\phi_i|)}$, hence $2^{\Omega(||G_i||+||H_i||)}$ as we claimed. ◀

We now present lower bounds for the remaining non-approximable query-instance class pairs, which are not exponential but rather *moderately* exponential. This is because our encoding of CNFs into these classes (specifically, Claim 3.6, and its images by Lemma 3.8 and Lemma 4.5) do not give a linear, but rather linearithmic bound. We leave to future work the question of proving strongly exponential lower bounds for these classes, like we did in Proposition 5.1.

▶ **Proposition 5.2.** *For any $\epsilon > 0$, there is an infinite family $G_1, G_2, \ldots$ of labelled* DWT *queries and an infinite family $H_1, H_2, \ldots$ of labelled* DWT *instances such that, for any $i > 0$, any DNNF circuit representing the Boolean function $\mathsf{Prov}_{H_i}^{G_i}$ has size at least $2^{\Omega\left((||G_i||+||H_i||)^{1-\epsilon}\right)}$.*

Proof. The proof is identical to that of Proposition 5.1, except that we apply Claim 3.6: for all $i > 0$, $||G_i|| + ||H_i|| = O(|\phi_i| \log |\phi_i|)$. We perform a change of variables: if we write $y = |\phi_i| \log |\phi_i|$, then we can show that $|\phi_i| = e^{W(y)}$, where W denotes the Lambert W function [12]; equivalently $|\phi_i| = y/W(y)$ as the W function satisfies $W(z)e^{W(z)} = z$ for all $z > 0$. Thus, the lower bound of $2^{\Omega(|\phi_i|)}$ on DNNF representations of ϕ_i implies that any DNNF for $\mathsf{Prov}_{H_j}^{G_j}$ has size at least $2^{\Omega\left(\frac{||G_i||+||H_i||}{W(||G_i||+||H_i||)}\right)}$. In particular, as W grows more slowly than n^ϵ for any $\epsilon > 0$, this gives a bound of $2^{\Omega\left((||G_i||+||H_i||)^{1-\epsilon}\right)}$ for sufficiently large ϕ_j. ◀

The proof for the following two claims are analogous to that of Proposition 5.2, but using Lemma 3.8 (for the first result) and Claim 4.4 (for the second result):

▶ **Proposition 5.3.** *For any $\epsilon > 0$, there is an infinite family $G_1, G_2, \ldots$ of labelled* 2WP *queries and an infinite family $H_1, H_2, \ldots$ of labelled* DWT *instances such that, for any $i > 0$, any DNNF circuit representing the Boolean function $\mathsf{Prov}_{H_i}^{G_i}$ has size at least $2^{\Omega\left((||G_i||+||H_i||)^{1-\epsilon}\right)}$.*

▶ **Proposition 5.4.** *For any $\epsilon > 0$, there is an infinite family $G_1, G_2, \ldots$ of unlabelled* 2WP *queries and an infinite family $H_1, H_2, \ldots$ of unlabelled* PT *instances such that, for any $i > 0$, any DNNF circuit representing the Boolean function $\mathsf{Prov}_{H_i}^{G_i}$ has size at least $2^{\Omega\left((||G_i||+||H_i||)^{1-\epsilon}\right)}$.*

We finish by remarking that all of the lower bounds above apply to acyclic query classes (i.e., queries of treewidth 1), for which non-probabilistic query evaluation is well-known to be linear in combined complexity [36]. Thus, these results give an interesting example of query classes for which query evaluation is in linear-time combined complexity, but computing even a DNNF representation of query provenance is (moderately) exponential.

6 Consequences

In this section, we consider some corollaries and extensions to the results above.

Optimality of a Previous Result. Recall from the introduction that, as was shown in [34], PQE for self-join-free conjunctive queries of bounded hypertree width admits a combined FPRAS (in the general setting of probabilistic databases, rather than probabilistic graphs):

▶ **Theorem 1.3** (Theorem 1 of [34]). *Let Q be a self-join-free conjunctive query of bounded hypertree width, and H a tuple-independent database instance. Then there exists a combined FPRAS for computing the probability of Q on H, i.e., an FPRAS whose runtime is* $\mathsf{poly}(|Q|, ||H||, \epsilon^{-1})$, *where ϵ is the multiplicative error.*

Can a stronger result be achieved? Our Proposition 4.3 immediately implies the following:

▶ **Corollary 6.1.** *Assuming* RP $\neq$ NP, *even on a fixed signature consisting of a single binary relation there is no FPRAS to approximate the probability of an input treewidth-1 CQ on an input treewidth-1 TID instance.*

Hence, tractability no longer holds with self-joins. So, as unbounded hypertree width queries are intractable in combined complexity even for *deterministic* query evaluation, we have:

▶ **Corollary 6.2.** *The result in Theorem 1.3 is optimal in the following sense: relaxing either the self-join-free or bounded-hypertree-width condition on the query implies the non-existence of a combined FPRAS, unless* RP = NP.

Network Reliability. The two-terminal network reliability problem asks the following: given a graph with probabilistic edges and with source and target vertices s and t, compute the probability that s and t remain connected, assuming independence across edges. Valiant showed that this problem is #P-complete [33, Theorem 1], and Provan and Ball showed that this holds already on directed acyclic graphs [29, Table 1]. Hardness also holds for the related problem of *all-terminal reliability* [29, Table 1], which asks for the probability that the probabilistic graph remains connected as a whole. Given the inherent #P-hardness of these problems, subsequent research has focused on developing tractable approximations.

Although significant progress has been made on FPRASes for all-terminal (un)reliability [19, 23], designing an FPRAS for two-terminal reliability has remained open. This question was even open for the restricted case of directed acyclic graphs; indeed, it was explicitly posed as an open problem by Zenklusen and Laumanns [37]. We now point out that the nOBDD construction of Proposition 3.1 implies an FPRAS for two-terminal reliability on DAGs, again by leveraging the approximate counting result of Arenas et al. [8]:

▶ **Theorem 6.3.** *There exists an FPRAS for the two-terminal network reliability problem over directed acyclic graphs.*

Proof. Given as input an unlabelled probabilistic DAG instance $H = (V, E)$ and two distinguished source and target vertices s and $t \in V$, construct the labelled DAG instance $H' = (V, E, \lambda)$ as follows. All vertices and edges are identical to that of H, but every edge of the form (s, x) emanating from s is assigned label $\lambda((s, x)) = R_s$, every edge (x, t) directed towards t is assigned label $\lambda((x, t)) = R_t$, and every other edge (x, y) is assigned the label $\lambda((x, y)) = R$. In the case that $(s, t) \in E$, then assign $\lambda((s, t)) = R'$.

Now, by the result in Proposition 3.1, we can construct an nOBDD for each of the following $|E|$ different labelled 1WP queries: $\xrightarrow{R'}$, $\xrightarrow{R_s}\xrightarrow{R_t}$, $\xrightarrow{R_s}\xrightarrow{R}\xrightarrow{R_t}$, $\dots$, $\xrightarrow{R_s}\left(\xrightarrow{R}\right)^{|E|-2}\xrightarrow{R_t}$. All of the nOBDDs have the same ordering (given by a topological ordering of the edges of H'), so we may take their disjunction to obtain a (complete) nOBDD D in linear time, whose accepting paths are in bijection with the (s, t)-connected valuations of the edges in H. From here we conclude by applying Theorem 2.3. ◀

We remark that, after submission of this work, Theorem 6.3 was also obtained independently (and with a different approach) in a recent preprint by Feng and Guo [17].

7 Conclusions and Future Work

We studied the existence and non-existence of *combined approximation algorithms* for the PQE problem, as well as the existence of polynomially-sized tractable circuit representations of provenance, under the lens of combined complexity.

We see several potential directions for future work. First, it would be interesting to see if the results in Proposition 3.1 and Theorem 6.3 can be extended beyond DAG instances: graph classes of bounded *DAG-width* [11] could be a possible candidate here. We also leave open the problem of filling in the two remaining gaps in Table 1. Namely, we would like to obtain either an FPRAS or hardness of approximation result for the equivalent problems $\mathsf{PHom}_{\not\subseteq}(\mathsf{1WP}, \mathsf{All})$ and $\mathsf{PHom}_{\not\subseteq}(\mathsf{DWT}, \mathsf{All})$. It is also natural to ask whether our results can be lifted from graph signatures to arbitrary relational signatures, or whether they apply in the *unweighted* setting where all edges are required to have the same probability [6, 1, 24]. Another question is whether we can classify the combined complexity of approximate PQE for *disconnected* queries, as was done in [7] in the case of exact computation, for queries that feature disjunction such as UCQs (already in the exact case [7]), or for more general query classes, e.g., with recursion [5].

References

1 Antoine Amarilli. Uniform reliability for unbounded homomorphism-closed graph queries. In *ICDT*, 2023. doi:10.4230/LIPIcs.ICDT.2023.14.
2 Antoine Amarilli, Pierre Bourhis, Mikaël Monet, and Pierre Senellart. Combined tractability of query evaluation via tree automata and cycluits. In *ICDT*, 2017. doi:10.4230/LIPIcs.ICDT.2017.6.
3 Antoine Amarilli, Pierre Bourhis, and Pierre Senellart. Provenance circuits for trees and treelike instances. In *ICALP*, 2015. doi:10.1007/978-3-662-47666-6_5.
4 Antoine Amarilli, Florent Capelli, Mikaël Monet, and Pierre Senellart. Connecting knowledge compilation classes and width parameters. *ToCS*, 2020. doi:10.1007/s00224-019-09930-2.
5 Antoine Amarilli and İsmail İlkan Ceylan. The dichotomy of evaluating homomorphism-closed queries on probabilistic graphs. *LMCS*, 2022. doi:10.46298/lmcs-18(1:2)2022.
6 Antoine Amarilli and Benny Kimelfeld. Uniform reliability of self-join-free conjunctive queries. *LMCS*, 2022. doi:10.46298/lmcs-18(4:3)2022.
7 Antoine Amarilli, Mikaël Monet, and Pierre Senellart. Conjunctive queries on probabilistic graphs: Combined complexity. In *PODS*, 2017. doi:10.1145/3034786.3056121.
8 Marcelo Arenas, Luis Alberto Croquevielle, Rajesh Jayaram, and Cristian Riveros. #NFA admits an FPRAS: efficient enumeration, counting, and uniform generation for logspace classes. *J. ACM*, 68(6), 2021. Extended version available as arXiv preprint arXiv:1906.09226 [cs.DS]. doi:10.1145/3477045.
9 Marcelo Arenas, Luis Alberto Croquevielle, Rajesh Jayaram, and Cristian Riveros. When is approximate counting for conjunctive queries tractable? In *STOC*. ACM, 2021. Extended version available as arXiv preprint arXiv:2005.10029 [cs.DS]. doi:10.1145/3406325.3451014.
10 Paul Beame, Jerry Li, Sudeepa Roy, and Dan Suciu. Exact model counting of query expressions: Limitations of propositional methods. *TODS*, 42(1), 2017. doi:10.1145/2984632.
11 Dietmar Berwanger, Anuj Dawar, Paul Hunter, Stephan Kreutzer, and Jan Obdrzálek. The DAG-width of directed graphs. *J. Comb. Theory, Ser. B*, 102(4), 2012. doi:10.1016/j.jctb.2012.04.004.
12 Robert M. Corless, Gaston H. Gonnet, D. E. G. Hare, David J. Jeffrey, and Donald E. Knuth. On the lambert W function. *Adv. Comput. Math.*, 5(1), 1996. doi:10.1007/BF02124750.
13 Nilesh N. Dalvi and Dan Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, 2004. doi:10.1016/B978-012088469-8.50076-0.
14 Nilesh N. Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6), 2012. doi:10.1145/2395116.2395119.
15 Adnan Darwiche. Decomposable negation normal form. *J. ACM*, 48(4), 2001. doi:10.1145/502090.502091.
16 Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *J. Artif. Intell. Res.*, 17, 2002. doi:10.1613/jair.989.
17 Weiming Feng and Heng Guo. An FPRAS for two terminal reliability in directed acyclic graphs, 2023. doi:10.48550/arXiv.2310.00938.
18 Martin Grohe and Dániel Marx. On tree width, bramble size, and expansion. *J. Comb. Theory, Ser. B*, 99(1), 2009. doi:10.1016/j.jctb.2008.06.004.

19 Heng Guo and Mark Jerrum. A polynomial-time approximation algorithm for all-terminal network reliability. *SIAM J. Comput.*, 48(3), 2019. doi:10.1137/18M1201846.

20 Tomasz Imielinski and Witold Lipski Jr. Incomplete information in relational databases. *J. ACM*, 31(4), 1984. doi:10.1145/1634.1886.

21 Abhay Kumar Jha and Dan Suciu. Knowledge compilation meets database theory: Compiling queries to decision diagrams. *ToCS*, 52(3), 2013. doi:10.1007/s00224-012-9392-5.

22 Ravi Kannan. Markov chains and polynomial time algorithms. In *FOCS*. IEEE, 1994. doi:10.1109/SFCS.1994.365726.

23 David R. Karger. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. *SIAM Rev.*, 43(3), 2001. doi:10.1137/S0036144501387141.

24 Batya Kenig and Dan Suciu. A dichotomy for the generalized model counting problem for unions of conjunctive queries. In *PODS*, 2021. doi:10.1145/3452021.3458313.

25 Jingcheng Liu and Pinyan Lu. FPTAS for counting monotone CNF. In *SODA*. SIAM, 2015. doi:10.1137/1.9781611973730.101.

26 Mikaël Monet. *Combined complexity of probabilistic query evaluation. (Complexité combinée d'évaluation de requêtes sur des données probabilistes)*. PhD thesis, University of Paris-Saclay, France, 2018. URL: https://pastel.archives-ouvertes.fr/tel-01980366.

27 Mikaël Monet. Solving a special case of the intensional vs extensional conjecture in probabilistic databases. In *PODS*. ACM, 2020. doi:10.1145/3375395.3387642.

28 Knot Pipatsrisawat and Adnan Darwiche. New compilation languages based on structured decomposability. In *AAAI*. AAAI Press, 2008. URL: http://www.aaai.org/Library/AAAI/2008/aaai08-082.php.

29 J. Scott Provan and Michael O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.*, 12(4), 1983. doi:10.1137/0212053.

30 Pierre Senellart. Provenance in databases: Principles and applications. In *Reasoning Web*, volume 11810 of *LNCS*. Springer, 2019. doi:10.1007/978-3-030-31423-1_3.

31 Allan Sly. Computational transition at the uniqueness threshold. In *FOCS*. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.34.

32 Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011. ISBN: 978-1608456802. doi:10.2200/S00362ED1V01Y201105DTM016.

33 Leslie G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3), 1979. doi:10.1137/0208032.

34 Timothy van Bremen and Kuldeep S. Meel. Probabilistic query evaluation: The combined FPRAS landscape. In *PODS*. ACM, 2023. doi:10.1145/3584372.3588677.

35 Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *STOC*. ACM, 1982. doi:10.1145/800070.802186.

36 Mihalis Yannakakis. Algorithms for acyclic database schemes. In *VLDB*. IEEE Computer Society, 1981.

37 Rico Zenklusen and Marco Laumanns. High-confidence estimation of small s-t reliabilities in directed acyclic networks. *Networks*, 57(4), 2011. doi:10.1002/net.20412.

A Proof of Theorem 2.3

▶ **Theorem 2.3.** *Let D be an nOBDD, and $w : \mathsf{vars}(D) \to [0,1]$ be a rational probability function defined on the variables appearing in D. Then there exists an FPRAS for computing* $\mathsf{WMC}(D, w)$*, running in time polynomial in $||D||$ and w.*

Proof. We may assume without loss of generality that D contains no variable v such that $w(v) = 0$ or $w(v) = 1$, since any such variable can be dealt with in constant time by conditioning D accordingly. We will use the fact that for any positive integer n and set of variables $S = \{x_1, \dots, x_k\}$ such that $k \geq \lceil \log n \rceil + 1$, we can construct in time $O(k)$ a complete OBDD $C_n(x_1, \dots, x_k)$, implementing a "comparator" on the variables of S, that tests if the integer represented by the binary string $x_1 \dots x_k$ is strictly less than n (hence, $C_n(x_1, \dots, x_k)$ has precisely n satisfying assignments, for any permitted value of k).

By [4, Lemma 3.16], we may assume that D is complete; thus, there is a bijection between the models of the Boolean function captured by D, and the paths from the root to the 1-sink of D. Now, complete the following procedure for every variable label v_i with weight $w(v_i) = p_i/q_i$ appearing in D. Set $k = \lceil \log d \rceil + 1$, where $d = \max\{p_i, q_i - p_i\}$. Send the 1-edge emanating from every node $r \in D$ labelled with v_i to the OBDD $C_{p_i}(x_1, \dots, x_k)$ (where $x_1, \dots, x_k$ are fresh variables), redirecting edges to the 1-sink of $C_p(x_1, \dots, x_k)$ to the original destination of the 1-edge from n. Do the same for 0-edge from r, but with the OBDD $C_{q_i - p_i}(x_1, \dots, x_k)$. Observe that D remains a complete nOBDD. Moreover, it is not difficult to see that there are now exactly p_i paths from the root to the 1-sink of D that pass through the 1-edge emanating from r, and $q_i - p_i$ paths passing through the 0-edge.

After repeating this process for every variable in D, we may apply Theorem 2.2, before normalizing the result by the product of the weight denominators $\prod q_i$. ◀

B Proof of Lemma 3.8

▶ **Lemma 3.8.** *For any* DWT *query G, we can compute in time $O(||G||)$ a* 2WP *query G' which is equivalent to G on* DWT *instances: for any* DWT *H, there is a homomorphism from G to H iff there is a homomorphism from G' to H.*

Proof. Let G be a DWT query. We build G' following a tree traversal of G. More precisely, we define the translation inductively as follows. If G is the trivial query with no edges, then we let the translation of G be the trivial query with no edges. Otherwise, let x be the root of G, let $x \xrightarrow{R_1} y_1, \dots, x \xrightarrow{R_n} y_n$ be the successive children, and call $G_1, \dots, G_n$ the DWT subqueries of G respectively rooted at $y_1, \dots, y_n$. We define the translation of G to be $\xrightarrow{R_1} G'_1 \xleftarrow{R_1} \cdots \xrightarrow{R_n} G'_n \xleftarrow{R_n}$, where $G'_1, \dots, G'_n$ are the respective translations of $G_1, \dots,$ G_n. This translation is in linear time, and the translated query has twice as many edges as the original query. Note that we can also inductively define a homomorphism from G' to G mapping the first and last elements of G' to the root of G: this is immediate in the base case, and in the inductive claim we obtain suitable homomorphisms from each G'_i to each G_i by induction and combine them in the expected way.

We claim that, on any DWT instance H, there is a match of G mapping the root to a iff there is a match of G' mapping both the first variable and last variable of the path to a. One direction is clear: from the homomorphism presented earlier that maps G' to G, we know that any match of G in H implies that there is a match of G' in H mapping the first and last elements as prescribed. Let us show the converse, and let us actually show by induction on G a stronger claim: if there is a match of G' mapping the first variable of G' to a vertex a, then the last variable is also mapped to a and there is a match of G mapping the root variable to a. If G is the vacuous query, then this is immediate: a match of the empty query G' mapping the first variable to a must also map the last variable to a (it is the same variable), and we conclude. Otherwise, let us write x the root of G and $x \xrightarrow{R_1} y_1, \dots, x \xrightarrow{R_n} y_n$ be the children and $G_1, \dots, G_n$ the subqueries as above. We know that the match of G' maps the first variable to a vertex a, and as H is a DWT instance it maps x to a child a_1 of a. Considering G_1 and its translation G'_1, we notice that we have a match of G'_1 where the first variable is mapped to a_1. Hence, by induction, the last variable is also mapped to a_1, and we have a match of G_1 where the root variable is mapped to a_1. Now, as H is a DWT instance, the next edge $\xleftarrow{R_1}$ must be mapped to the edge connecting a and a_1, so that the next variable in G' is mapped to a. Repeating this argument for the successive child edges and child queries in G, we conclude that the last variable of G' is mapped to a, and we obtain matches of $G_1, \ldots, G_n$ that can be combined to a match of G. ◀

C Proof of Proposition 4.1

▶ **Proposition 4.1.** $\mathsf{PHom}_{\not L}(\mathsf{1WP}, \mathsf{DAG})$ *is* $\#\mathsf{P}$*-hard already in data complexity, but it admits an FPRAS.*

Proof. As mentioned in the main text, the positive result directly follows from the existence of an FPRAS in the labelled setting, which was shown in Proposition 3.1. It remains to show $\#\mathsf{P}$-hardness here. Consider the following reduction from the $\#\mathsf{P}$-hard problem ♯BIS, which asks to count the independent sets of a bipartite undirected graph $B = (X, Y, E_G)$. We reduce to $\mathsf{PHom}_{\not L}(G, \mathsf{DAG})$, where G is fixed to be the $\mathsf{1WP}$ of length three, i.e., $\rightarrow\rightarrow\rightarrow$.

Construct a probabilistic graph $H = (\{s, t\} \sqcup X \sqcup Y, E_H)$ and probability labelling π, where s and t are fresh vertices, and the edge set E_H comprises:

- a directed edge $s \rightarrow a$ for every $a \in X$, with probability 0.5;
- a directed edge $a \rightarrow b$ for every $(a, b) \in E_G$, with probability 1 (i.e., the original bipartite graph instance, with every edge directed from X towards Y);
- a directed edge $b \rightarrow s$ for every $b \in Y$, with probability 0.5.

We claim that $\Pr_\pi(G \rightsquigarrow H)$ is precisely the number of independent sets of B, divided by $|X \sqcup Y|$. Indeed, just consider the natural bijection between the subgraphs of H and the independent sets of B, where for $a \in X$ we keep the edge $s \rightarrow a$ iff a is in the independent set, and for $b \in Y$ the edge $b \rightarrow s$ iff b is in the independent set. ◀

D Proof of Proposition 4.2

▶ **Proposition 4.2** ([7]). $\mathsf{PHom}_{\not L}(\mathsf{DWT}, \mathsf{DAG})$ *is* $\#\mathsf{P}$*-hard already in data complexity, but admits an FPRAS.*

Proof. Hardness follows directly from Proposition 4.1, so we show the positive result here. Let G be a DWT query graph, and m its height, i.e., the length of the longest directed path it contains. Let G' be this $\mathsf{1WP}$ of length m, computable in polynomial time from G. We claim the following.

▷ Claim D.1. For any $H \in \mathsf{DAG}$, $\mathsf{PHom}_{\not L}(G, H) = \mathsf{PHom}_{\not L}(G', H)$.

Proof. Certainly, if $H' \subseteq H$ admits a homomorphism from G, then it admits one from G' too since $G' \subseteq G$. On the other hand, if H' admits a homomorphism from G', then it also admits one from G: just map all vertices of distance i from the root of G to the image of the i-th vertex of G'. ◁

Now the result follows from the FPRAS for $\mathsf{PHom}_{\not L}(G', \mathsf{DAG})$ given by Proposition 3.1. ◀